Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Méréstechnika és Információs Rendszerek Tanszék

Nyéki Lajos

ANALÓG FÉNYTECHNIKAI ESZKÖZÖK VEZÉRLÉSE DMX JELLEL

KONZULENSEK

Bogár István,
Dr. Sujbert László

BUDAPEST, 2012

Tartalomjegyzék

HALLGATÓI NYILATKOZAT

Alulírott **Nyéki Lajos**, szigorló hallgató kijelentem, hogy ezt a szakdolgozatot meg nem engedett segítség nélkül, saját magam készítettem, csak a megadott forrásokat (szakirodalom, eszközök stb.) használtam fel. Minden olyan részt, melyet szó szerint, vagy azonos értelemben, de átfogalmazva más forrásból átvettem, egyértelműen, a forrás megadásával megjelöltem.

Hozzájárulok, hogy a jelen munkám alapadatait (szerző(k), cím, angol és magyar nyelvű tartalmi kivonat, készítés éve, konzulens(ek) neve) a BME VIK nyilvánosan hozzáférhető elektronikus formában, a munka teljes szövegét pedig az egyetem belső hálózatán keresztül (vagy autentikált felhasználók számára) közzétegye. Kijelentem, hogy a benyújtott munka és annak elektronikus verziója megegyezik. Dékáni engedéllyel titkosított diplomatervek esetén a dolgozat szövege csak 3 év eltelte után válik hozzáférhetővé.

Kelt: Budapest, 2012. 05. 15.

.....

Nyéki Lajos

Kivonat

Szakdolgozatom egy olyan színpadvilágításban használható fénytechnikai vezérlő tervezésének lépéseit mutatja be, mely alkalmas egy DMX 512A szabvány szerint működő rendszerbe illesztésre és vezérelni tud analóg 0-10V-os vezérlőjellel működő füstgépet és stroboszkópot.

A munka első fázisában irodalomkutatást végeztem, és megvizsgáltam a megvásárolható hasonló eszközöket is, és az így szerzett ismeretekből kiindulva alkottam meg az eszköz specifikációját. Figyelembe véve, hogy egy prototípus-fejlesztésről van szó, csak egy-egy füstgépet és stroboszkópot képes kezelni. A forgalomban kapható termékek gyakran 16-32 eszközt is kezelnek, bár azokat első sorban dimmerek (fázishasításon alapuló fényerő szabályzók) vezérlésére tervezték, amelyekből általában jelentős mennyiségre van szükség.

A munka második fázisában elkészítettem a hardvertervet, mely magába foglalta a blokkvázlat, a kapcsolási rajz és a kétoldalas nyomtatott áramköri lemez huzalozásának megtervezését. Nagy hangsúlyt fektettem a megfelelő alkatrészek kiválasztására, és felhasználók általi könnyű kezelés megvalósítására.

A nyomtatott áramköri lemez legyártása után hozzákezdtem az alkatrészek beszerzéséhez és beültetéséhez. Ezek után a szoftverfejlesztés következett, amihez az Atmel Studio 6 fejlesztőkörnyezet használtam. A szoftvert C nyelven írtam, és figyeltem az áttekinthető kód készítésére.

Az elkészült eszköz továbbfejleszthető, mivel a hardver lehetővé teszi bonyolultabb szoftver készítését is. Mivel a nem használt lábak ki vannak vezetve, ezért új hardver is illeszthető hozzá, például egy numerikus billentyűzet. Az így keletkezett termék egy prototípus, mely nem elég robusztus egy sok példányban gyártott termékhez.

Abstract

This Bachelor Thesis presents the details of the design of a stage lightning control, that is able to link in a DMX 512A system and control old strobe and smoke machines using 0-10V analog control signal.

In the first phase of my work, I made literature research and I examined available similar products. With my new knowledge I prepared the specifications of the new equipment. I took into consideration, that I design a prototype, so it can control only one strobe and one smoke machine simultaneously. The available products are usually designed to control dimmers, and use 16-32 channels, because people need lots of them in a stage.

In the second phase of my work I designed the hardware, including block diagram, schematic and the layout design. I focused on selecting the right components, and designing for the ease of use.

After producing the printed circuit board, I purchased and soldered the components. Then I designed the software with Atmel Studio 6 integrated development environment. I wrote the code in C, and I have paid attention to make a clear code.

The completed equipment can be upgraded, because the hardware supports more complicated softwares. Since the unused pins are routed out, new hardware can be attached, for example a numeric keypad. This is a prototype, which is not enough good for producing more than one piece, because it is not robust enough.

1 Bevezetés

A Szakdolgozat-készítés tárgy keretében választott feladatom a fénytechnikában használatos DMX 512A szabványú digitális jelet fogadni tudó, analóg vezérlőjel kiadására alkalmas eszköz megtervezése volt. A feladatot az Önálló laboratórium tárgy során kezdtem el, amikor eljutottam a hardver megtervezéséig. Ebben a félévben a hardver legyártásával (NYÁK legyártatása, alkatrészek beszerzése, beforrasztása, áramkör felélesztése) és a vezérlő szoftver megírásával foglalkoztam. Így egy olyan eszközt sikerült alkotni, amely alkalmas 0-10V-os analóg jeleket fogadni képes eszköz (mint például analóg vezérlésű dimmer vagy füstgép), illetve triggerjelet váró berendezés (mint például stroboszkóp) vezérlésére.

Fontos volt a tervezés során a felhasználóval való kommunikáció kényelmes biztosítása, továbbá extra funkciók implementálása és a bővíthetőség lehetőségének meghagyása. Így az eszköz megfelelő szoftver esetén akár nagyon bonyolult vezérléseket is alkalmazhat. Az audio bemenet segítségével lehetőség nyílik a zene lüktetését paraméterként felhasználva vezérelni a kimeneteket.

A tervezési munka során megismerkedtem a hardvertervezés és a beágyazott szoftvertervezés tudományával, olyan ismeretekre tettem szert, mely segítségével már egy közepes komplexitású eszköz megtervezhető. A feladat elvégzése közben a folyamatosan felmerülő újabb és újabb problémákra kerestem a megoldást, mely szakmai fejlődésemet nagyban segítette.

2 Irodalomkutatás, hasonló eszközök

Viszonylag kevés szakkönyv foglalkozik a színpadvilágítás témakörével. Böröcz Sándor Színpadvilágítás című könyve [1] tűnt használhatónak, mely segítségével a kezdeti információk beszerezhetőek voltak. A részletesebb technikai megoldásokra az internet segítségét kellett hívni.

A nyomtatott áramkör tervezéséhez az Altium Designer szoftvert használtam. Ehhez a youtube-on található rengeteg tutorial videó, ami nagyon hasznos volt. [12…16]

A szoftvertervezéshez az Atmel honlapján található útmutatók, alkalmazás jegyzetek voltak nagy segítségemre [18…26] illetve néhány idegen nyelvű könyv és a szakirányomon (Beágyazott és ambiens rendszerek) elsajátított ismeretek.

2.1 AC Studio & Live

2009 őszén kerültem kapcsolatba a Simonyi Károly Szakkollégium hangosítással és stúdiófelvételekkel foglalkozó szakmai körével, az AC Studio & Live-val. Azóta a kör vezetője lettem, és sok mindent sikerült tanulnom. Mivel a kör a rendezvények lebonyolításának szinte minden technikusi feladatkörével foglalkozik, így a színpadvilágítás és fénytechnika se kerülhette el a figyelmemet. Itt kerültem közelebbi kapcsolatba a DMX vezérlésű és analóg fénytechnikai eszközökkel. Különösen nagy előny volt, hogy bármikor lemehettem kipróbálni az eszközöket, kísérletezgetni, hogy mi hogyan működik. Így jelentősen gyorsabban tudtam a szükséges tudást megszerezni, mint ha csak szakkönyveket olvastam volna. Idősebb tagok is szívesen segítettek a munkában.

A kör fénytechnikai eszközparkja elég változatos, sok eszköz kipróbálására volt lehetőségem. A DMX-es eszközeink között mozgófejes robotlámpák, mozgótükrös szkennerek, intelligens fénytechnikai effektek és egy ködgép van. Az analóg eszközök közül pedig füstgép és stroboszkóp állt rendelkezésre. Utóbbiakhoz saját vezérlők vannak, a DMX-es eszközökhöz pedig két vezérlő pultunk is van.

2.2 DMX 512A szabvány

A DMX 512A („Digital Multiplex with 512 pieces of information) a fénytechnikában használatos szabvány a digitális kommunikáció megvalósítására. Alapvetően az RS-485 soros kommunikációs szabvány fizikai rétegére alapul. A szimmetrikus jelvezetés és a csavart érpár alkalmazása miatt elég jó a közös módusú zavarelnyomása.

Egy csomag maximum 512 darab 8 bites információ (frame) elküldéséből áll. Ezeket a csomagokat a vezérlő folyamatosan küldi. A vezérelt eszközök fogadják ezeket az adatokat, és mindegyik rendelkezik egy báziscímmel, ami azt jeleni, hogy az 512 frame-ből melyik az első, ami az adott eszközre vonatkozik. Innentől kezdve az eszköz annyi adatot vesz figyelembe, amennyire fizikai felépítéséből adódóan szüksége van. Ezért a címzésnél ezt figyelembe kell venni. Pl. ha egy robotlámpának a kezdőcíme 120 és nyolc adatot vár, akkor a következő eszközt a láncban 128-as címre kell állítani.

Az elküldött csomag öt részből áll. Kezdetben egy úgynevezett „Break" jelsorozat van. Ezután jön egy „Mark After Break" jel. Ezek választják el egymástól a csomagokat. Ezután jön egy „Start Code", mely egy 00H értékű 8 bites soros csomag, start bittel és két stop bittel. Ezután egy „Mark Time Between Frames" nevezetű magas jelszint következik, mely elválasztja frame-eket egymástól. Ezek után jönnek a „Channel Code" frame-ek, melyek felépítése azonos a „Start Code" felépítésével.

A „Break" és a „Mark After Break" üzenetek hosszára a szabványban minimum feltételek vannak megadva. A fogadó eszköznek minimum 88 μs ideig kell a „Break" jelet és 8 μs ideig kell a „Mark After Break" jelet látni. Ezek azért fontosak, mert ez alapján tudja az eszköz kiszámolni a címzést.

A rendszer felépítése a következő: egy vezérlő pult adja a vezérlő jelet a rendszerbe, az eszközök erre a vonalra „daisy chain" rendszerrel vannak felfűzve. A szabvány megköveteli a 120 Ohmos hullámimpedanciájú kábelek használatát, és a lánc végén lezáró ellenállás alkalmazását. Egyszerűen Y-ozni a jeleket nem szabad, de mivel túl hosszú láncok esetén nagy lehet a késleltetés, és a jel erőssége is csökken, ezért alkalmaznak ún. DMX splitter nevű eszközt, mely biztonságosan osztja több láncra a jelet. Nagyon hosszú távolságok áthidalása esetén bizonyos távolságok után be kell iktatni erősítőket is a rendszerbe. [3,4,8,10]

2.3 Fénytechnikai eszközök

A fogalmak tisztázásaképpen bemutatok pár fénytechnikai eszközt, mely a dolgozatomban szerepel.

A stroboszkóp egy nagy teljesítményű reflektor, mellyel bizonyos időközönként villantanak egyet. Arra használják, hogy alapvetően sötét színpadon vagy tánctéren, pillanatok erejéig hirtelen világosságot csináljanak. Működésének lényege, hogy egy kondenzátor hirtelen kisülésekor van a villanás, a villanások között pedig tölt a kondenzátor. A stroboszkóp segítségével vizuálisan megállíthatók a mozgások vagy éppen visszafordíthatók. Legszemléletesebben ezt úgy lehet leírni, mint ha egy filmben egy autó kerekét nézzük. A valóságos forgási sebességtől és másodpercenkénti képek számától függ, hogy előre vagy hátra forogni vagy pedig állni látjuk a kereket. Ezt a jelenséget a filmekben is stroboszkópikus hatásnak nevezik. Ezzel a megoldással nagyon érdekes effekteket lehet létrehozni. [11]

A dimmer alatt a fényerő szabályzást értik a fénytechnikában. Legtöbbször ez fázishasítással működik. A hálózati 50 Hz-es szinuszos feszültséget a hullám megadott fázisáig nullán tartja, és csak utána kapcsol be. Ezt a megoldást halogén izzók esetén szokták használni. Fém halogén izzóknál ez nem megoldható, ott legtöbbször mechanikus szerkezeteket szoktak használni. LED-ek esetén, pedig pulzus szélesség modulációt (PWM) szoktak alkalmazni. [9]

A ködgép apró részecskéket porlaszt szét a levegőben, ezáltal ködszerű hatást ér el. A porlasztás legtöbbször ultrahang segítségével történik. Abban segít, hogy a lámpák fénycsóváját láthatóvá teszi. Ezzel szemben a füstgépet alapvetően arra használják, hogy eltakarjanak vele valamit. A füstöt az különbözteti meg a fénytechnikában a ködtől, hogy a füstön nem lehet átlátni. [6]

Az intelligens lámpák, pl. mozgófejes robotok, mozgótükrös szkennerek, leginkább a koncerteken és szórakozóhelyeken használatosak. Nagyon sok paraméterrel rendelkeznek. A legalapvetőbb funkcióik a fényerősség szabályzás, fénysugár mozgatása (tükör mozgatásával vagy a lámpatest mozgatásával, szervomotorok segítségével), gobó (egy ábrát vetít ki, amit tud forgatni, rázni), és a dichro szűrő (fényvisszaverődésen alapuló színszűrő, mint az olajréteg a vízen). [1]

A fénytechnikai eszközök vezérléséhez fényvezérlő pultot (lightning control console) használunk. Ennek a berendezésnek sok fajtáját használják. A legegyszerűbb változat az úgynevezett „preset board”, mely kettő vagy több féder bankból áll, amiket

„színeknek” neveznek. A bankon belül a féderek az egyes eszközökhöz vannak rendelve, annak egy-egy paraméterét állítják, pl. fényerősség. Egy adott féder minden bankban ugyanazt a paramétert jelenti. A fénypult kezelője így offline meg tudja tervezni a következő színt, majd át tud „úszni” a már megtervezett színre.

Eggyel bonyolultabb megoldást „memory consol”-nak hívják. Ezek a berendezések mára már szinte teljesen felváltották a preset boardokat. A különbség az, hogy a színeket el lehet menteni, és bármikor előhozni, így már az előadás előtt meg lehet tervezni az összes megvilágítást, és csak váltani kell a színek között. Ez a megoldás főleg színházakban terjedt el.

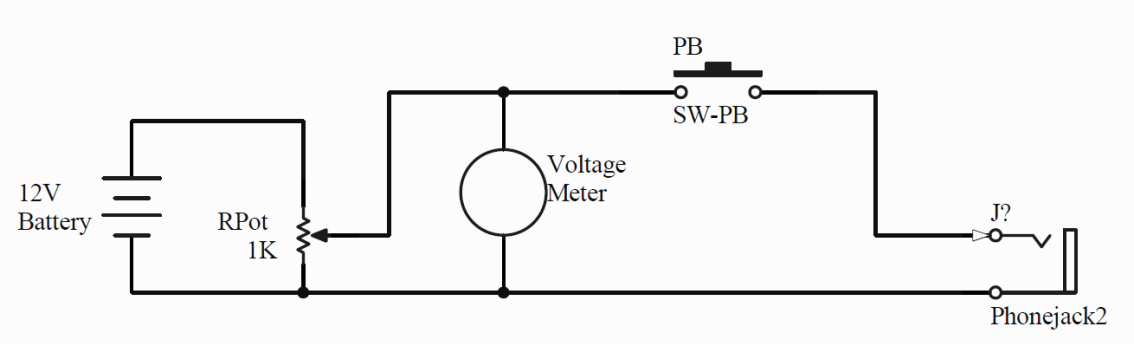
A bonyolultabb pultokba több intelligenciát építenek. Míg az előző megoldás főleg dimmerek vezérlésére volt alkalmas, addig az úgynevezett „moving light controller”-ek már mozgófejes és mozgótükrös robotlámpák kezelésére is kiválóan alkalmas. Ezeken a pultokon egy joystick segítségével mozgathatóak a lámpák, és a mozgás folyamatát is előre le lehet programozni és elmenteni. Főleg koncerteken és szórakozóhelyeken szokás alkalmazni, ahol előre megírt mozgásokat programoznak le.

Egy másik gyakran alkalmazott megoldás egy DMX-USB átalakító segítségével PC-ről vezérelni a fénytechnikai berendezéseket. Sok szoftver létezik erre az esetre, köztük drága fizetősek és open source megoldások is. Ilyen ingyenes szoftver pl. a Freestyler. [7]

2.4 Kísérletezések

Mivel alapvetően analóg eszközként dimmert, füstgépet és stroboszkópot ismertem, ezért ezekkel kezdtem el foglalkozni. A kollégium tulajdonában voltak ilyen eszközök, ezekkel kezdtem el kísérletezgetni.

Nehéz volt találni leírást ezekről az eszközökről, mivel már elég régiek, így azt tettem, hogy gyártottam egy 12V-os elemből, potenciométerből és kapcsolóból álló eszközt, melyhez hozzá kötöttem egy multimétert is. Így lehetett mérni, hogy mekkora feszültségű jelnél mit reagál az eszköz. Az eszköz kapcsolási rajza a 2-1. ábrán látható.



2-1. ábra: Mérőberendezés kapcsolási rajza

Alapvetően a füstgép a feszültség növelésével bekapcsolta az előfűtést, majd tovább növelve beindította a kompresszorát is, amivel a füstöt kijuttatja a levegőbe. A stroboszkópon ezzel szemben impulzus jellegű működést figyeltem meg. A 0-10V-os felfutó élre triggerelt, ennek pillanatában villant egyet.

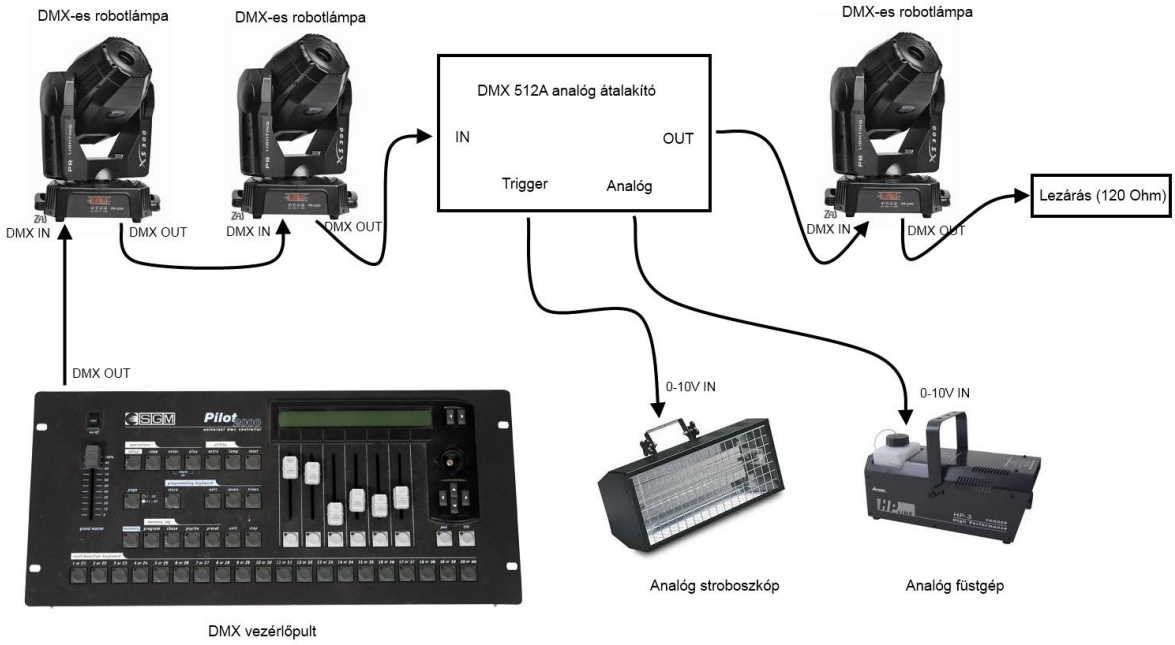
2.5 Hasonló eszközök

Léteznek olyan eszközök, melyek át tudják alakítani a DMX jeleket analóg 0-10V-os jelekké. Ezeket általában kis cégek gyártják kis sorozatban. Ezek az eszközök azonban mást általában nem csinálnak, minthogy a beérkező 8 bites adatot kiküldik analóg 0-10V-os feszültség formájában. Általában sok csatornát szoktak így meghajtani, ami így alkalmas mondjuk egy 24 csatornás régi dimmer analóg vezérlésére.

Példaként itt a Dezelectric által gyártott 24 csatornás DMX-512 / 0-10V -os átalakító. Bitkapcsoló segítségével bármely címtől kezdve vesz 24 csatornát. DMX jel vételét kontrol LED jelzi (amennyiben nem világít nincs DMX bejövő jel, vagy hibás). A tápfeszültség meglétét kontrol led jelzi (amennyiben nem világít, nincs tápfeszültség). A kimeneten a vett DMX érték (0-255) átalakított formában jelenik meg (arányosan 0-10V). Ajánlott felhasználás: bármely, max 24 csatornás 0-10V-os berendezés vezérelhető DMX-es pultról. [28]

Az általam tervezett eszköz annyiban tud ennél többet, hogy több lehetőséget biztosít a vezérlésre, több intelligenciát lehet vinni bele, köszönhetően az audio bemenetnek, ráadásul a trigger kimenet segítségével a stroboszkóp is nagyon egyszerűen vezérelhető. Tulajdonképpen ez egy átalakító és az egyszerűbb modern lámpákban található intelligencia egyesítése.

2.6 A DMX vezérlő elhelyezése egy rendszerben



2-2. ábra: Az eszköz egy teljes rendszerben elhelyezve

A 2-2. ábrán látható egy példa, hogy milyen módon lehetséges az általam tervezett eszköz elhelyezése egy felépített kisebb rendszerben. A DMX vezérlőpult kiadja a DMX 512A szabványú jelet, mely fel van fűzve az eszközökre a láncban. A DMX láncon négy eszköz van, három mozgófejes robotlámpa és az általam tervezett DMX-analóg átalakító. Az átalakítóval egy stroboszkópot és egy füstgépet vezérlek, melyek analóg 0-10V-os jellel vezérelhetőek.

Természetesen az eszköz ennél sokkal bonyolultabb rendszerekben is működőképes. Egy komplexebb rendszerben ezeken kívül előfordulhatnak más lámpák, mint pl. mozgótükrös szkennerek, dimmerek. Előfordulhat, hogy nem egy vezérlőpult, hanem egy PC adja ki a DMX jelet egy USB-DMX átalakító segítségével, és ha hosszú a lánc, előfordulhatnak DMX splitterek és DMX erősítők.

3 A hardvertervezés

A hardver megtervezéséhez az Altium Designer szoftvert használtam, mely remekül alkalmazható kétoldalas nyomtatott áramkörök tervezésére. [12…16]

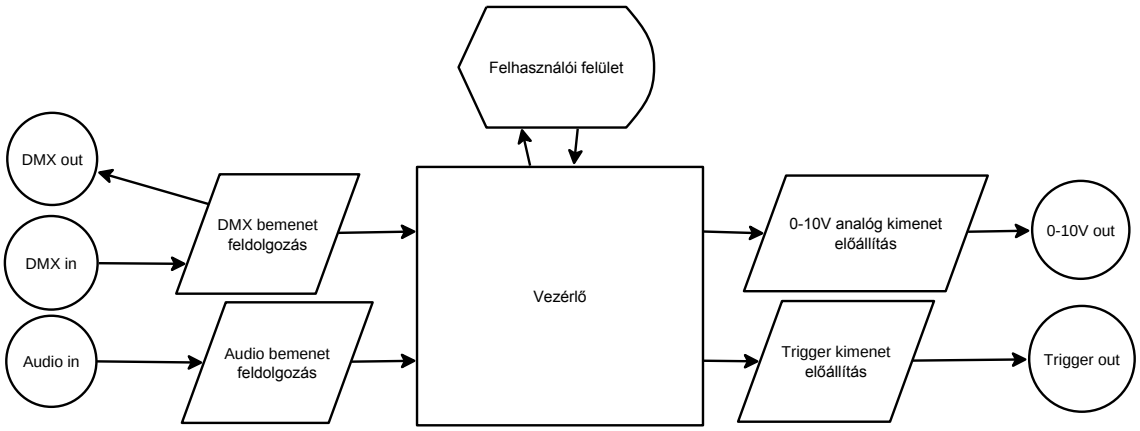
Első lépésként a blokkvázlatot terveztem meg. Ehhez papír-ceruza módszer is elegendő volt, majd digitális formában a Dia nevezetű ingyenes szerkesztőprogram segítségével készítettem el. Ez a program alkalmas különféle vektorgrafikus diagramok és ábrák készítésére, a szoftvertervezés során a folyamatábráknál is ezt használtam.

Ezután az alkalmazandó alkatrészeket kellett kiválasztani. A rendszert az Atmel cég ATmega128-as mikrokontrollere köré építettem, mivel ennek használatában már volt gyakorlatom, ugyanis a Méréstechnika és Információ Rendszerek tanszék saját fejlesztésű „MITmót” eszközének ilyen a vezérlője. Továbbá megfelelő számú I/O lábbal és memóriával rendelkezett ahhoz, hogy alkalmas legyen az eszköz megvalósítására. [19]

A műveleti erősítőknél a fontos szempont az volt, hogy lehessen +12V-os tápfeszültséget adni neki, a DAC átalakítónál, hogy lehetőleg ne legyen nagyon nagy csatornaszáma, mert felesleges, és soros kommunikációval lehessen tölteni rá programot. A többi alkatrésznél nem voltak speciális követelmények, csak az, hogy lehessen kapni Magyarországon.

Az alkatrészek folyamatos kiválasztásával párhuzamosan az Altium Designer programmal megterveztem a kapcsolási rajzot, majd pedig a nyomtatott áramköri lemez layout tervét.

3.1 Blokkvázlat



3-1. ábra: Blokkvázlat

A 3-1. ábrán látható blokkvázlaton megfigyelhetjük a hardver elvi felépítését. A körök a csatlakozókat, a paralelogrammák a periféria áramköröket jelölik. A vezérlő a DMX bemenet, az audio bemenet és a felhasználói felületről érkező információk alapján előállítja a 0-10V-os kimenetet és a trigger kimenetet.

A felhasználói felület sokféle paraméter megadására lehet alkalmas. A megfelelő számú nyomógomb és kijelző esetén akár bonyolult menürendszert is kezelhet. Ami biztos, hogy a DMX kezdőcímét itt kell megadni. A periféria áramkörök erősítéseket, szűréseket, DAC áramköröket, protokollillesztéseket és védelmet tartalmaznak.

Mivel ez egy fogadó eszköz, a DMX out természetesen a DMX in továbbfűzése, szükség esetén lezárással. [8]

3.2 Atmel ATmega128

A berendezés vezérlését az Atmel cég ATmega128 nevű mikrokontrollere végzi, amely egy 8 bites, Harvard architektúrás, RISC utasításkészletű vezérlő. Az utasítás-végrehajtás pipeline rendszerének köszönhetően egy utasítás egy órajel alatt hajtódik végre. Mivel maximálisan 16 MHz-es külső oszcillátor köthető rá, ezért 16 MIPS sebesség érhető el.

A vezérlő hat darab nyolcbites és egy ötbites I/O porttal rendelkezik, amelyek képesek közvetlenül is meghajtani LED-eket. Az I/O lábak iránya egyenként változtatható.

Minden porthoz tartozik három regiszter, a DDRx, a PORTx és a PINx, ahol x az adott port betűjele. A DDRx regiszter az x port egyes lábainak az irányát állítja. Ha

pl. a második bit „1” értékű, akkor az x port második lába kimenet lesz. A PORTx egy bemeneti lábon a felhúzó ellenállást állítja be, egyébként pedig a kimeneten a PORTx tartalma jelenik meg. Bemenet esetén a PINx regiszterből olvasható ki az adat.

A portok lábai általános célú I/O lábakként is használhatóak, de vannak másodlagos funkciói is. Ezek közül én az I^2C, UART, ADC funkciókat használom. A többi funkciót általános I/O lábon valósítom meg (pl. kijelző vezérlése, nyomógombok beolvasása, analóg trigger kimenet).

A mikrokontroller egy 10 bites AD konverterrel van felszerelve. Mivel 8 bites regiszterek vannak a vezérlőben, ezért az adat kettő regiszteren tárolódik (16 bit), és be lehet állítani, hogy balra vagy jobbra legyen igazítva az adat.

Három időzítő/számláló használható az időzített feladatokhoz. Ezek közül kettő 8 bites és egy 16 bites. [17,19,27]

3.3 Kapcsolási rajz

Az alapanyagok kutatása után elkezdődött a kapcsolási rajz összerakása. Ennek során figyelembe kellett venni, hogy milyen eszközök szerezhetőek be Magyarországon, milyen eszközök felelnek meg a specifikációnak, és milyen eszközöknek van footprintje a nyomtatott áramkör megtervezéséhez. Mivel ezek a feltételek ritkán teljesültek egyszerre, így adatlapok alapján szükség volt áramköri szimbólum és footprint szerkesztésre is.

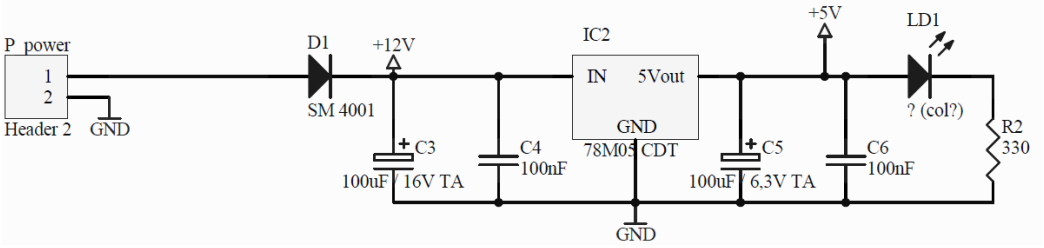
A mikrokontroller nem használt lábai is ki vannak vezetve bővíthetőség céljából. Négy DIP kapcsoló segítségével lehetőség nyílhat üzemmódok közötti manuális váltásra. Egy soros port is ki van vezetve egy MAX232 IC segítségével tesztelési célokra.

A hétszegmenses kijelző számjegyeinek időmultiplex vezérlése tranzisztorokkal van megoldva. A kijelző egy külön nyomtatott áramköri lemezen van megvalósítva.

Mivel az eszköz fix feszültségről működik, ezért az energiafelvétel változására áramváltozás következik be. A tápforrásban található feszültségstabilizátor feladata, hogy megfelelő árammennyiséget biztosítson stabil feszültség mellett, azonban megahertzes frekvenciájú változások esetén ezt nem tudja lekövetni. Ezért szükséges a táphoz és minden digitális áramkörhöz hidegítőkondenzátorokat tenni. Így amikor hirtelen áramigény keletkezik, a kondenzátorok tudják szolgáltatni az áramot. Fontos azonban, hogy ez a megoldás a terhelésváltozáskor bekövetkezett feszültség ingadozásokat nem tünteti el teljes mértékben, csak csillapítja azt.

Mivel a kondenzátorok a valóságban nem csak kapacitást jelentnek, hanem ellenállást és induktivitást is (dielektrikum és az alkatrészlábak illetve a kivezetés a tokozáson kívülre), ezért nem minden frekvencián alkalmasak a hidegítésre. Azért, hogy megnöveljük azt a frekvenciatartományt, ahol jól hidegít, egy kicsi és egy nagy kapacitású kondenzátor párhuzamos kapcsolását alkalmazzuk minden IC lábánál. A kicsi kapacitásúból kerámia kondenzátort, a nagy kapacitásúból tantál kondenzátort használtam. [29,30]

3.3.1 Táp

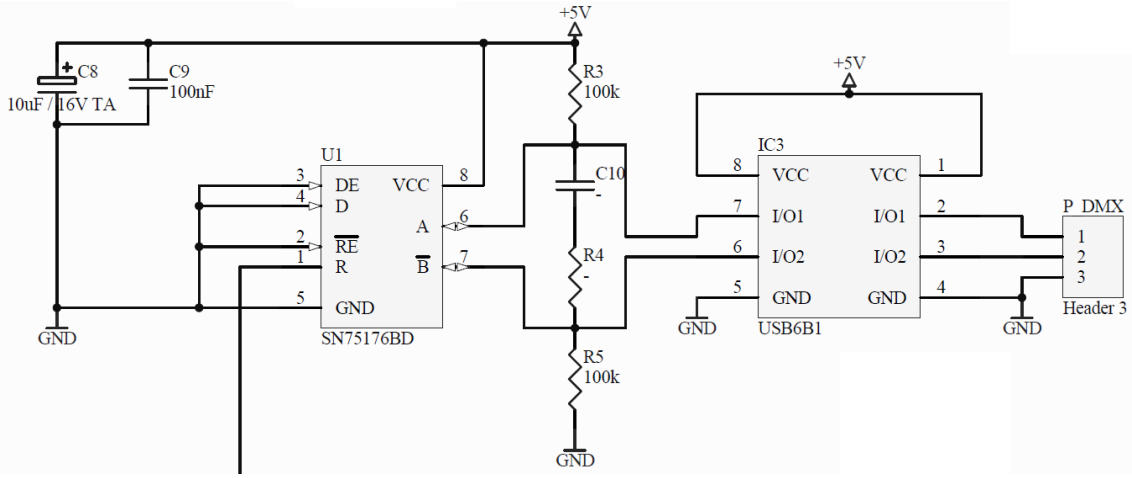


3-2. ábra: A táp kapcsolási rajza

Kétféle tápfeszültségre van szükség az áramkörben. A 12V-os feszültséget a kimenetek műveleti erősítői kapják meg, az 5V-os feszültség kell a többi helyre, melyet egy 78M05 feszültség stabilizáló IC-vel oldok meg a hozzá tartozó megfelelő hidegítőkondenzátorokkal. A LED megmutatja, hogy áram alatt van-e a berendezés. A D1-es dióda polaritás védelem miatt kell. Ennek a kapcsolási rajzát a 3-2. ábra mutatja.

A mikrokontroller lábánál egy LC taggal szűrjük a tápfeszültséget, hogy a beépített AD átalakítót a lehető legkevésbé zavarják a digitális áramkörök által keltett nagyfrekvenciás zavarok.

3.3.2 DMX bemenet



3-3. ábra: A DMX bemenet kapcsolási rajza

Nézzük a felépítést a DMX csatlakozó felől. Az USB6B1 USB vonalak védelmére szolgáló IC. Mivel az USB is differenciálisan szállítja az adatot, így megfelelő lesz védelemre. A C10 és R4 alkatrészek nem kerülnek beültetésre, csak hagytam nekik helyet, hogyha kellenének, lezárásra használhatóak legyenek.

Az SN75176BD RS-485-ös vonalillesztő IC gondoskodik arról, hogy a DMX adat helyesen legyen fogadva. Az adó lábak a földre vannak kötve, nehogy adni tudjon, mi csak vételre akarjuk használni. A vevő láb a mikrokontroller egyik lábára van kötve.

Természetesen, mint minden digitális IC-hez, itt is szükséges a hidegítőkondenzátorok használata.

A DMX bemenet kapcsolási rajza a 3-3. ábrán látható.

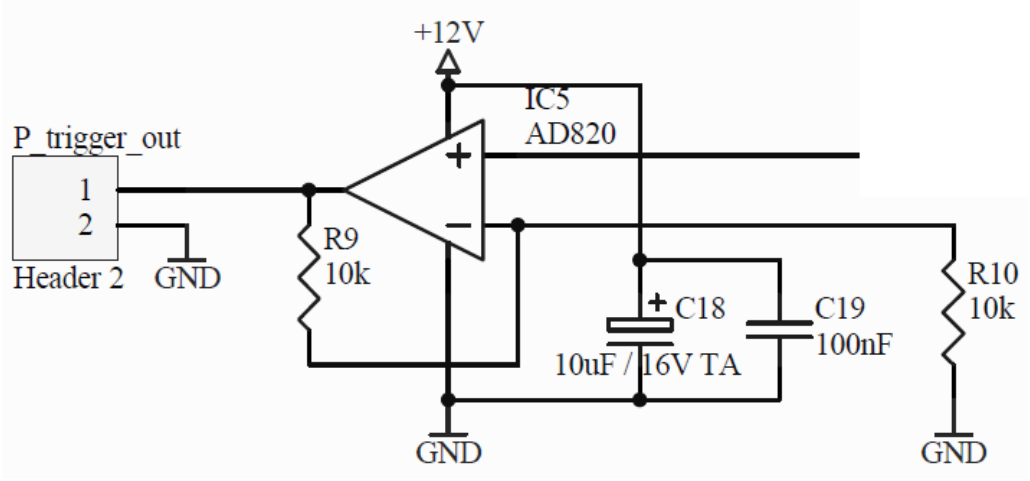
3.3.3 Analóg kimenetek

A kétféle analóg kimenet megoldásához szükség volt kétszeres erősítésre, ugyanis a mikrokontroller 5V-os tápot kap, ezért nem fog tudni 0-10V-os jelet kiadni magából. Így a 0-5V-ot erősítjük 12V-os táppal rendelkező nem invertáló kapcsolású műveleti erősítővel. Ebben az esetben az erősítés a következőképp alakul:

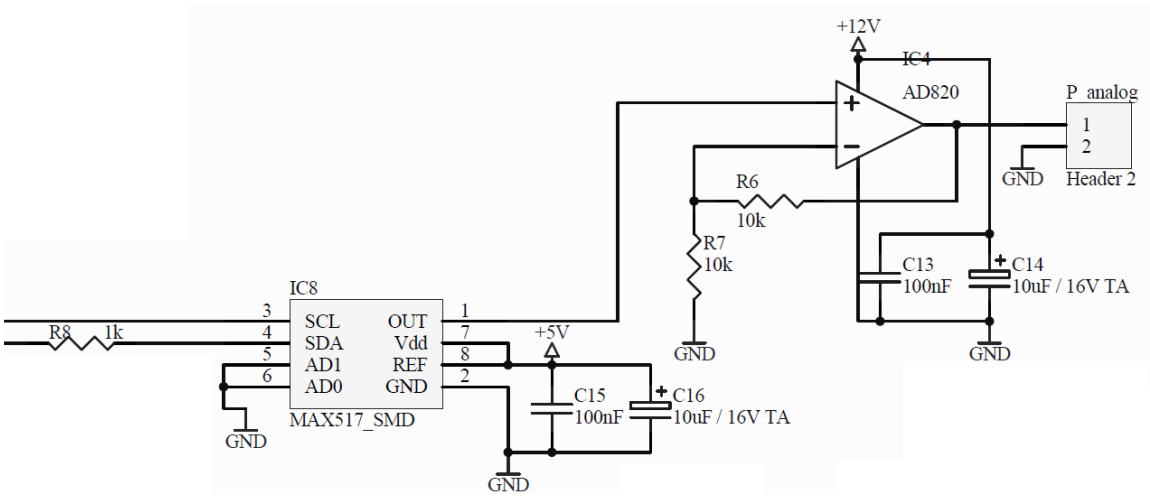
$$A = 1 + \frac{R_9}{R_{10}} = 1 + \frac{10k}{10k} = 2$$

A trigger kimenetnél másra nincs is szükség, ugyanis amikor a vezérlő logikai „0”-ról logikai „1”-re vált, akkor elő is állítja a 0-5V-os felfutó élt, ami kell nekünk. Kapcsolási rajzát megfigyelhetjük a 3-4. ábrán.

A másik analóg kimenetnél viszont egy nyolcbites számot kell kiadni analóg módon. Ehhez egy 5V-os referencia feszültségű DAC átalakítót használunk. Ez I2C buszon kommunikál a vezérlővel, és az így megkapott adatot alakítja át 0-5V-os analóg jellé. A 3-5. ábrán látható SDA lábra kötött 1k-s ellenállást az IC adatlapja ajánlotta.

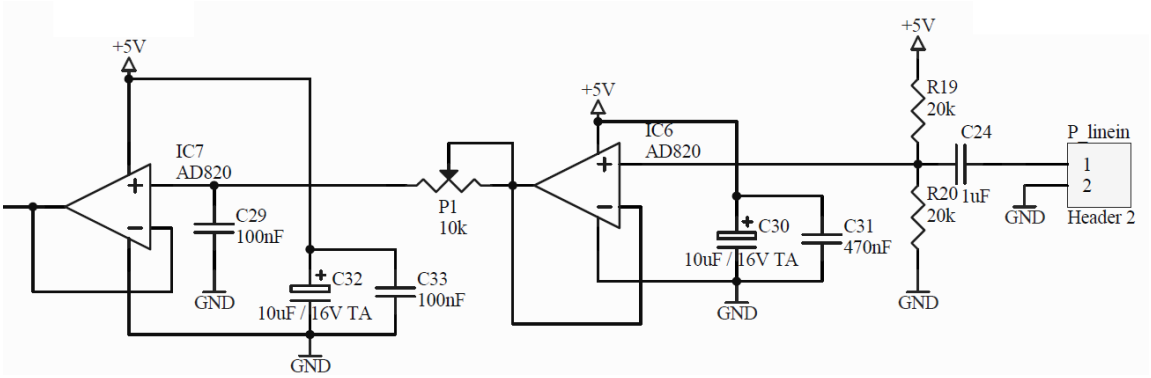


3-4. ábra: Trigger kimenet kapcsolási rajza



3-5. ábra: Analóg kimenet kapcsolási rajza DAC-vel

3.3.4 Audio bemenet



3-6. ábra: Audio bemenet kapcsolási rajza

Két nagy ellenállással beállítjuk a tápfeszültség felére a munkapontot, így nem lesz szükség negatív tápfeszültségre az erősítőnél. A bejövő jelből ehhez le kell választani az esetleges egyenáramú komponenst, ha valami oknál fogva lenne benne. Ezt egy elsőfokú felüláteresztő szűrővel oldom meg. Mivel a zene üteme, amihez szeretnénk igazítani a fényeket, tipikusan a mély tartományokban van jelen legkönnyebben kivehetően, ezért érdemes a magasakat is kiszűrni. Ezt egy elsőfokú aluláteresztő szűrő biztosítja, melynek vágási frekvenciáját a P1 potenciométer segítségével változtathatjuk. A két szűrő együtt egy sáváteresztő szűrőt hoz létre, melynek az átviteli karakterisztikája a következő:

$$\frac{U_{ki}}{U_{be}} = \frac{j\omega(R_{19} \times R_{20})C_{24}}{(1 + j\omega(R_{19} \times R_{20})C_{24})(1 + P_1 j\omega C_{29})}$$

Ez csomóponti potenciálok módszerével könnyen felírható, de jósága abból is látszik, hogy egy zérus van a nullában (ezt várjuk a felüláteresztő jelleg miatt), és két pólus, ami nyilván ahhoz kell, hogy az átviteli függvény meredeksége a 20 dB/dekádos meredekségből negatív (-20 dB/dekádos) meredekségre váltson az aluláteresztő jelleghez.

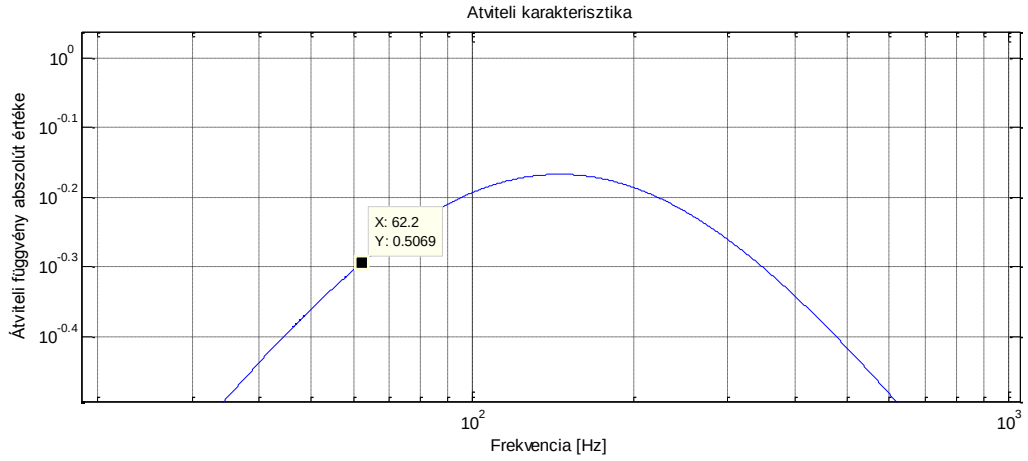
Mivel $R_{19} = R_{20}$, ezért a továbbiakban legyen $R = (R_{19} \times R_{20}) = \frac{R_{19}}{2} = \frac{R_{20}}{2}$

Az alábbi MATLAB kóddal kirajzolhatjuk az átviteli karakterisztikáját:

```
R=10000;
C24=10^-6;
C29=470*10^-9;
w=[0:0.1:20000];
P1=10000;
```

```
A=abs((j.*w.*R.*C24)./((1+j.*w.*R.*C24).*(1+j.*w.*P
     1.*C29)));
loglog(w,A)
title('Atviteli karakterisztika');
xlabel('Frekvencia [Hz]');
ylabel('Átviteli függvény abszolút értéke');
grid on;
```

Mivel P1=10 000 beállítást használtunk, ezért ez a legkisebb vágási frekvenciát mutatja. Akkor lehet érdemes kisebbre állítani, ha a vezérlő zene magasabb frekvenciája alapján akarunk vezérelni. Ez előfordulhat a stroboszkóp esetén, ha mondjuk nem a kísérethez, hanem valamilyen szóló ritmusához vagy pedig kísérőcinhez akarjuk igazítani a villogást. Megfelelő formázás után a3-7. ábrához jutunk:

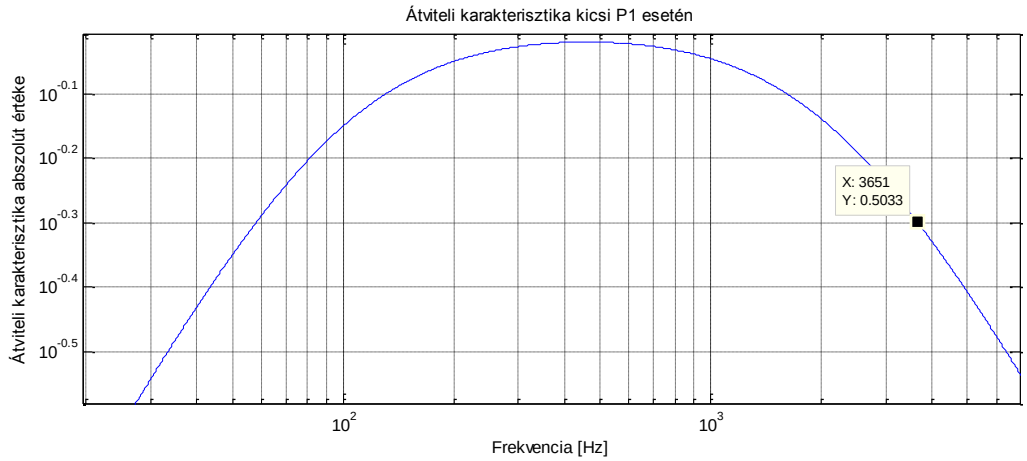


3-7. ábra: Az audio bemenet átviteli függvénye P1=10k esetén

Ebből jól látszik, hogy ha -3 dB-t vesszük az átvitel tartományának kezdeteként, akkor 62 Hz fölött átenged. Az aluláteresztő rész változtatható, a legkisebb állásban 340 Hz-nél kezdődik a záró tartomány.

Ha pl. a tizedére csökkentjük P1 értékét, akkor a következőt kapjuk

```
P1=1000;
A=abs((j.*w.*R.*C24)./((1+j.*w.*R.*C24).*(1+j.*w.*P
     1.*C29)));
loglog(w,A)
title('Átviteli karakterisztika kicsi P1 esetén');
xlabel('Frekvencia [Hz]');
ylabel('Átviteli függvény abszolút értéke');
grid on;
```



3-8. ábra: Az audio bemenet átviteli függvénye P1=1k esetén

A 3-8. ábrán látható, hogy itt már kHz-es tartományban található komponensekre is lehet vezérelni. Például egy kísérőcin már ebben a tartományban van.

3.3.5 Interface

Elsősorban a DMX cím beállítására alkalmas kijelző tartozik az eszközhöz, bár megfelelő szoftverrel sok más extra funkciókat is elláthat. A kijelző időmultiplex módon működik, melynek a lényege, hogy egyszerre csak egyik számjegy világít. Sorban megfelelő időzítéssel mindegyik számjegy felvillan. A felvillanások között eltelt idő olyan kicsi kell, hogy legyen, hogy az emberi szem felbontó képessége ne lássa villogásnak, ez nagyságrendileg századmásodperces villogást jelent.

A négy nyomógomb szolgál az adatok bevitelére. Kettővel lehet például léptetni a címet egyesével, a másik kettővel pedig nagyobb számokkal léptetni a gyors beállításhoz, de akár funkciók közötti választást is lehetővé tehet.

A négy LED főleg hiba visszajelzésre lehet használatos. Ha pl. nem kap DMX jelet vagy hangvezérlés van beállítva és mégse tudja feldolgozni a bemenő jelet.

Az interface kapcsolási rajza megtalálható a függelékben, az 5-3. ábrán.

3.4 Huzalozási rajz

A nyomtatott áramköri megavalósítás során két lemezen helyeztem el az áramköröket. Az egyik a vezérlőt tartalmazza, a másik a kijelzőt, nyomógombokat, és egyéb jelző LED-eket. Erre azért volt szükség, mert az utóbbi a készülék oldalán kell, hogy helyet foglaljon (ha pl. egy rackbe szerelhető verzióról gondolkozunk, az egész elektronika viszont nem férne el ott).

3.4.1 Vezérlő áramkör

Alapvetően 10 miles csíkszélességet használtam, kivéve ott, ahol a vastagabb indokolt volt nagyobb áramok miatt. Ezért a tápvonalakat 60 mil szélességgel húztam, kivéve ahol nem volt erre lehetőség (pl. IC lábára maximum 15-20 mil szélességgel lehet csak rámenni). A LED-ekhez, kijelzőhöz tartozó vezetékeken is nagyobb áramok folynak, mint a jelvezetékeken, ezért itt 20 miles vastagságot használtam. A panel bal oldalán levő csatlakozókhoz használtam 8 miles vastagságot, ahol a kis hely ezt indokolta.

Az alkatrészek moduláris rendszerben vannak elhelyezve, az összetartozó alkatrészek egy helyen vannak, így szemmel látható, hogy melyik blokk hol található a lemezen.

Tokozások esetében a felületszerelt alkatrészeket részesítettem előnyben, de a kapcsolókat és a csatlakozókat furatszerelten oldottam meg. Ennek oka egyrészt a nagyobb mechanikai stabilitás, másrészt a beszerezhetőség is ezt a verziót támogatta. Az ellenállások, kondenzátorok, tekercsek esetén főleg a 1206-os (tantál kondenzátor esetén „D”-s) tokozással dolgoztam. Valójában lehetett volna kisebbeket is használni, mivel annyira nagy áramok nem folynak ebben az eszközben. Mivel a mikrokontroller hidegítőkondenzátorait nem tudtam ekkora méretben elég közel rakni az IC-hez, ezért ott 0805-ös (tantál kondenzátor esetén „A”-s) tokozást használtam.

Az IC-k hidegítésénél figyeltem arra, hogy a lehető legközelebb tegyem az IC lábaihoz a hidegítőkondenzátorokat, a kettő közül a kisebbet közelebb téve. A kvarckristályt szintén elég közel kellett tenni a mikrokontrollerhez, ugyanis ilyen magas frekvencián már ezen a távon is parazita induktivitások jelennek meg.

A földet a hátsó oldalon elhelyezett polygonnal oldottam meg. Ez annyit jelent, hogy a szabad felületek a hátoldalon rézzel borítottak, így nagy felületen, kis ellenállással lehet bekötni a földet, továbbá a hőelvezetést is megoldja. A táp körül a hőelvezetés miatt van elég sok via a földre.

Rögzítés céljából mechanikai furatok lettek elhelyezve a lemezen, amik segítségével hozzá lehet csavarozni egy műszerdobozhoz. A huzalozási rajzok megtalálhatóak a függelékben, az 5-4., 5-5., 5-6., 5-7. ábrákon.

3.4.2 Előlap

Az előlap megtervezése hasonló megfontolásokkal történt, itt azonban nem használtam polygont.

3.5 Nyomtatott áramkör gyártása

A nyomtatott huzalozású lemezt a BME ETT tanszékén gyártattam le. Mivel erre várni kellett, míg elkészül, addig elkezdtem megírni a szoftvert.

4 A szoftvertervezés

A berendezésben szereplő ATmega128 feladata elvégezni a kijelző és a nyomógombok kezelését, a DMX jel fogadását, a hangbemenet feldolgozását, és az analóg kimenetek kiküldését. Ebben a fejezetben az ezt megvalósító szoftver tervezését mutatom be.

4.1 A szoftver megvalósult specifikációja

A szoftver végleges specifikációja úgy alakult, hogy képes legyen a DMX jel fogadására, és ez alapján előállítani 0-10V-os analóg vezérlőjeleket, melyből az egyiknél a jelszinttel a másiknál frekvenciával arányos a fogadott 8 bites adat. Ez alapján két csatornát foglal el a rendszerben, mely technikai okok miatt és a használati kényelmességét is figyelembe véve, egymás mellett helyezkednek el. Megfelelő adat esetén az eszköz át tud váltani hangvezérlésbe.

A kijelző mindig mutatja az eszköz első címét. A második ennél eggyel nagyobb. A nyomógombok segítségével pedig állítani lehet a címet.

4.2 A szoftverfejlesztő környezet leírása

A mikrokontroller szoftverét C nyelven írtam. Ehhez az Atmel Studio 6 fejlesztőkörnyezetet és WinAVR fordítóprogramot használtam.

Munkám során három forrásfájlt készítettem. Egyik a főciklust (DMX_AVR_main.c), a másik a függvények, interrupt rutinok deklarációit, definiált makrókat (DMX_AVR.h), a harmadik pedig a függvénydefiníciókat tartalmazza. Ennek köszönhetően áttekinthetőbb a kód.

A programot a berendezésre az Atmel cég JTAG In-Circuit Emulator mkII nevű eszközével töltöttem fel. Fontos lépés az úgynevezett fuse bitek helyes beállítása, pl. a külső órajel forrás beállítása, és a JTAG engedélyezése. [21]

4.3 Részletes implementáció

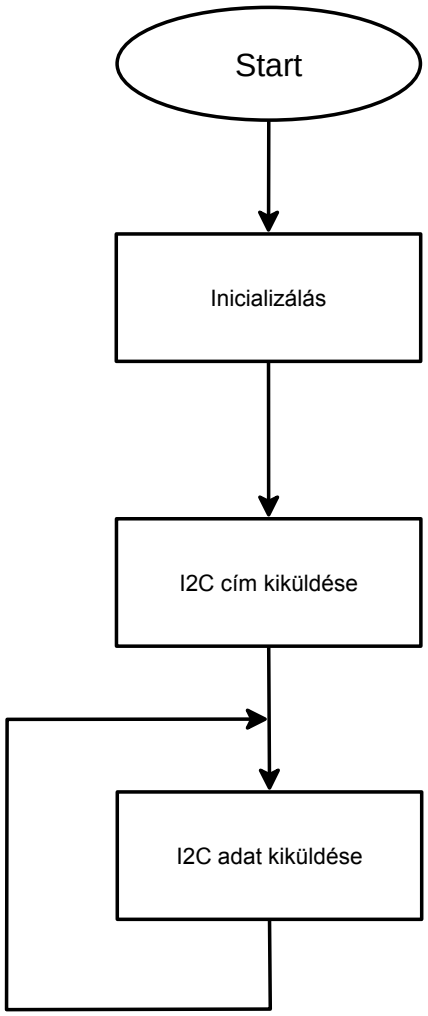
Az alapvető gondolata a szoftvernek, hogy a főciklusban inicializálások és az analóg kimenet vezérlésén kívül (I^2C buszon adatok küldése a DAC átalakítónak)

minden más interrupt rutinok segítségével működjön. Erre a pontos időzítések miatt volt szükség, továbbá egy lekérdezéses adatkapcsolat nagyon sok időt venne igénybe.

Igyekezetem sok függvényt használni, minden logikailag egybetartozó műveletsort kiemeltem egy külön függvénybe. Így az adott absztrakciós szinten atomi műveletek kerültek egy függvénybe, ami olvashatóbb, tisztább kódot eredményezett. Kommenteket csak azokon a helyeken használtam, ahol a kód másképp nem lenne értelmes (pl. egy regiszterbe betöltött konstans funkciója nem derül ki a kódból)

A szoftver egy Init_AVR() függvénnyel indul, ami az összes használt erőforrást (portokat, I^2C kommunikációt, időzítőket, AD konvertert, programban használt változókat) inicializál. Itt engedélyeződnek az interruptok is.

Sok globális változót használtam, mivel ezt tartottam a legegyszerűbbnek, hogy kommunikáljanak az interrupt rutinok egymással és a főciklussal.



4-1. ábra: A főciklus folyamatábrája

A 4-1. ábrán látható a főciklust ábrázoló folyamatábra. Ezt a folyamatot szakítják meg interruptok, amik biztosítják egyes feladatok pontos ütemezését, és a processzoridő jobb kihasználását. [17]

4.3.1 Inicializálás

Az inicializálás egy függvény segítségével történik, mely további inicializáló függvényeket hív meg. Itt inicializálódnak a portok (input/output, felhúzó ellenállások), időzítők, nem konstans változók (a konstansok a header fájlban vannak inicializálva), a soros kommunikáció, az analóg-digitális átalakító, és az I^2C kommunikáció.

A függvény ezen kívül engedélyezi az interruptokat, és kigyújtja az egyik LED-et a kijelzőn. Ezzel jelzi, hogy az eszköz inicializálva van, működésre kész.

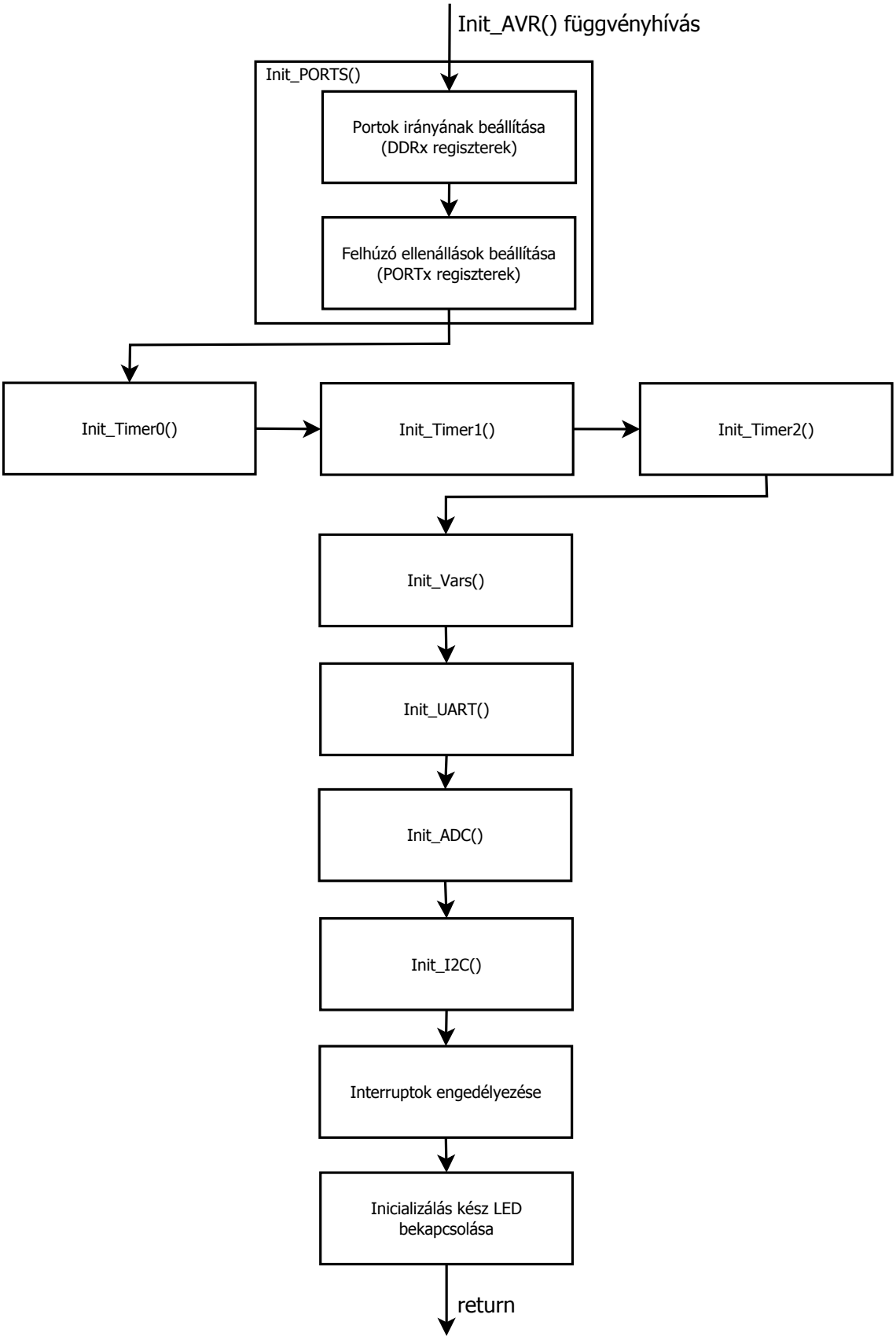
A portok inicializálásának összefoglalását az 1. táblázat adja, mely megmutatja, hogy melyik lába milyen irányú, mire való, és hogy szükséges-e az automatikus felhúzás tápra.

Port	Láb	Irány	Pull-up	Funkció
A	0	input	+	Dip switch
	1	input	+	
	2	input	+	
	3	input	+	
	4	output		Trigger out
	5	input		-
	6	input		Auxilary I/O 2
	7	input		
B	0	output		SPI
	1	output		
	2	output		
	3	input		
	4	input	+	Display buttons
	5	input	+	
	6	input	+	
	7	input	+	

C	0	output		Display segments
	1	output		
	2	output		
	3	output		
	4	output		
	5	output		
	6	output		
	7	output		
D	0	output		I2C (DAC)
	1	output		
	2	input		DMX IN
	3	output		-
	4	output		Displax LEDs
	5	output		
	6	output		
	7	output		
E	0	input		-
	1	output		-
	2	input		Auxilary I/O 1
	3	input		
	4	input		
	5	input		
	6	input		
	7	input		
F	0	input		Audio IN (ADC)
	1	output		Digit select
	2	output		
	3	output		
	4	input		JTAG
	5	input		
	6	output		
	7	input		

1. táblázat: A portok inicializálása

Ez a függvény minden elindításkor (vagy RESET után lefut). A függvény folyamatábrája a 4-2. ábrán látható. [17]



4-2. ábra: Az inicializáló függvény folyamatábrája

4.3.2 DMX jel beolvasása

A DMX jelet a mikrokontroller RX0 lábára kötöttem, és egy aszinkron soros port szoftvert írtam hozzá. A kommunikáció 8 bites adatcsomagokból áll, kettő stop bittel és paritás ellenőrzés nélkül. A baud rate beállításához a következő képlet használandó:

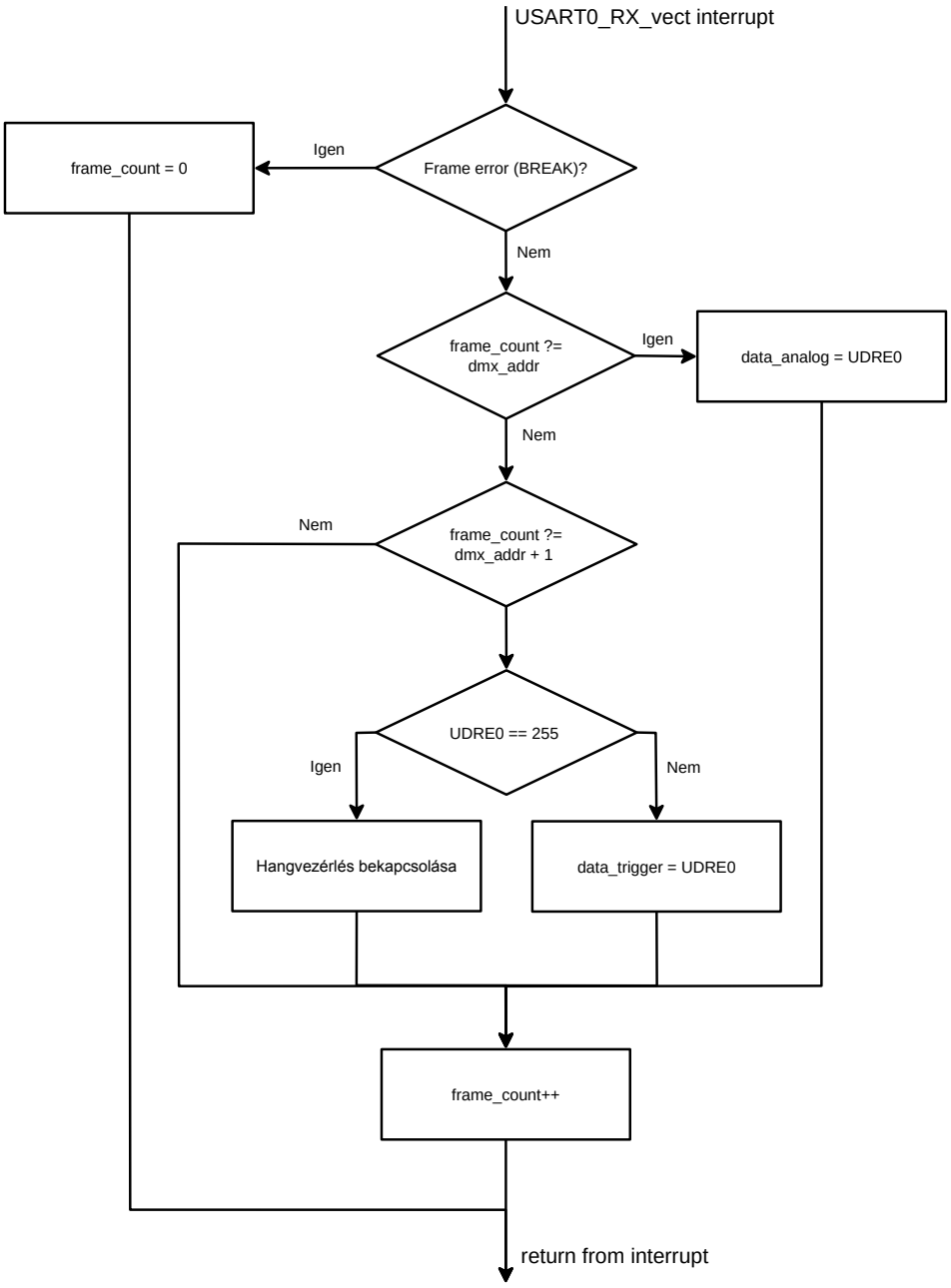
$$UBRR = \frac{F_{osc}}{16(baudrate)} - 1 = \frac{16M}{16(250k)} - 1 = 3.$$

Ezt az értéket kell az inicializálás során az UBRR (USART Baud Rate Register) regiszterbe tölteni. Mivel ez egy 16 bites szám lehet, valójában két regiszterbe kell beletölteni (UBRR0H, UBRR0L). Az inicializáló függvényben ezen kívül az UCSR (USART Control and Status Register) regiszterekbe (három van belőlük: UCSR0A, UCSR0B, UCSR0C) engedélyezni kell a fogadás befejeződését jelző interruptot, engedélyezni kell az adat fogadását, beállítani, hogy nincs paritás bit, 8 bites a kommunikáció, és 2 darab STOP bit van.

Amikor egy bájt beolvasása megtörtént, egy USART0_RX_vect interruptot kapunk. Első körben megvizsgáljuk, hogy nincs-e frame error. Amennyiben találunk, akkor ez valószínűleg azért volt, mert egy „break" jel jött, ami minimum 88 µs-ig „0"-ban tartotta az adatvonalat (egy bit átvitele 4 µs a 250 000 kbit/s sebesség miatt). Ha break jelet kaptunk, kinullázzuk a számlálót, ami megmutatja, hogy hányadik csatorna adata következik (frame_count).

Ha nem volt break, akkor beolvassuk a bájtot, és feldolgozzuk a kapott adatot. Alapvetően két adat érdekes csak, ezeket kell elkapni, a többit figyelmen kívül hagyjuk. Ami vezérlés szempontjából fontos adat, az nyomógombokkal beállított, az eszköz kijelzőjén is látható DMX címre és az az utánra érkező adat. Ezt az adatot a dmx_addr változóban tárolom. Amikor a számláló állása megegyezik a DMX címmel, akkor az analóg kimenet vezérlő adatát kaptuk meg, ha eggyel nagyobb, akkor pedig a trigger kimenetét. A trigger kimenet esetén beállíthatjuk a hangvezérlést. Ezt úgy tehetjük meg, ha 255-ös adatot kap a trigger kimenethez tartozó DMX csatornán. Egyéb esetekben beleírja a data_trigger változóba a kapott adatot, a rendszer többi része pedig eldönti majd, hogy mit kell csinálni vele. A sima analóg kimenetnél egyszerűen csak átadjuk a data_analog változónak a kapott értéket.

Minden esetben, amikor nem kaptunk frame errort, növelni kell eggyel a számláló értékét. A DMX jel beolvasásának folyamatát a 4-3. ábrán láthatjuk. [17,19,26]



4-3. ábra: A DMX jel beolvasásának folyamatábrája

4.3.3 Analóg kimenetek vezérlése

Az I^2C buszon kiküldi a program az adatokat a DAC átalakítónak. Ehhez négy egyszerű függvény megírására volt szükség: I2C_Init(), I2C_Start(), I2C_Write(unsigned char data), I2C_Stop. Valójában az I2C_Stop sem szükséges, ugyanis nem akarjuk leállítani a kommunikációt.

A program az Init_AVR() függvényben először meghívja az inicializáló függvényt. Itt beállítódik a prescaler és a bitrate. A TWSR (Two-Wire Status Register) regiszterben kell beállítani, hogy ne legyen prescaler és a TWBR (Two-Wire Bitrate

Register) regiszterbe pedig 14-et kell tölteni. Ez után a következő képlettel adódik az SCL vezeték órajelének frekvenciája:

$$SCL = \frac{Clk}{16 + 2(TWBR) \cdot 4^{prescale}} = \frac{16MHz}{16 + 2(14) \cdot 4^0} = 363{,}63kHz$$

Ez után a TWCR (Two-Wire Control Register) regiszterben engedélyezni kell a kommunikációt.

Az I2C_Start() függvény lefoglalja a buszt úgy, hogy a TWCR regiszterben bebillenti a start bitet (TWSTA). Ezután lekérdezéssel megvárja, míg a TWINT bit „1”, lesz. Ez azt jelenti, hogy a kommunikáció befejeződött.

Az I2C_Write(unsigned char data) kiírja a buszra a data változóban lévő adatot. A folyamat pont ugyanúgy működik, mint a start jel kiadásánál, csak itt a start bit bebillentése helyett a TWDR (Two-Wire Data Register) regiszterbe tölti a data változóban lévő adatot. A kommunikáció kezdetén meg kell hívni ezt a függvény úgy, hogy a data változóban az eszköz címe van, ami jelen esetben 0. Ezután a főciklusban folyamatosan tudjuk küldeni a DAC átalakítónak az aktuális adatot a függvény segítségével.

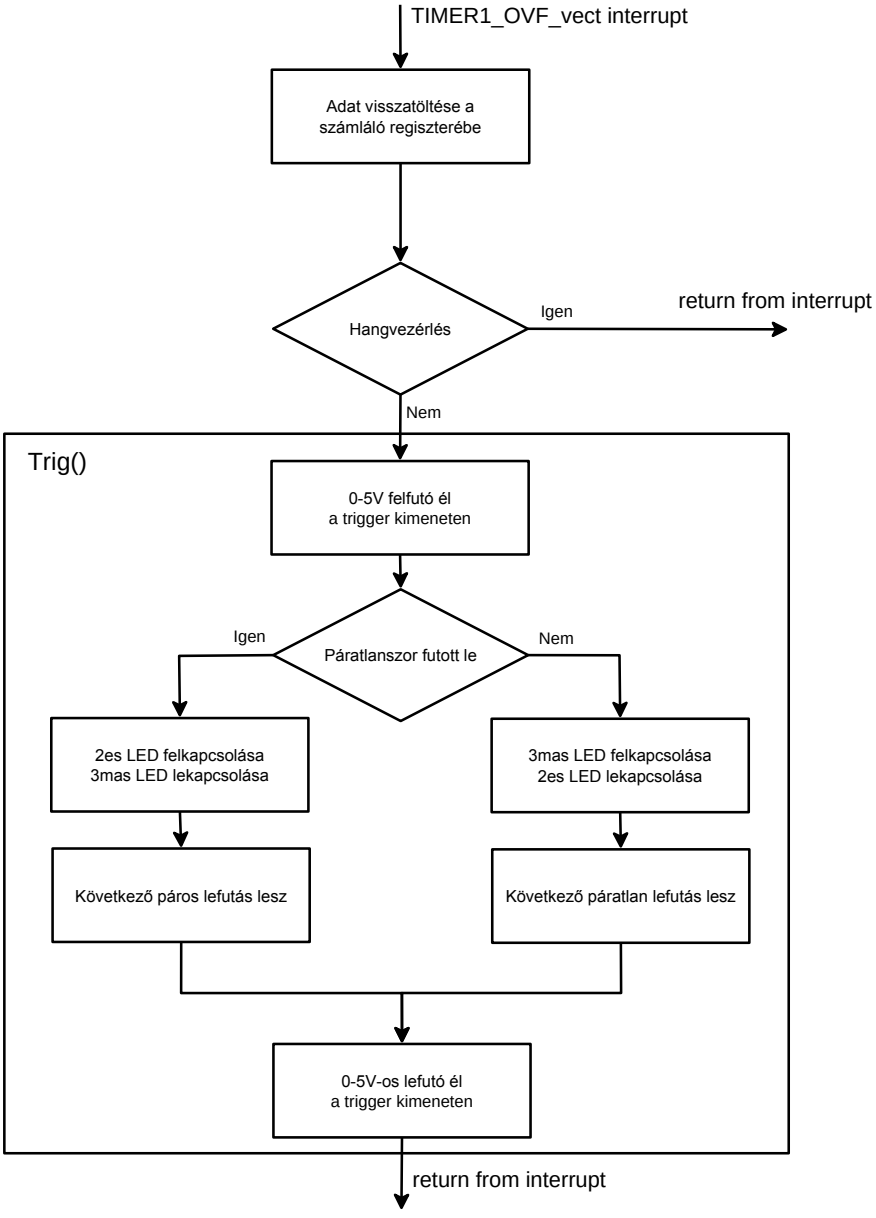
A trigger kimenet a Timer1 időzítővel van megoldva. Azért szükséges a Timer1 használata, mert az 16 bites számlálóval rendelkezik. Míg egy 8 bites számlálóval maximum $\frac{2^8}{\frac{16MHz}{1024}} = 0{,}16s$ időzítést lehet elérni, addig egy 16 bitessel $\frac{2^{16}}{\frac{16MHz}{1024}} = 4{,}19s$ időzítést lehet alkalmazni. Egy stroboszkóppal előfordulhat, hogy másodpercenként villog csak egyet. Így nyílván a 8 bites időzítővel nem megoldható a feladat.

Az időzítő számlálójának csak a felső 8 bitjét állítjuk, az alsó nyolc bitet mindig 0 értekre állítjuk. Ez azért kézenfekvő, mert a DMX jelből is csak 8 bites adatot tudunk fogadni. Így az időzítés $\frac{2^8}{\frac{16MHz}{1024}} = 0{,}16s$ és $\frac{2^{16}-2^8}{\frac{16MHz}{1024}} = 4{,}18s$ között állítható.

Ez alapján a Timer1 inicializálásánál a TCNTH1 és TCNTL1 (Timer/Counter High és Low) regiszterekbe rendre a kapott frekvencia adatot és nullát kell tölteni. A TCCRA és TCCRB (Timer/Counter Contorl Register A és B) regiszterekben pedig be kell állítani a normál módot az 1024-es prescalert.

Az interrupt rutinban lényegében annyi történik, hogy a mikrokontroller megfelelő lábát egyesbe billenti, majd vissza nullára. Ezen kívül a kijelzőn lévő második és harmadik LED állapotát is megváltoztatja (az inicializáláskor egyik világított, a másik nem). Így lényegében egy metronómszerű villogót kapunk, ami egy visszajelzés a felhasználó számára, hogy jól működik az eszköz. Ez a folyamat

természetesen csak akkor történik, ha nincs hangvezérlés, tehát a control változó MSB-je „0” értékű. Egyébként az interrupt rutin nem csinál semmit. Ezek a műveletek egy külön Trig() függvénybe ki vannak emelve, hogy a hangvezérlésnél is felhasználhassuk azokat. A 4-4. ábrán láthatjuk ezt egy folyamatábrán, a könnyebb érthetőség kedvéért. [17,19,20,24,25]



4-4. ábra: A trigger kimenet vezérlésének folyamatábrája

4.3.4 Hangvezérlés

Az audio bemenet az AVR beépített AD átalakítójához van csatlakoztatva. A mintavételezést a Timer2 időzítő segítségével végezzük. A prescalert 1024-re állítva 00H értéket töltünk a számlálójába, így $\frac{16M}{1024} = 15{,}625k$ mintavételi frekvenciát

használunk. A mintavételi törvény alapján elméletileg olyan jeleket tudunk detektálni, melyek frekvenciája a mintavételi frekvencia felénél kisebbek. Bár ezzel nem lehet minden hallható hangot felvenni, de ez nem is baj, mert magas hangokra úgyse akarunk triggerelni. Maximum kísérőcinekre, de az alapfrekvenciája annak is jóval 7,8 kHz alatt van. A hardverben lévő aluláteresztő szűrő miatt amúgy se fognak a magas hangok bejönni.

A vezérlés csak akkor történik itt, ha hangvezérlés van beállítva. Ezt egy control nevű változóban tároljuk. Ha a változó MSB-je „1” értékű, akkor hangvezérlés lesz. Alapvetően a hangjel csúcsait keressük egy nagyon egyszerű módon. Egy makróban definiálva van egy komparálási (treshold, küszöb) szint. Amikor a jel mintája meghaladja ezt a jelszintet, de az előző minta nem haladta meg, akkor meghívjuk a Trig() függvényt. Ezt a paramétert 512 és 1023 között érdemes tartani, mivel 512 a nulla szint a munkapont beállító ellenállások miatt (amik tápfeszültség felével egyenlő egyenfeszültségű komponenst adnak a jelhez) és 1023 a teljes kivezérlés (10 bites ADC van a mikrokontrollerben.

A hangvezérlés ötletének szimulálására készítettem egy MATLAB szkriptet, amely egy zeneszám részletének hangjára vizuálisan kirajzolja, hogy mekkora treshold értékeknél, hol fog triggerelni az eszköz. A Prodigy - Invaders Must Die számát választottam, mivel jól lüktető kísérete alkalmas arra, hogy megvizsgáljuk a problémát:

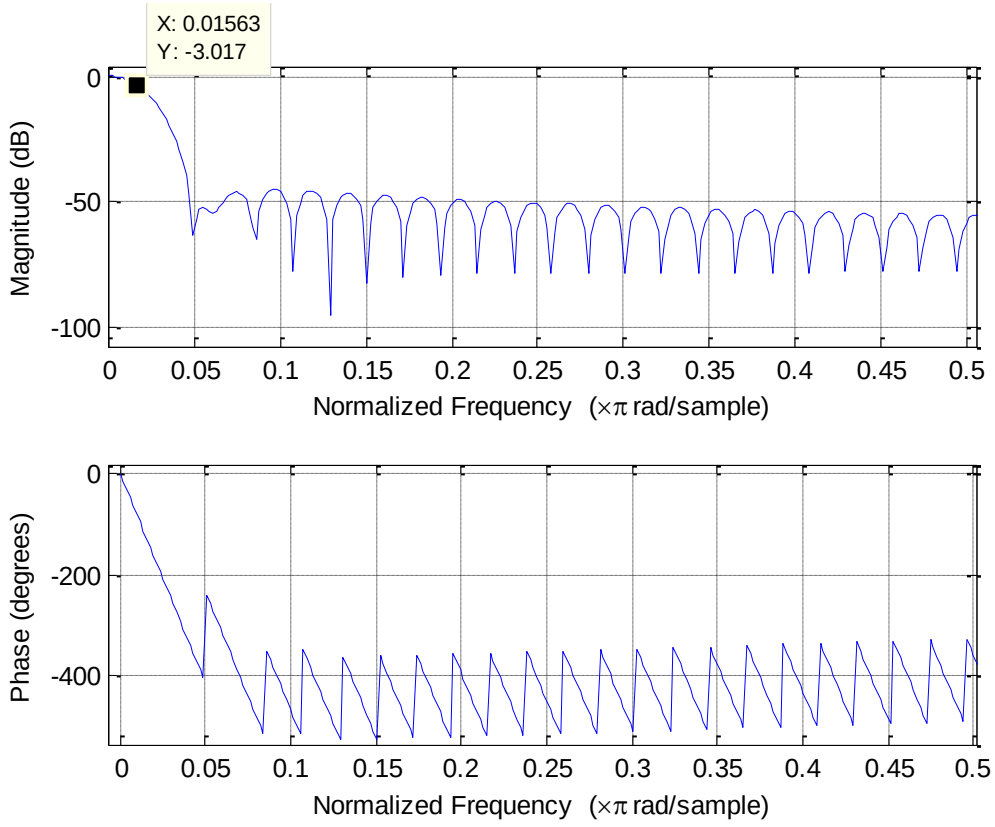
A MATLAB szkript:

```
%zene beolvasása
[music,fs]=wavread('D:\Invaders.wav');
y = music/max(music);

t = [1:length(y)];
treshold = 848 / 1024 * 5 * ones(1,length(y));

b = fir1(92,[60 400]/fs);
figure;
freqz(b,1,512);

filtered = filter(b,1,y);

%időtartományban
figure;
plot(t,2*filtered + 2.5,'b',t,treshold,'r');
title('Időtartomány');
```

```
xlabel('Idő');
ylabel('Feszültség [V]');
```

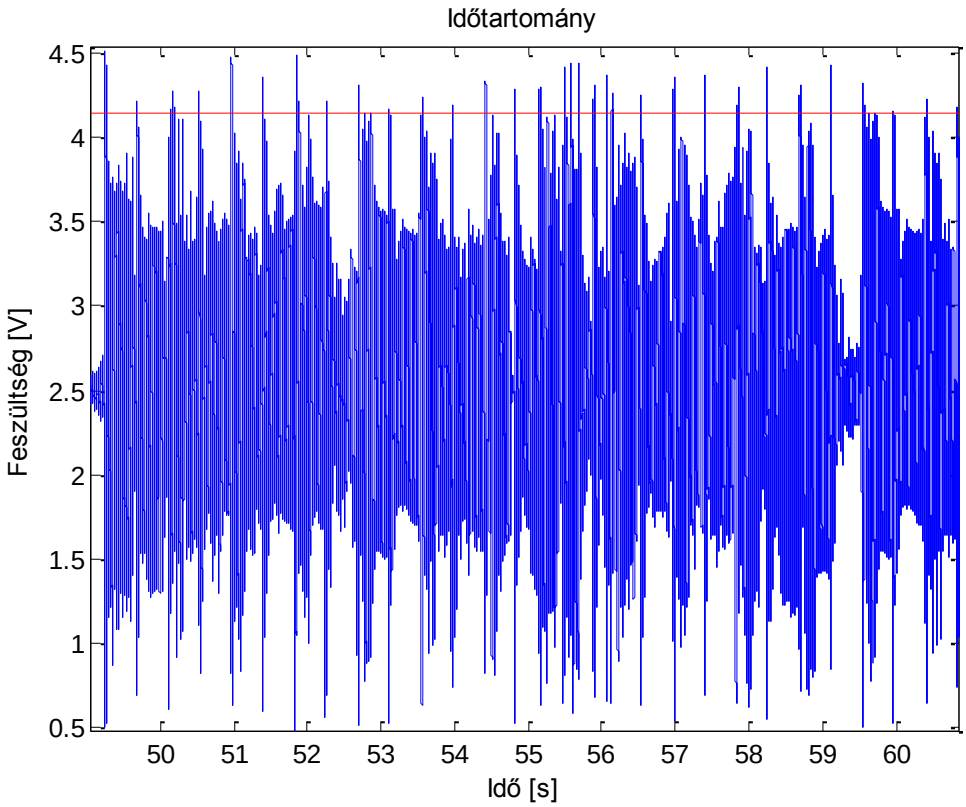
A szkript elején beolvassuk a wave fájlt, utána normáljuk 1-re (digitális Full Scale). Létrehozunk egy treshold szintet, amit ábrázolni is tudunk. A tresholdot 848-ra állítottam be. Ez fizikailag $treshold = \frac{848+1}{1024}5V = 4{,}146V$ feszültséget jelent. Ezután pedig egy FIR szűrő segítségével megszűrjük a zenét. Egy olyan sáváteresztő szűrőt választottam, melynek töréspontí frekvenciái 60 és 400 Hz-nél vannak. Ez jól modellezi a bemeneten lévő sávszűrést. Mivel 44.1 kHz a mintavételi frekvencia, ezért a szűrőkészítő függvény nem nagyon veszi figyelembe a felülátereszto részt, legalábbis ilyen fokszám esetén, de ez nem okoz semmi problémát.

A megszűrt jelet a treshold szinttel együtt ezután a 4-6. ábrán láthatjuk, a szűrő Bode diagramját pedig a 4-5. ábrán.



4-5. ábra: A tervezett FIR szűrő Bode diagramja

Az ábrából láthatjuk, hogy körülbelül megfelel a specifikációnak a szűrő. A data cursor segítségével láthatjuk, hogy nagyjából a $0.01563 \cdot 44100Hz = 689Hz$ frekvenciánál van a töréspont (-3 dB-es pont). Mivel nem használtam túl nagy fokszámú szűrőt, ezért ez a pontosság a vártnak megfelelő.



4-6. ábra: A zenefájl az időtartományban, a treshold szinttel együtt

Az idő azért kezdődik 49 s-tól, mert ott kezdett el bejönni a zeneszámba a lábdob. A magas csúcsok pillanatában van a lábdob csattanása, ezek a pontok haladnak át a treshold szinten. [17,18,19,24,25]

4.3.5 Kijelző

A kijelző időmultiplex módon működik. Ehhez a Timer0 időzítőt használtam. Az iniciálás során 1024-es prescale-t használtam, tehát a 16 MHz-es órajel helyett 15,625 kHz-es órajelet kapott az időzítő. A számláló (időzítő) kezdeti értékét egy makróban definiáltam annak érdekében, hogy könnyen változtatható legyen 16 ms és 64 µs értékek között a timer0 interruptok között eltelt idő. Ez azért fontos, mert minél ritkábban következik be, annál halványabb lesz a kijelző fényereje (esetleg villogást is látunk), viszont egyre több minden másra lesz ideje a mikrokontrollernek (kevesebb erőforrást igényel a kijelző vezérlés). Így egy paraméter változtatásával tudjuk szabályozni a kijelző erőforrásigényét.

A Timer0 interrupt rutinja újratölti a timer0 számlálójának értékét, először beállítja, hogy melyik digit legyen az aktív, majd, hogy melyik szegmenseket kell az adott digiten kigyújtani. Ezt úgy teszi, hogy egy tömbben benne vannak az értékek,

amiket ki kell küldeni a C porton, és ezt a tömböt indexeli. Fontos, hogy digitváltás közben ne küldjön semelyik szegmensre se „1” értéket, mert különben mindegyik digiten a szegmensek „vagy” kapcsolata jelenik meg. Pl. ha a kijelzőn a 166 számot akarnánk megjeleníteni, akkor hibásan a 888 számot látnánk.

A kijelzőn kijelzendő számot egy globális háromelemű karaktertömbben tárolom. Amikor valami mást kell kijelezni, ezt a tömböt változtatja meg. Jelenleg csak számokra van felkészítve az eszköz, de könnyen belátható, hogy betűket is kiírathatnánk kis módosítással vele (Pl. Egy „ERR” kijelzést, ha valami hiba történt).

A kijelző vezérlésének folyamatábrája a következő pontnál, a nyomógombokkal együtt kerül ismertetésre. [17,19,22,24,25]

4.3.6 Nyomógombok, kapcsolók

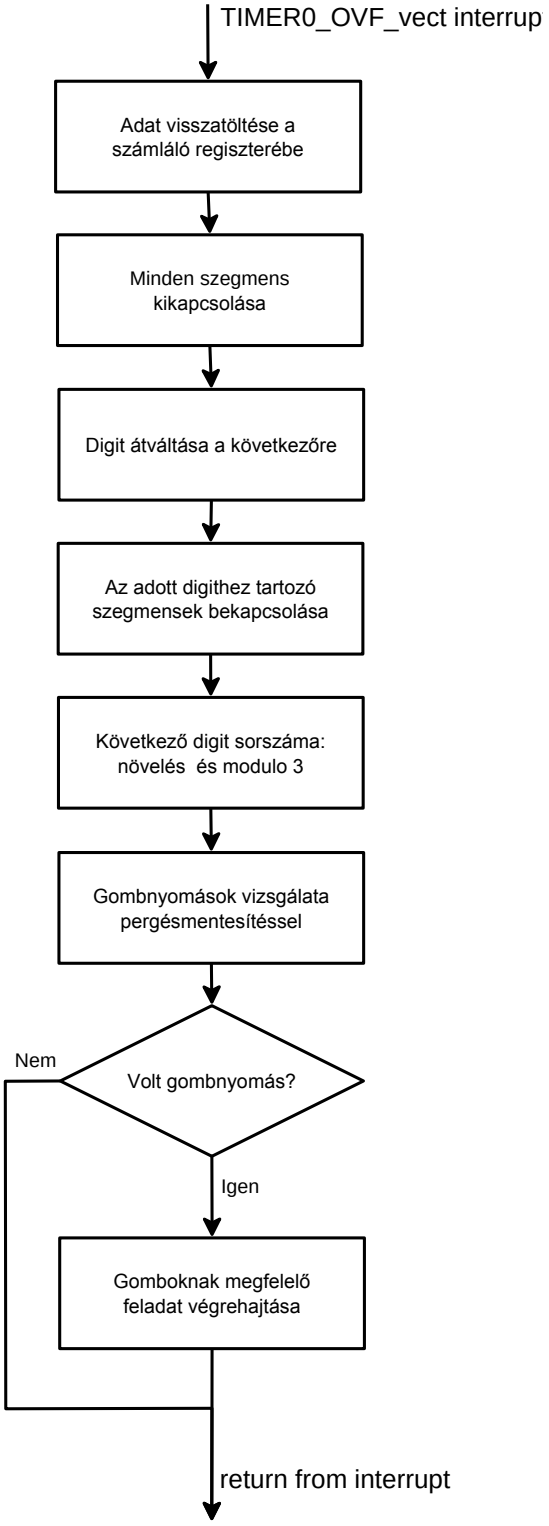
A nyomógombok pergésmentesítését úgy oldottam meg, hogy a Timer0 segítségével mintavételezem a nyomógombok állását (ez vezérli a kijelzőt is). Minden nyomógombhoz létezik egy változó, amivel egy állapotgépet hozok létre. Ha a nyomógomb be van nyomva, akkor növeli a változó értékét, ha nincs, akkor kinullázza. Így ha egymás után többször is látjuk megnyomva a gombot, akkor már biztos, hogy egy tényleges gombnyomásról beszélünk.

A négy nyomógomb alapvetően ebben a szoftververzióban arra jó, hogy léptessük a DMX címet. A felső gomb eggyel léptet előre, az alsó pedig hátra. A jobb oldali pedig tízzel léptet előre, a bal oldali pedig hátra. A gombnyomásokat egy külön függvény kezeli, ami figyel arra is, hogy a DMX cím mindig 1 és 512 között legyen.

A gombnyomások során nem csak a DMX címet tartalmazó változót kell módosítani, hanem a kijelzőn lévő számot is.

A NYÁK-on található DIP kapcsolók nincsenek leprogramozva. Debug célokra illetve bonyolultabb szoftver esetén extra funkciók előhozására, mint pl. eszközök tesztelése lehet alkalmas.

A nyomógombokat és a kijelzőt vezérlő Timer0 interrupt rutin a 4-7. ábrán szemléltethető. [17,19]



4-7. ábra: A TIMER0_OVF_vect interrupt rutin

4.3.7 Működési állapotjelző LED-ek

A négy LED közül egy az eszköz inicializálásának befejeződését jelzi. Ha világít, az eszköz elvileg működésre kész. A másik kettő metronóm jellegű villogót

valósít meg a trigger kimenethez. A negyedik nincs leprogramozva, bonyolultabb szoftver tudná használni. [17,19]

5 Eredmények értékelése, továbbfejlesztési lehetőségek

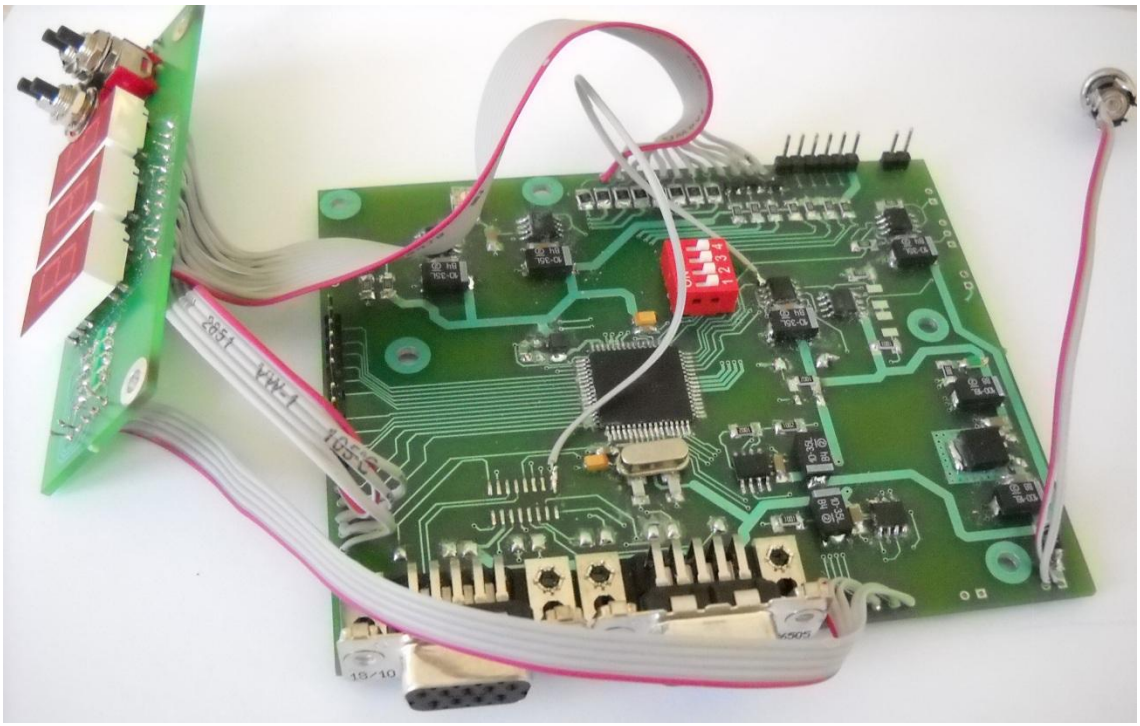
5.1 Hardvergyártás

A hardver legyártása során rengeteg tanulságot sikerült levonni, ami alapján sok mindent máshogy terveznék a nyomtatott áramköri lemezen.

Egyik probléma az volt, hogy nem minden alkatrészből sikerült olyan tokozású alkatrészt beszerezni, amilyen footprint volt a NYÁK-on. Ezért azt a megoldást választottam, hogy az így hiányzó D-tokozású 1 µF-os tantál kondenzátorok helyére furatszerelt kondenzátorokat forrasztottam.

A debug célokra tervezett soros portot kezelő IC-t nem forrasztottam be, helyette inkább átkötöttem az RS-485 illesztő IC-t a soros portra. Így a soros portot használva kevesebb processzoridőbe telik a DMX jel feldolgozása, ugyanis hardveresen lehet figyelni a kommunikációt, és csak a beérkezett bájtoknál keletkezik interrupt, nem bitenként, mint ha ezt a változtatást nem végeztem volna el.

Ezek mind olyan megoldások, amik egy prototípusnál még teljesen elfogadottak. Ha több példányban készülő, forgalomba hozott eszközről lenne belőle, akkor ezeken a pontokon újra kellene tervezni a hardvert.



5-1. ábra: Az elkészült hardver műszerdoboz nélkül

Az 5-1. ábrán az elkészült hardver műszerdobozba szerelése előtti állapota látható. A két áramköri lap közötti vezetékezés nem túl szépen van megoldva. Nyilván, ha gyártásba kerülne az eszköz, akkor szalagkábel csatlakozókkal lenne megoldva, és bármikor lehúzható lenne a panel, nem pedig fix beforrasztással.

A hardvergyártás során sok energia ment el az alkatrészek kiválasztásával, beszerzésével, átvételével, a számlák illetve pénzügyek intézésével, a beforrasztással, hibakereséssel és javítással.

5.2 A hardver élesztése

Az első tápra dugásnál egy táp-föld zárlat miatt nem indult el a rendszer. Elég sokáig tartott kideríteni, hogy mi volt a probléma forrása, ugyanis ekkor már minden alkatrész be volt forrasztva. Arra jutottam, hogy legközelebb érdemes néhány alkatrészenként ráadni az eszközre a tápot, hogy kiderüljön, ha valami nem jó. Ez jelentősen leegyszerűsítette volna a hiba megtalálását. A problémát amúgy egy zárlatos tantál kondenzátor okozta.

Ezután jöhetett a mikrokontroller programjának feltöltése. A kezdeti sikertelenségek oka az volt, hogy a cég, amelyik legyártotta a nyomtatott áramköri lemezt, egy technológiai hibát vétett, és egy zárlat keletkezett, ami miatt az eszköz folyamatosan RESET állásban tartotta a mikrokontrollert. Valószínűleg a gyártás során egy hajszál esett a lemezre levilágításkor, így ott nem érte megfelelő UV fény, és ezért maradt rajta felesleges vezető. Ezt a zárlatot mikroszkóppal sikerült megtalálni, majd az átvezetést késsel óvatosan kivágni, így már programozhatóvá vált a hardver.

5.3 Használati útmutató

Az elkészült hardverhez egy 12V-os DC tápegységet kell csatlakoztatni. A DMX láncba be kell fűzni az eszközt a DMX in és DMX out XLR csatlakozók segítségével. Amennyiben ez a lánc utolsó eleme, a DMX out csatlakozóba egy 120 Ω-os lezárást kell csatlakoztatni. A lezárás nem más, mint egy XLR csatlakozóba forrasztott 120 Ω-os ellenállás.

A vezérelendő 0-10V-os analóg jellel működő eszközöket a megfelelő jack csatlakozós kimenetre kell dugni. Kettőnél több eszközt is vezérelhetünk a berendezéssel, de akkor meg kell oldani a vezérlőjelek Y-ozását. Természetesen így az adott eszközök csak együtt lesznek vezérelhetőek. Viszont így sem célszerű nagyon sok eszközt csatlakoztatni, mert romolhat a vezérlés megbízhatósága.

A hang bemenő jelét a megfelelő jack csatlakozóba kell csatlakoztatni. Érdemes erősen kompresszált jelet küldeni az eszközre, hogy ne kelljen sokszor állítani a bemenő hangerőt. Így a kisebb dinamikatartomány miatt könnyebb triggerelni a jelre.

Miután mindent csatlakoztattunk, a nyomógombok segítségével be kell állítani a DMX címet. A felső és alsó gombokkal egyet tudunk felfelé és lefelé lépni, a bal és jobb gombbal pedig 10et lefelé és 10et felfelé. Az aktuális beállított címet a kijelzőn láthatjuk. Mikor beállítottuk a címet, figyeljünk arra, hogy a DMX jel forrását (pl. DMX-es fényvezérlő pultot) is felprogramoztuk erre a címre és a két csatornaszámra.

Ha mindezt megtettük, a berendezés működésre kész. A kijelzőn látható csatorna száma a jelszinttel arányos kimenet címét, az annál eggyel nagyobb pedig a frekvenciával arányos kimeneti jelhez tartozó címet látjuk. Amennyiben 255-öt küldünk az utóbbi csatornára, aktiváljuk a hangvezérlést.

5.4 Továbbfejlesztési lehetőségek

Lehetséges szoftveres továbbfejlesztési lehetőség egy bonyolult menürendszer megvalósítása, melyben kiválasztható például egy teszt üzemmód, ahol az analóg kimeneteken az összes lehetséges értéket kiküldi sorban, megadott gyorsasággal. A triggerre bekapcsolható kétszerezés vagy négyszerezés is hasznos lehet, ha pl. a kíséret alapján villogtatunk, és a lábdob pl. negyedeket üt, akkor annak a ritmusára fog beállni (mert a szűrt tartományban valószínűleg az lesz a domináns), viszont mi a pergőre szeretnénk ráállítani, ami tegyük fel, kétszer üt egy lábdob alatt.

Mivel a mikrokontroller nem használt lábai is ki vannak vezetve, így bővíthető az eszköz további áramkörökkel. Például egy numerikus billentyűzetet is lehet akár csatlakoztatni hozzá.

Bár már a téma határait súrolja, de egy személyi számítógépen futó tesztalkalmazás is hasznos lehet, mert nem mindig (és nem mindenkinek) adódik lehetőség egy komplett rendszerben kipróbálni az eszközt. Ez esetben viszont egy USB/DMX átalakító hardver beszerzésére (esetleg tervezésére és legyártására) is szükség van.

Irodalomjegyzék

[1] Böröcz Sándor: Színpadvilágítás, elmélet és technika
2006, Budapest, Colorom Media

[2] DMX – Wikipedia
http://hu.wikipedia.org/wiki/DMX
(2011. december 3.)

[3] DMX512 – Wikipedia, the free encyclopedia
http://en.wikipedia.org/wiki/DMX512
(2011. december 3.)

[4] EIA-485 – Wikipeadia, the free encyclopedia
http://en.wikipedia.org/wiki/EIA-485
(2011. december 3.)

[5] Stage lighting – Wikipedia, the free encyclopedia
http://en.wikipedia.org/wiki/Stage_lighting
(2011. december 3.)

[6] Fog machine – Wikipedia, the free encyclopedia
http://en.wikipedia.org/wiki/Fog_machine
(2011. december 3.)

[7] Lighting console – Wikipedia, the free encyclopedia
http://en.wikipedia.org/wiki/Lighting_console
(2011. december 3.)

[8] Daisy chain (electrical engineering) – Wikipedia, the free encyclopedia
http://en.wikipedia.org/wiki/Daisy_chain_%28electrical_engineering%29
(2011. december 3.)

[9] Dimmer – Wikipedia, the free encyclopedia
http://en.wikipedia.org/wiki/Dimmer
(2011. december 3.)

[10] Twisted pair – Wikipedia, the free encyclopedia
http://en.wikipedia.org/wiki/Twisted_pair
(2011. december 3.)

[11] Stroboscopic effect - Wikipedia, the free encyclopedia
http://en.wikipedia.org/wiki/Stroboscopic_effect
(2012. május 7.)

[12] Altium Designer Tutorial: Schematic capture and PCB layout (1of2) (improved)- Youtube
http://www.youtube.com/watch?v=Y7PY1nBtImk
(2011. december 3.)

[13] Altium Designer Tutorial: Schematic capture and PCB layout (2of2) (improved) – Youtube
http://www.youtube.com/watch?v=9u0Fzpb0yZU
(2011. december 3.)

[14] Altium Designer Tutorial: Create a component library – Youtube
http://www.youtube.com/watch?v=pW_kpk5lQlM
(2011. december 3.)

[15] Altium Designer Tutorial: Copper planes and pours – Youtube
http://www.youtube.com/watch?v=pg6l02Rinos
(2011. december 3.)

[16] Altium Designer Tutorial - Modify / Edit polygon shape – Youtube
http://www.youtube.com/watch?v=gL1oSBMsLn8
(2011. december 3.)

[17] Muhammad Ali Mazidi, Sarmad Naimi, Sepehr Naimi:
The AVR microcontroller and embedded systems: using Assembly and C
2011, New Jersey, Paerson Education

[18] Atmel AVR126: ADC of megaAVR in Single Ended Mode (Application Note)
2011, San Jose, Atmel Corporation

[19] ATmega128A (complete datasheet)
2011, San Jose, Atmel Corporation

[20] AVR310: Using the USI module as a I^2C master (Application Note)
2004, San Jose, Atmel Corporation

[21] AVR 067: JTAGICE mkII Communication Protocol (Application Note)
2009, San Jose, Atmel Corporation

[22] AVR242: 8-bit Microcontroller Multiplexing LED drive and a 4 x 4 Keypad (Application Note)
2002, San Jose, Atmel Corporation

[23] AVR186: Best Practices for the PCB layout of Oscillators (Application Note)
2008, San Jose, Atmel Corporation

[24] AVR134: Real Time Clock (RTC) using the Asyncronous Timer (Apllication Note)
2009, San Jose, Atmel Corporation

[25] AVR130: Setup and Use the AVR Timers (Application Note)
2002, San Jose, Atmel Corporation

[26] AVR306: Using the AVR UART in C (Application Note)
2002, San Jose, Atmel Corporation

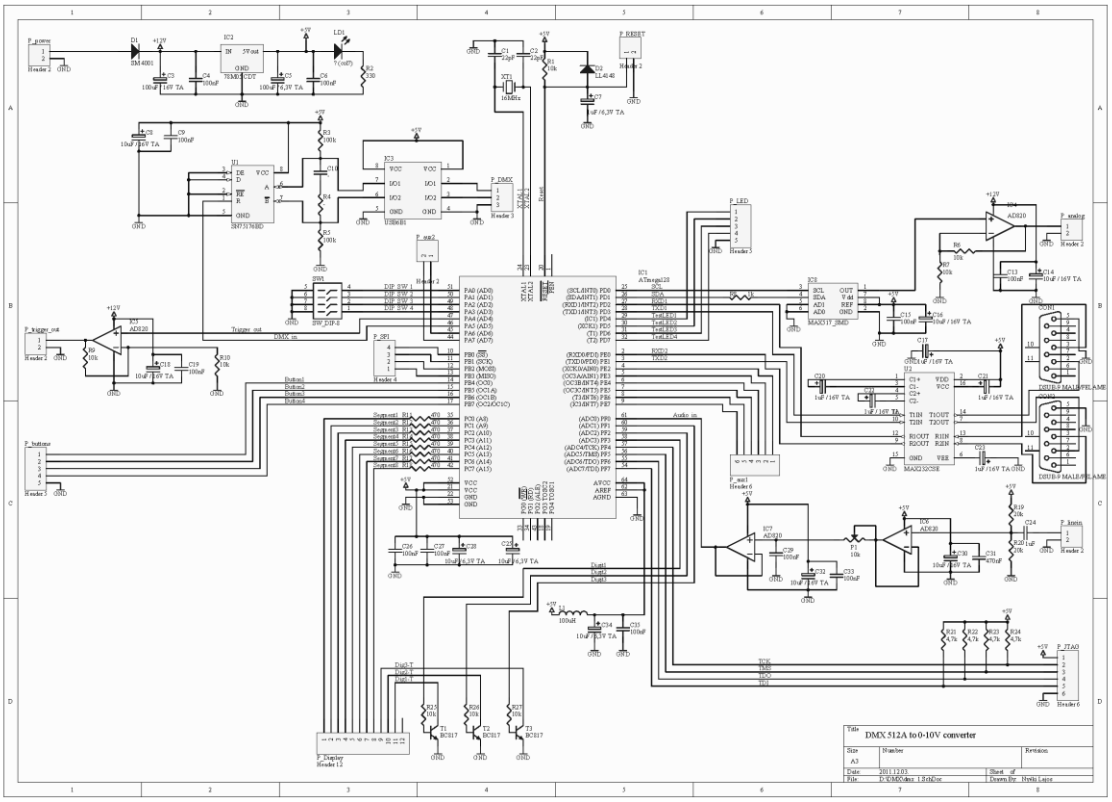
[27] Atmel AVR – Wikipedia, the free encyclopedia
http://en.wikipedia.org/wiki/Atmel_AVR
(2012. május 7.)

[28] Zaj Csoport
http://www.zaj.hu/
(2012. május 12.)

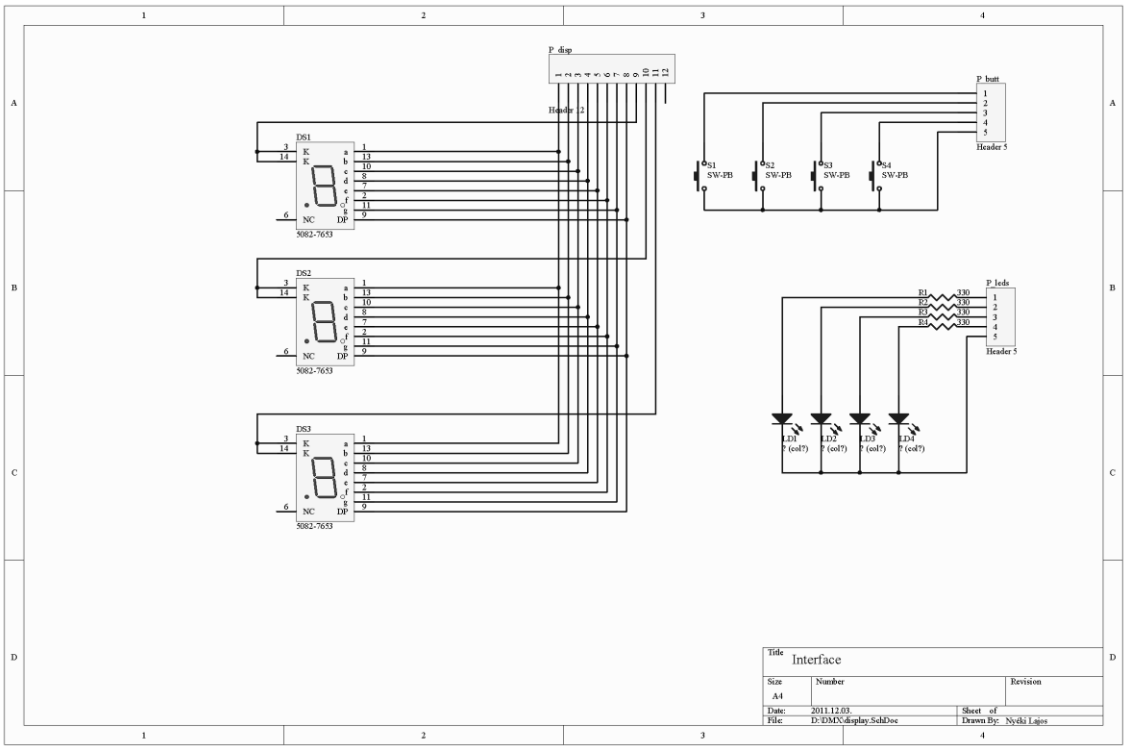
[29] Decoupling capacitor – Wikipedia, the free encyclopedia
http://en.wikipedia.org/wiki/Decoupling_capacitor
(2012. május 12.)

[30] Mészáros Balázs: Digitális zajszintmérő tervezése (Szakdolgozat)
2011, Budapest, BME-MIT
Konzulensek: Bogár István, Sujbert László

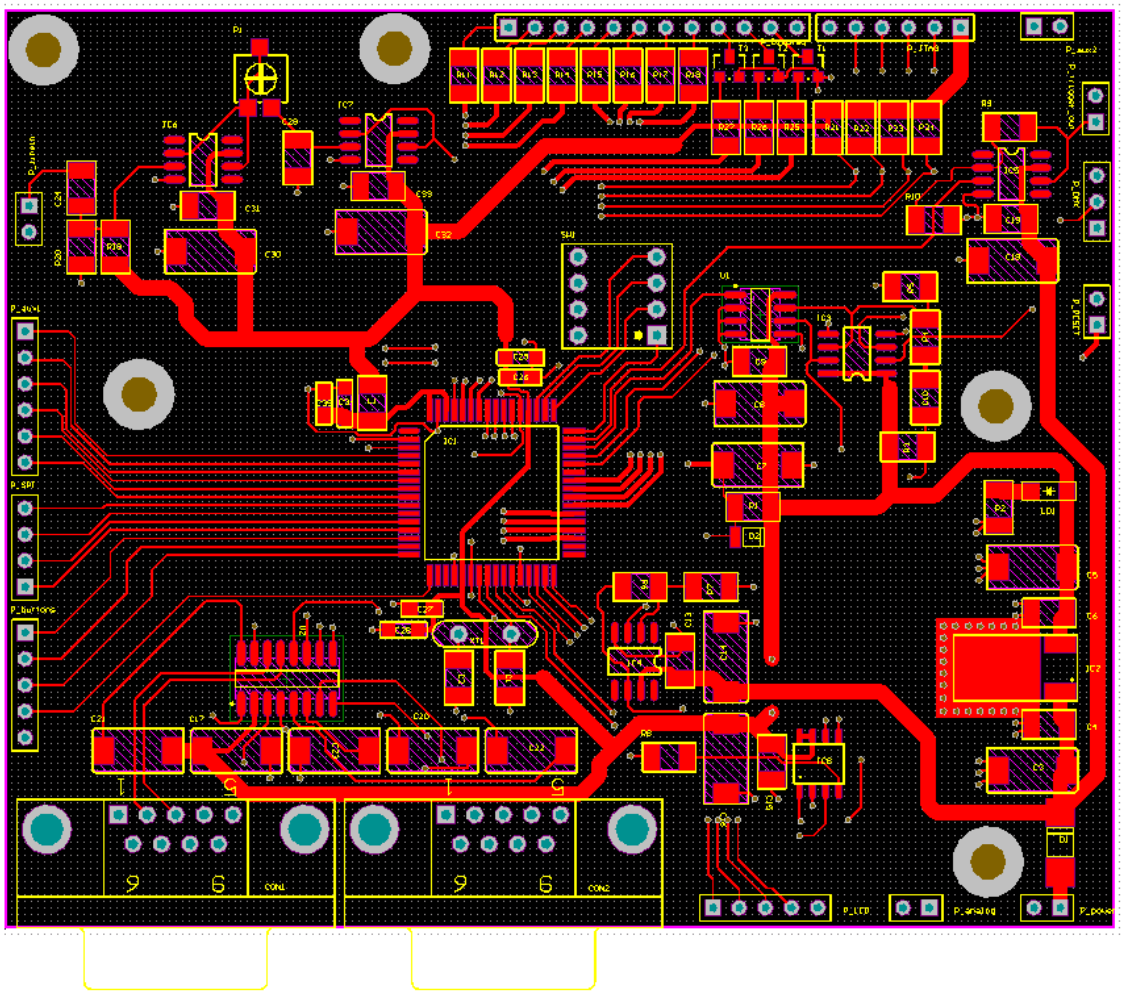
Függelék



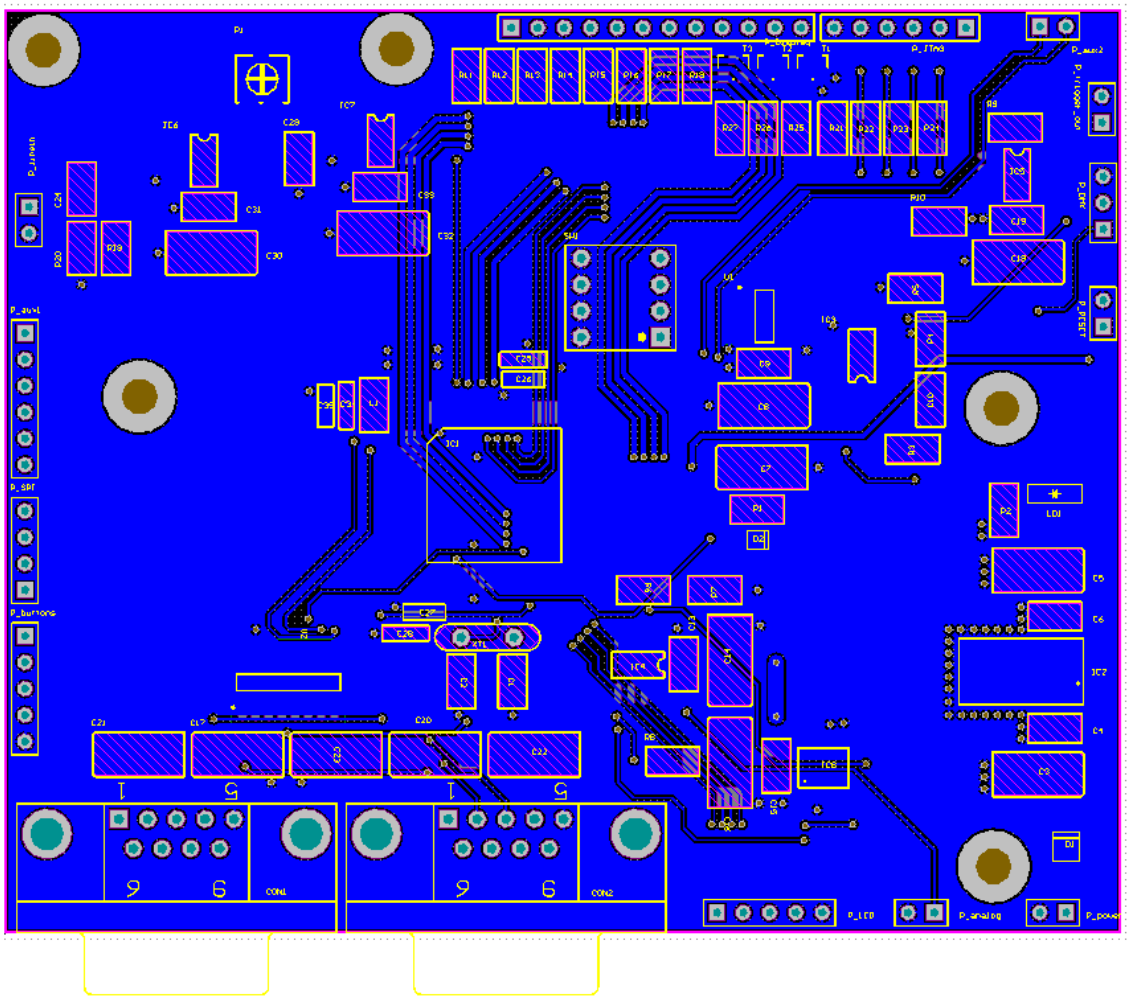
5-2. ábra: Alaplap kapcsolási rajza



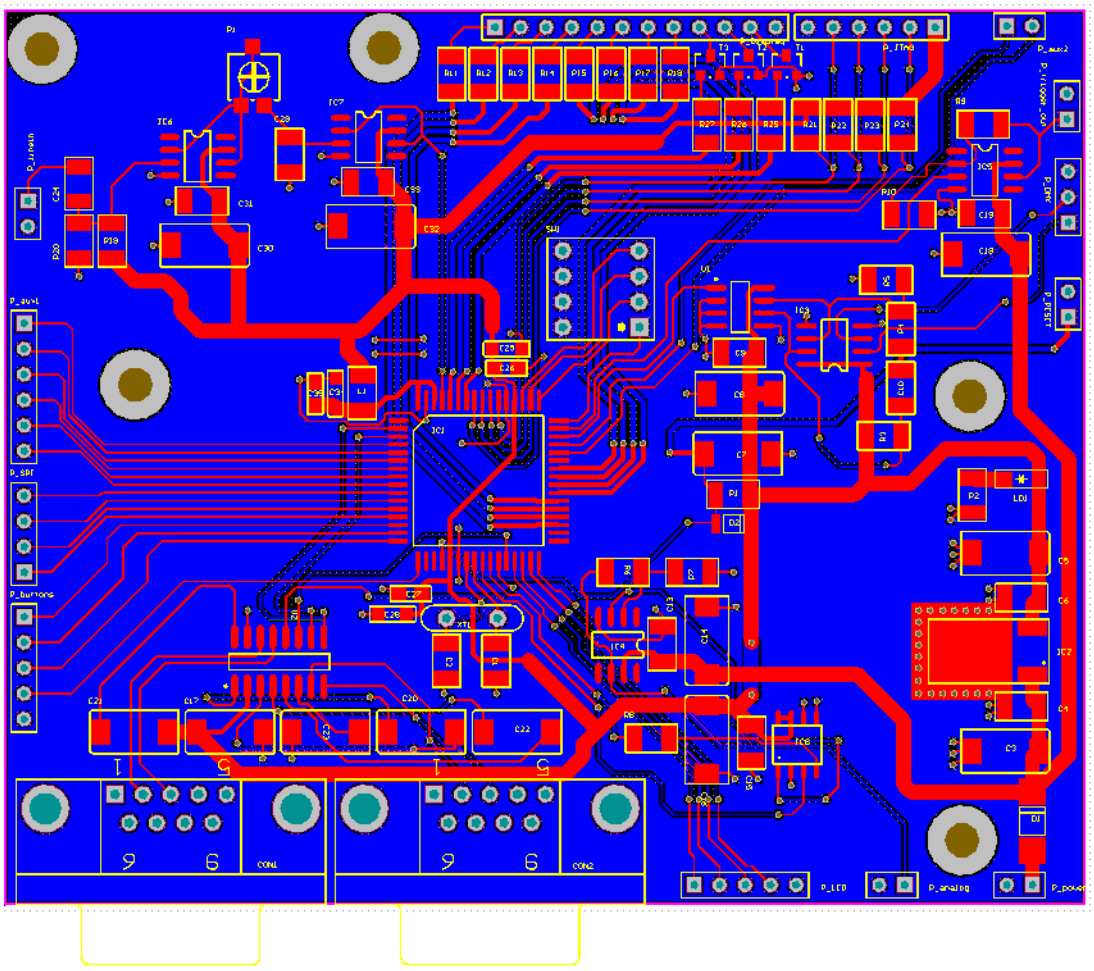
5-3. ábra: Kijelző kapcsolási rajza



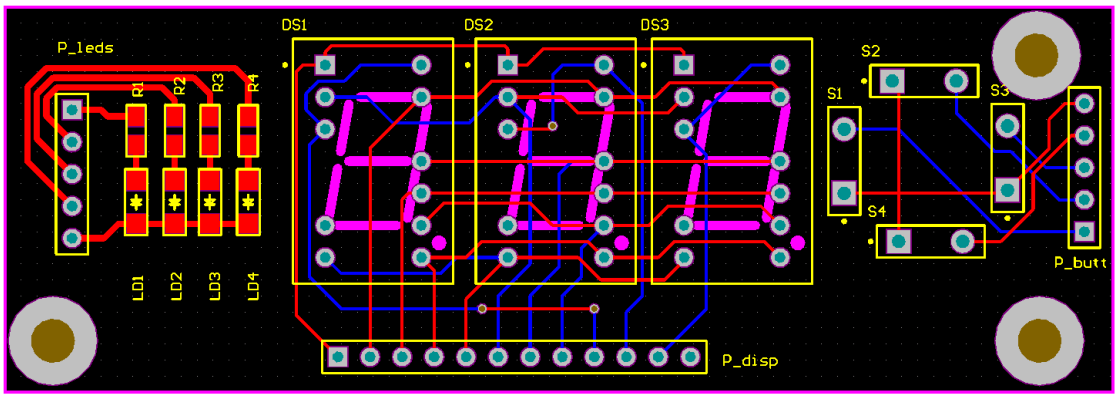
5-4. ábra: Alaplap felső rétege



5-5. ábra: Alaplap alsó rétege



5-6. ábra: Alaplap alsó és felső rétege



5-7. ábra: Előlap huzalozási terve

Ábrajegyzék