

Budapesti Műszaki és Gazdaságtudományi Egyetem

Villamosmérnöki és Informatikai Kar

Méréstechnikai és Információs Rendszerek Tanszék

Bibók Andor

Digitális szűrőtervezés jelfeldolgozó processzoron

Konzulens

Orosz György

BUDAPEST,

Tartalomjegyzék

[Összefoglaló 4](#_Toc357348307)

[Abstract 5](#_Toc357348308)

[1 Bevezető 6](#_Toc357348309)

[2 Digitális szűrők 7](#_Toc357348310)

[2.1 FIR szűrők 7](#_Toc357348311)

[2.1.1 Tervezési eljárások 8](#_Toc357348312)

[2.2 IIR szűrők 10](#_Toc357348313)

[2.2.1 Tervezési eljárások 11](#_Toc357348314)

[2.2.2 Megvalósítás 25](#_Toc357348315)

[3 Fejlesztői környezet 30](#_Toc357348316)

[3.1 ADSP-BF537 EZ-KIT Lite 30](#_Toc357348317)

[3.1.1 ADSP-BF537 Blackfin processzor 31](#_Toc357348319)

[3.1.2 ADC és DAC 31](#_Toc357348321)

[3.2 Visual DSP++ 5.0 31](#_Toc357348322)

[4 A feladat megvalósítása 33](#_Toc357348323)

[4.1 Specifikáció és rendszerterv 33](#_Toc357348324)

[4.1.1 A feladat ismertetése 33](#_Toc357348325)

[4.2 Szűrőtervezés 33](#_Toc357348326)

[4.3 Vezérlés 33](#_Toc357348327)

[4.4 Kommunikáció 33](#_Toc357348328)

[Függelék 35](#_Toc357348329)

Hallgatói nyilatkozat

Alulírott **Bibók Andor**, 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 hitelesített 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,

...…………………………………………….

Bibók Andor

Összefoglaló

A digitális jelfeldolgozás legalapvetőbb elemei közé tartoznak a digitális szűrők. Felhasználási körük igen széles, a szórakoztató elektronikától kezdve, a professzionális méréstechnikáig gyakorlatilag mindenhol megjelennek. Az elektronikus szűrők megjelenése óta rengeteg szűrési módszert fejlesztettek ki. Ennek legfőbb oka, hogy különböző alkalmazások esetén más és más módszer vezetett az ideális eredményhez.

Digitális szűrők esetén jól bevált módszer, hogy a szűrőt egy speciális program segítségével megtervezzük, majd egy célhardverre feltöltve használjuk. A digitális szűrők egyik fő előnye abban rejlik, hogy nagymértékben átkonfigurálhatóak, így egy hardveren könnyen megvalósítható több különböző karakterisztikájú szűrő is.

Szakdolgozatom témája egy olyan rendszer megvalósítása, amelynek segítségével kikerülhetjük a speciális programok használatát, és a specifikációt a hardveren futó alkalmazásnak megadva, közvetlenül állíthatjuk elő a kívánt karakterisztikájú digitális szűrőt.

Munkám során megismerkedtem több digitális szűrőfajta tervezési eljárásaival, megvalósítási módjaikkal, illetve az azok során fellépő nehézségekkel. A munka során elkészült egy rendszer, amelynek segítségével több fajta véges impulzusválaszú szűrő tervezhető.

Abstract

Digital filters are one of the most basic elements of digital signal processing. They are used in a wide range of applications, from consumer electronics to professional signal measurements. From the appearance of electronic filters many filter design method was invented. One of the main reasons is that different applications needed different approaches to produce an ideal solution.

With digital filters it is a long standing method to design the filter offline with a special software then upload it to a hardver. One of the main advantages of the digital filters is that they can be reconfigured to a great extent, and by this way it is easy to implement filters with different characteristics.

My thesis is about the design of a system which can be used to design a filter online by giving the specification directly to the application running on the hardware, bypassing the special softwares used in the old way of filter design.

# Bevezető

# Digitális szűrők

Az egységimpulzusra adott válasz hosszától függően két nagy csoportra oszthatjuk a digitális szűrőket. Ez a fajta felosztás döntő jelentőséggel bír, mivel a két csoport tervezési módszerei, és műszaki paraméterei jelentősen eltérnek egymástól [1]. Véges impulzusválaszú, más néven FIR (Finite Impulse Response) szűrőkről beszélünk, ha az impulzusválasz véges időn belül lecseng, ellenkező esetben végtelen impulzusválaszú (IIR ‑ Infinite Impulse Response) szűrőkről beszélhetünk.

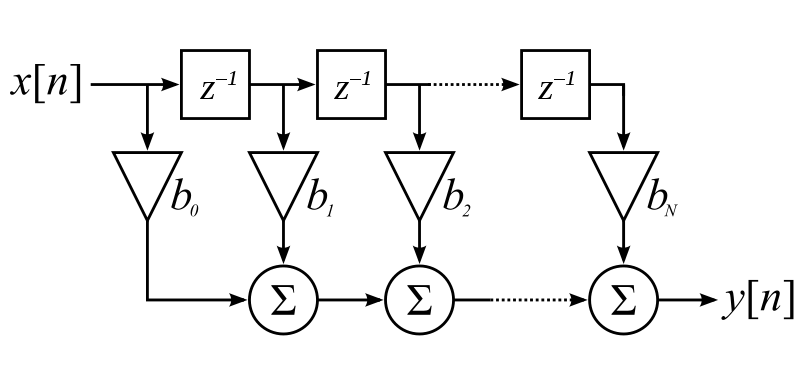
## FIR szűrők

A digitális szűrők egyik nagy osztálya a véges impulzusválaszú (FIR) szűrők [2]. Ahogy a nevük is mutatja, bemenetüket impulzussal gerjesztve, a kimeneten megjelenő válasz véges időn belül nulla lesz. A véges impulzusválaszból következik, hogy egy FIR szűrő mindig stabil. A FIR szűrő válasza az aktuális bemenet és véges számú előző bemenet súlyozott számtani átlagaként áll elő:

ahol *bi* az i-vel előbbi bemenet súlya, *N* a fokszám, *x*[*n-i*] az i-vel előbbi bemenet, és *y*[*n*] az aktuális kimenet. A fenti művelethez tartozó jelfolyamgráf a 2-1. ábrán látható; ez a FIR szűrők megvalósításának legelterjedtebb módszere. Érdemes megjegyezni azonban, hogy a megvalósítás más fajta struktúrával is lehetséges.

Impulzus gerjesztés esetén a válasz megegyezik a súlyok sorozatával:

Ennek a transzformáltjaként kapjuk a FIR szűrők általános átviteli függvényét:



2‑1. ábra. *N*-ed fokú FIR szűrő egy lehetséges megvalósítása

A FIR szűrők egyik nagy előnye, hogy könnyű úgy tervezni őket, hogy lineáris fázismenettel rendelkezzenek, ami különösen fontos, ha a jelet alakhűen kell átvinni. Másik előnyük, hogy gyors konvolúciós eljárásokkal (FFT) könnyen és hatékonyan megvalósíthatóak. A nem rekurzív megvalósításból adódóan egyrészt mindig stabilak, másrészt a megvalósítás módjából származó hibák (pl.: kerekítések) jól kézben tarthatóak.

A FIR szűrőknek előnyeik mellett hátrányaik is vannak. Ezek közül az egyik legjelentősebb, hogy nagy szelektivitás eléréséhez jóval nagyobb fokszám szükséges, mint egy IIR szűrő esetén. Ennek következménye, hogy a késleltetések, és a szűrő együtthatóinak tárolására nagyméretű memóriával kell rendelkezni. Egy FIR szűrő tervezése nagyobb számítástechnikai apparátust igényel, mint egy IIR szűrőé, ráadásul a számítástechnikai igény a fokszámmal arányosan, a lineárisnál gyorsabban növekszik, ugyanakkor bonyolult, általános és optimális amplitúdó karakterisztikák közelítése is lehetséges.

### Tervezési eljárások

FIR szűrők tervezésére számos módszert dolgoztak ki. Az alábbiakban két, egymástól lényegesen eltérő megoldás kerül bemutatásra.

#### Szűrőtervezés ablakozással [3]

Az ablakozásos módszer célja, hogy az ideális szűrő végtelen hosszú impulzusválaszát véges hosszúra csonkolja, egy megfelelően megválasztott függvény segítségével. Ennek fő oka a Gibbs - oszcilláció csökkentése, ami a frekvenciamenetben jelenlévő ugrások miatt jelentkezik. Az ablakozás nem túl hatékony módszer, de mégis gyakran alkalmazzák, mivel tervezése könnyű. A tervezés fő feladata az ablakfüggvény alakjának és hosszának meghatározása. A tervezett szűrő átviteli függvényét az ideális szűrő karakterisztikájának, és az ablakozó függvénynek a konvolúciójából kapjuk. Egy ablakozó függvény annál jobban közelíti az ideális szűrőt, minél keskenyebb a főhulláma, és annál kisebb az ingadozása, minél kisebbek a mellékhullámok. Az alábbiakban néhány ablakozó függvény kerül bemutatásra:

**Háromszög ablak**

Az első mellékhullám elnyomása 26dB.

**Általános Hamming ablak**

Hanning ablakról beszélünk, ha α=β=1. Nevezik még Hann, von Hann, vagy emelt koszinusz ablaknak is. A mellékhullámok lecsengése 18dB/oktáv.

Hamming ablakról beszélünk, ha α=0.54 β=0.46. A Hamming ablak úgy lett optimalizálva, hogy az első mellékhullám minimális legyen.

**Kaiser ablak**

ahol a nullafokú módosított első fajú Bessel függvény, *N* a fokszám, és *α* az ablak formáját megadó paraméter. Nagyobb *α* mellett szűkül a főhullám, de a mellékhullámok növekszenek, kisebb *α* mellett ennek a fordítottja igaz. A Kaiser ablak a főhullámban lévő energiasűrűséget próbálja maximalizálni.

#### Szűrőtervezés Remez-algoritmussal [4]

A FIR szűrők tervezésének egy másik változata, hogy megpróbáljuk a hiba maximumát minimalizálni. Erre a feladatra a leggyakrabban használt módszer a Remez‑algoritmus. A Remez-algoritmus egyenletes ingadozású, lineáris fázisú FIR szűrő tervezésére alkalmas. Ez a fajta közelítés Csebisev-approximáció, a szűrő átviteli karakterisztikája:

ahol a megvalósítandó karakterisztikától függően (pl.: aluláteresztő). A Remez-algoritmus úgy állítja be a együtthatókat, hogy a maximális hiba minimális legyen.

## IIR szűrők

A végtelen impulzusválaszú szűrők alkotják a digitális szűrők másik nagy csoportját [5]. A végtelen impulzusválasz elnevezés arra utal, hogy stabil esetben is végtelen ideig tart a válasz lecsengése, tart a nullához, de nem áll be véges időn belül. Megvalósításuk rekurzív struktúrákkal történik, visszacsatolást tartalmaznak.

Az IIR szűrők legtöbb hibája pont a rekurzív jellegből származik. A visszacsatolás miatt a szűrő nem feltétlenül lesz stabil, amit a véges hosszon történő számábrázolás tovább nehezít. A megvalósítás során, műveletvégzés közben általában növekszik a szóhossz (pl.: szorzásnál), amit visszacsatolás előtt vissza kell csonkolni, ami információveszteséggel jár. Ez az információveszteség jobb esetben zajnak tekinthető, rosszabb esetben azonban, ha összemérhető a jellel (alacsony bemeneti szint mellett), stabilitási problémákhoz is vezethet. Az első esetben, jó megvalósítás mellett minimalizálható a zaj hatása, azonban analitikusan nehezen vagy egyáltalán nem számítható a visszacsatolások miatt. A második eset a határciklusokra vonatkozik. Ezek akkor lépnek fel, ha alacsony jelszint mellett, a szűrő önfenntartó oszcillációba kezd a rendszer, ami lehet konstans, vagy periodikus jellegű. A számábrázolásból adódó másik probléma nagy jelszinteknél jelentkezik, ugyanis ilyenkor alul- vagy felülcsordulás léphet fel. Ez ellen a változó szaturációjával (a változó minimálisra vagy maximálisra állítása túlcsordulás esetén) védekeznek. Elvi jellegű hátrány még, hogy nem tervezhető lineáris fázismenet, mivel a stabilitás miatt nem lehet az egységkörön kívül pólus.

A végtelen impulzusválaszú szűrők egyik fő előnye, hogy létezik analóg megfelelőjük. Tervezésük is nagy részben ezen alapszik, mivel a digitális szűrők megjelenésekor az analóg szűrőtervezési módszerek már jól kiforrottak voltak. Másik előnyük, hogy a FIR szűrőkhöz képest jóval kevesebb memóriára van szükségük ugyanazon specifikáció megvalósításához.

Az IIR szűrők általános időfüggvénye:

alakú, ahol az aktuális kimenet, az *i*-vel előbbi bemenet, a *j*-vel előbbi kimenet, az *i*-vel előbbi bemenet együtthatója, a *j*-vel előbbi kimenet együtthatója. Az általános átviteli függvény:

### Tervezési eljárások

A végtelen impulzusválaszú szűrők tervezésének legkézenfekvőbb módszere az analóg szűrők transzformálása [6]. Ennek fő oka, hogy a digitális szűrők megjelenésekor az analóg szűrők tervezési eljárásai már jól kidolgozottak voltak. A tervezés menete a következő: megtervezzük a kívánt karakterisztikájú analóg aluláteresztő referens szűrőt, majd transzformáljuk a kívánt frekvenciára és formára (alul-, felül áteresztő, sáváteresztő, sávzáró), végül a kapott szűrőt diszkretizáljuk. Az így keletkezett digitális szűrő ezután implementálható egy választott megvalósítási struktúrával.

Implementálás

Frekvenciatranszformáció

Paraméterek normalizálása

Digitalizáció

Specifikáció

Transzformált szűrő

Digitális szűrő

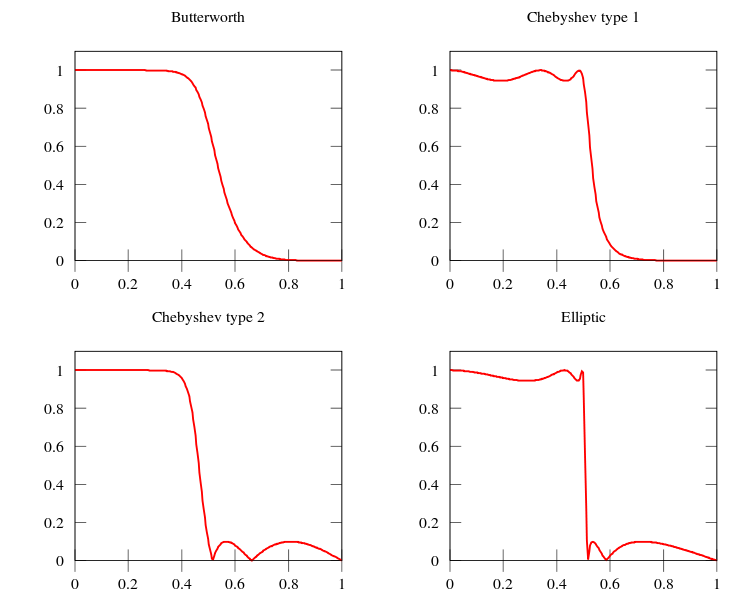
Referens szűrő

Megvalósítás

2‑2. ábra. A szűrőtervezés folyamata

#### Analóg referens szűrő tervezése

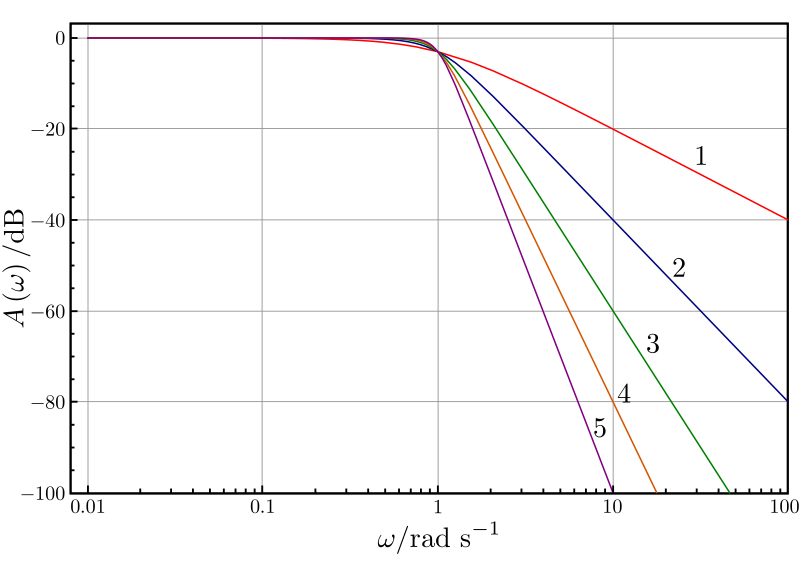
A referens szűrő egy *ω* = 1 törésponti frekvenciájú analóg aluláteresztő szűrő. Tervezésekor az ideális aluláteresztő szűrőt próbáljuk közelíteni egy adott alakú átviteli karakterisztikával (2‑3. ábra). Az approximáció során megkapjuk a választott karakterisztikának, a megadott specifikációt kielégítő, paramétereit. Az approximációra többféle eljárás ismert, a következőkben a leggyakoribb módszereket ismertetem (2‑3. ábra).



2‑3. ábra. Általánosan elterjedt approximációs típusok

**Butterworth szűrő**

A Butterworth szűrő, másik nevén maximálisan lapos szűrő, úgy közelíti az ideális aluláteresztő szűrőt, hogy az áteresztő sávban minél laposabb legyen az átviteli karakterisztika. A maximális laposság előnye, hogy az áteresztő sávba eső frekvenciakomponenseket nagyjából egységnyi erősítéssel viszi át és a fázismenete is közel van a lineárishoz. (Ez a törésponti frekvenciához közeledve leromlik.) Hátránya, hogy a többi approximációs típushoz képest kisebb a szelektivitása.



2‑4. ábra. Butterworth átviteli karakterisztikák, különböző fokszámok esetén

Átviteli karakterisztikája a következő alakban írható fel:

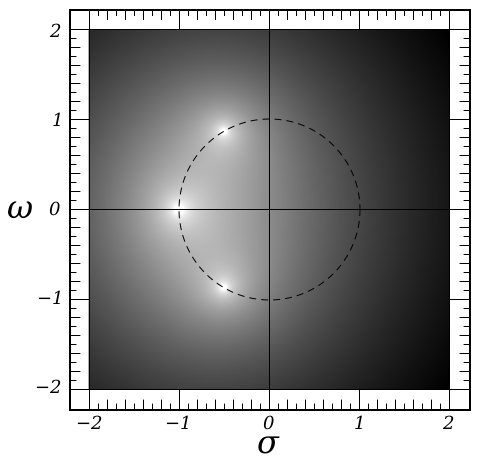
ahol *N* a fokszámot jelöli. Az *ε* értékét a legnagyobb megengedett csillapításból (az áteresztő sáv szélén) kapjuk meg:

Amennyiben a zárósáv paraméterei (záró frekvencia *Ω*z, minimális csillapítás *a*z) vannak megadva, a szűrő fokszáma a következő módon számítható:

A fokszám, és az *ε* paraméter ismeretében már ki tudjuk számolni az átviteli karakterisztika pólusait (az analóg Butterworth szűrőnek zérusai nincsenek):

ahol:

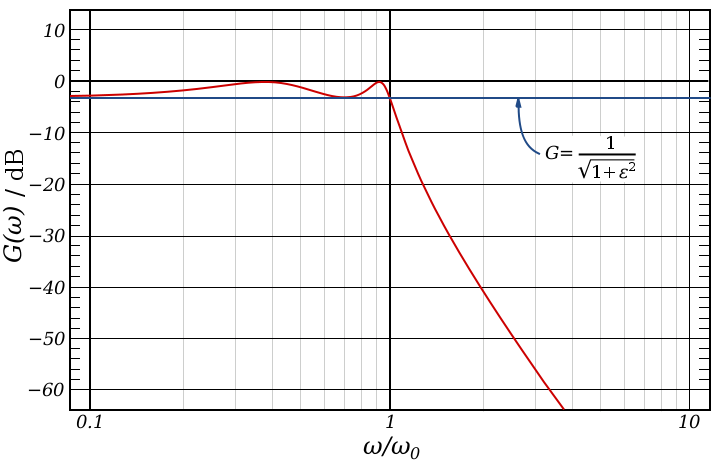
A fentebbi képletek alapján megállapíthatjuk, hogy a gyökök egy *A* sugarú kör bal felén helyezkednek el egymástól egyenletes távolságban (2‑5. ábra).



2‑5. ábra. Egy harmadfokú Butterworth szűrő pólus-zérus eloszlása (a pólusok fehérrel)

**Csebisev szűrő [7]**

A Csebisev approximáció során, a frekvenciamenet gyorsabb letörésének érdekében az áteresztő sávban feláldozzuk annak maximális laposságát, és megengedünk egy maximális ingadozást. Erre a feladatra kitűnően alkalmasok a Csebisev-polinomok, mivel a [-1,1] tartományban ±1 között egyenletesen ingadoznak, és a végtelenben a végtelenhez tartanak. A Csebisev-polinommal való közelítés következtében az áteresztő tartományban a frekvenciamenet egyenletesen fog ingadozni a specifikációban megadott értékek között, szélsőértékeit a szűrő fokszámával megegyező esetben veszi fel. A törésponti frekvencián megegyezik a maximálisan megengedett hibával, azon túl monoton csökken.



2‑6. ábra. A Csebisev szűrő amplitúdó karakterisztikája

A szűrő átviteli karakterisztikája:

ahol *N* a szűrő fokszáma, *ε* határozza meg az ingadozást, és *TN(ω)* az *N*-edrendű Csebisev-polinom:

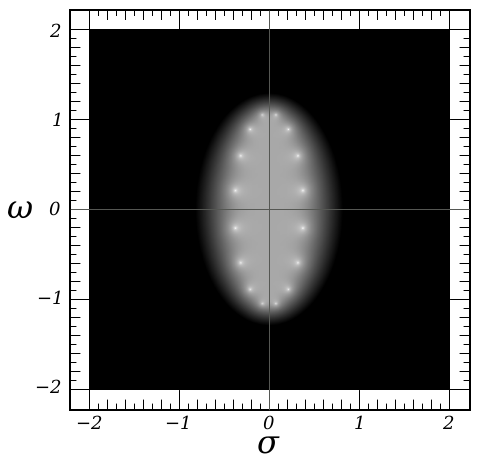
A kívánt maximális ingadozás ismeretében *ε*-t a következő képlettel számíthatjuk ki:

Ha a zárósáv paraméterei (záró frekvencia *Ωz*, minimális elnyomás *az*) vannak megadva, a szűrő fokszáma a következő módon számítható:

A szűrő fokszámának, és az *ε* ismeretében a pólusok:

ahol:

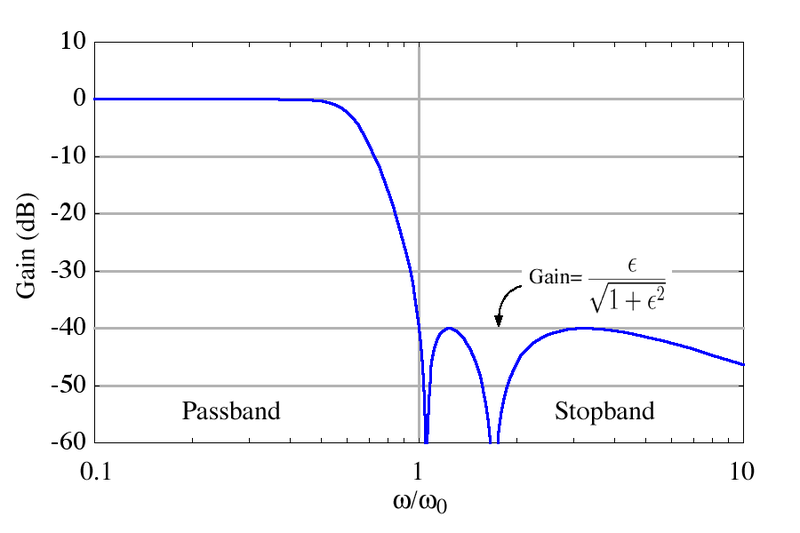
Könnyen belátható, hogy a szűrő pólusai, a komplex számsíkon ábrázolva, egy ellipszis bal ívének mentén helyezkednek el (2‑7. ábra. Csebisev szűrő pólus-zérus eloszlása2‑7. ábra.) A Csebisev szűrő átviteli függvénye, a pólusokkal felírva:



2‑7. ábra. Csebisev szűrő pólus-zérus eloszlása

**Inverz Csebisev szűrő [7]**

Az inverz Csebisev szűrő esetén az áteresztő tartományban maximálisan laposan közelítjük az ideális karakterisztikát, míg a záró tartományban megengedjük annak bizonyos ingadozását. A frekvenciamenet monoton csökken az áteresztő tartományban, a záró frekvencián megegyezik a minimális elnyomással, a felett egyenletesen ingadozik a minimális- és a végtelen elnyomás között (2‑8. ábra).



2‑8. ábra. Inverz Csebisev szűrő frekvenciamenete

A szűrő átviteli karakterisztikája, ha a záró frekvencia egységnyi, és a minimális elnyomást meghatározó tényező az *ε*:

Azonban, ha a referens szűrő általunk használt definícióját kívánjuk alkalmazni, a fentebbi egyenlet bonyolultabb formát ölt:

Az *ε* tényezőt a következőképpen számíthatjuk:

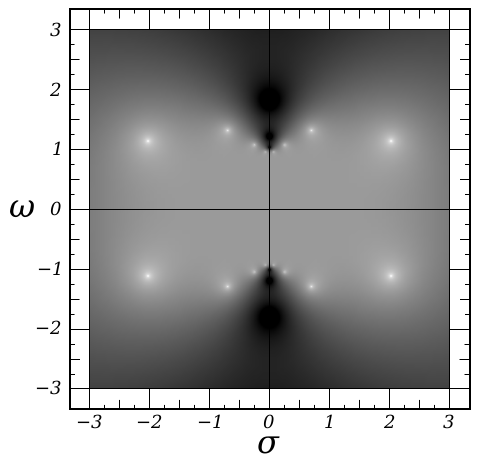
ahol *a0* a törésponti frekvencián, *az* a stop frekvencián mért erősítés.

Amennyiben nem ismerjük a szűrő fokszámát, azt az alábbi képlettel számíthatjuk:

Azonban, mivel *N*-et a következő egészre kerekítettük fel, rögzített *Ωz* esetén *ε* csökkenni fog, aminek következtében az áteresztő vagy a záró tartományban megadott specifikációt túl fogjuk teljesíteni. A lentebb bemutatott pólus-zérus számítási algoritmus a záró tartományban megadott minimális elnyomással számol, így végső soron az áteresztő tartományban lesz kisebb a maximális elnyomás. A pólusok és zérusok kiszámítása a következő módon történik:

ahol:

Az inverz Csebisev szűrő pólusokkal és zérusokkal felírt átviteli függvénye a következő:



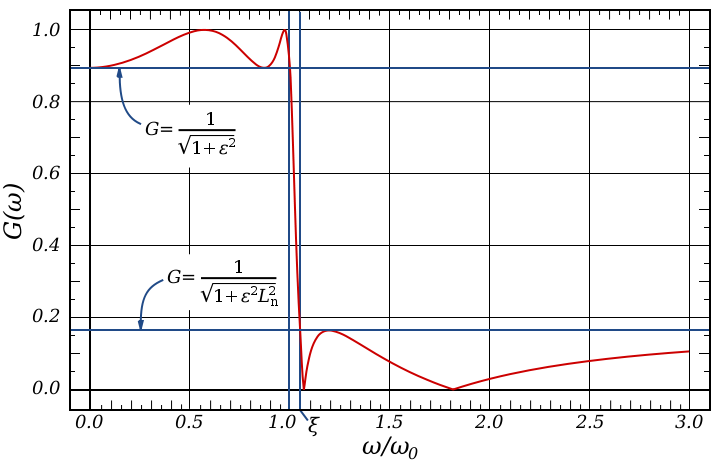
2‑9. ábra. Inverz Csebisev szűrő pólus-zérus eloszlása

**Elliptikus (Cauer) szűrő [8]**

A Cauer szűrő esetében, a szelektivitás növelésének érdekében további engedményeket teszünk a karakterisztika lapossága felé. Mind az áteresztő, mind a záró tartományban megengedünk egy bizonyos fokú ingadozást (2‑10. ábra). Az átviteli függvény a következő:

ahol *GN(ω)* a Csebisev-polinom általánosítása, számításához elliptikus integrálokra van szükség. A polinom nem csak *ω* függvénye, így lehetőség van különböző mértékű ingadozás felírására a két tartományban.

Az elliptikus szűrő számítása, az elliptikus integrálok miatt, jóval nehezebb, mint az első három tárgyalt szűrőé, részletezésével nem foglalkozunk.



2‑10. ábra. Elliptikus szűrő amplitúdó karakterisztikája

#### Transzformálás [8]

A frekvencia-transzformáció két lépésből áll. Miután megkaptuk a referens aluláteresztő szűrőt, azt át kell alakítani a kívánt formájúvá, majd el kell tolni a specifikációban megadott frekvenciára.

**Aluláteresztő**

Aluláteresztő szűrő esetén az első lépés kihagyható. A kívánt karakterisztika az

helyettesítéssel kapható. Ez esetben a pólusok és a zérusok a következőképpen módosulnak:

**Felüláteresztő**

Felüláteresztő szűrő esetén a helyettesítés a következő:

A pólusok és zérusok megváltozása:

Ezen felül pólus esetén bejön egy differenciátor, illetve zérus esetén egy integrátor.

**Sáváteresztő**

Sáváteresztő szűrő esetében már bonyolultabb az átalakítás. A behelyettesítés:

ahol ω0 a sávközépi frekvencia, ωd a sávszélesség, és teljesülnek a következők:

Ha elvégezzük a behelyettesítést egy aluláteresztő tagba, akkor az a következőképpen fog alakulni (felüláteresztő tag esetén a reciproka):

A pólusok és zérusok megváltozása itt már bonyolultabban adódik:

**Sávzáró**

Sávzáró transzformációnál a behelyettesítendő a sáváteresztő reciproka:

ω0-ra és ωd-re a sáváteresztőnél megadott feltételek vonatkoznak. A transzformált aluláteresztő tag:

A transzformáció az eredeti egy pólust kettőre bontja szét, és behoz egy tisztán képzetes zéruspárt az ω0 frekvenciára. Az új pólusok a sáváteresztőhöz hasonló módon számíthatóak:

#### Áttérés a diszkrét időtartományba

Eddig a pontig gyakorlatilag egy analóg szűrőt terveztünk meg. Az alábbiakban néhány olyan módszer kerül bemutatásra, amelynek segítségével áttérhetünk a diszkrét időtartományba:

**Impulzus-invariáns transzformáció [9]**

Ennél a módszernél a digitális- és az analóg szűrő impulzusválaszát próbáljuk összhangba hozni. A módszer alapja az folytonos idejű impulzusválasz mintavételezése:

ahol a diszkrét-, a folytonos impulzusválasz és a mintavételi idő. Az impulzusválaszokat transzformálva:

Az átalakítás során a pólusokat a leképzéssel kapjuk meg, a zérusokról nincs közvetlen információnk, meghatározásuk az és alapján történik. Az átalakítás a folytonos rendszer stabil pólusait stabil diszkrét pólusba viszi át. Az impulzus-invariáns transzformáció során átlapolódás léphet fel, ezért ügyelni kell a jel sávhatároltságára.

**Illesztett Z-transzformáció [10]**

Az illesztett z-transzformációnál mind a pólusokat, mind a zérusokat a leképzéssel visszük át a z-tartományba.

ahol zérus, pólus és a mintavételi idő. Az illesztett-z transzformációra is igaz, hogy stabil folytonos pólust stabil diszkrét pólusba visz át, illetve ebben az esetben is ügyelni kell a jel sávhatároltságára, különben átlapolódás léphet fel.

**Bilineáris transzformáció [11]**

A bilineáris transzformáció, jó tulajdonságainak köszönhetően, a leggyakrabban használt leképzési módszer. Megőrzi a rendszer stabilitását, az s-sík bal felét a z-síkon az egységkörön belülre képzi le, míg képzetes tengelyt magára az egységkörre. Ugyanúgy, ha a szűrő minimál fázisú volt folytonos időben, ezt a jellegét megtartja a diszkrét időtartományban is. A leképzés a következő behelyettesítéssel hajtható végre:

A paraméter egy skálázási tényező, feladata a digitális és analóg frekvenciapontok megfelelő átvitele. Alapesetben a nulla frekvencia kerül pontos átvitelre, ami esetén igaz. A kifejezést visszahelyettesítve megkaphatjuk a diszkrét és a folytonos frekvenciatengely közötti összefüggést:

ahol a folytonos idejű körfrekvencia, a diszkrét körfrekvencia, és a mintavételi idő. A fenti képlet alapján látszik, hogy a bilineáris transzformáció nemlineáris torzítással felelteti meg az analóg és a diszkrét frekvenciatengelyt. Ennek a nemlineáris leképzésnek az egyik legfontosabb következménye, hogy megakadályozza az átlapolódást. Ugyanakkor megváltoztatja a töréspontok (pólusok, zérusok) egymáshoz viszonyított helyzetét, és ezzel módosítja az amplitúdó- és fáziskarakterisztikát. A frekvenciatorzítás hatásainak csökkentése érdekében vagy előtorzítjuk az analóg körfrekvenciát:

ahol a kívánt diszkrét körfrekvencia, vagy a paraméterrel közvetlenül beállíthatjuk, hogy melyik frekvencia legyen pontosan leképezve:

ami esetén megegyezik az alap -sel.

Egy aluláteresztő tag bilineáris transzformáltja a következő:

ahol a folytonos-, a diszkrét pólus, és igaz, hogy:

### Megvalósítás

Az IIR szűrők implementációjára általánosan jellemző a rekurzív jelleg. Sok féle megvalósítási módszer létezik, az alábbiakban ezek közül kerül néhány részletesebb ismertetésre.

#### Direkt formák [12]

A direkt formák főként elméleti jelentőséggel bírnak. Előnyük, hogy közvetlenül felírhatóak az átviteli függvényből, ugyanakkor nagyobb fokszám esetén (akár kettő fölött is), az együtthatók és a gyökök közötti nemlineáris összefüggés miatt, az együtthatók kis megváltozása nagy gyökeltérést okozhat, ami jobb esetben a karakterisztika megváltozásával jár, rosszabb esetben pedig akár instabilitáshoz is vezethet. A fenti okokból kifolyólag kettőnél magasabb fokszám mellett általában nem használnak direkt megvalósítási formákat, azonban másodfokú rendszereknél még jól kezelhetőek. A négy fajta direkt forma, másodfokú esetben, a következő pontban kerül részletes ismertetésre.

#### Biquad

A biquad egy másodfokú rekurzív szűrőtag. Neve arra utal, hogy átviteli függvénye két másodfokú függvény hányadosaként áll elő:

ahol , a két zérus, , a két pólus. Egy biquad tagnál általában célszerű minél közelebbi pólusokat és zérusokat választani, hogy ne legyenek nagy ugrások a karakterisztikában.

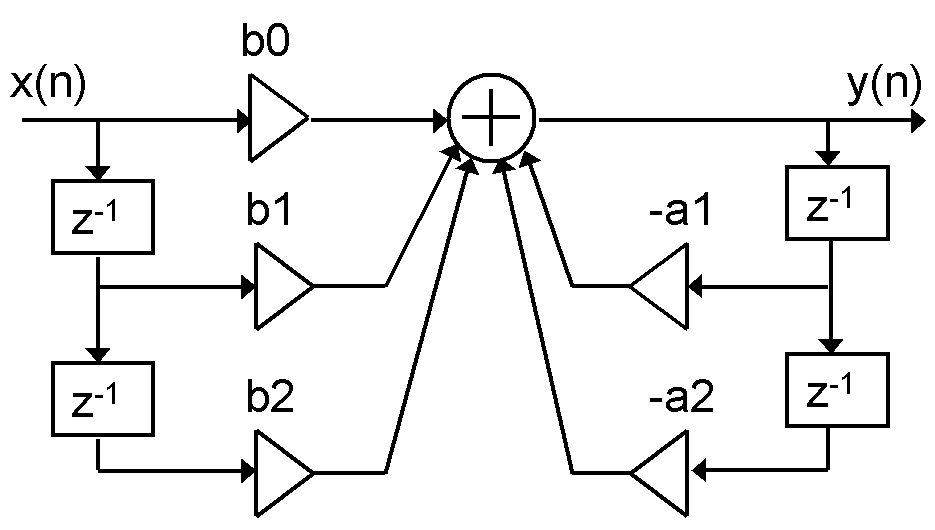
Blackfin_537_EZ-Kit-Lite61.png

2‑11. ábra kamu

A fenti átviteli függvény többféleképpen is megvalósítható. Az alábbiakban a különböző másodfokú direkt formák kerülnek részletezésre, de ezeken kívül még rengeteg megvalósítási módszer létezik.

**Direkt I.**

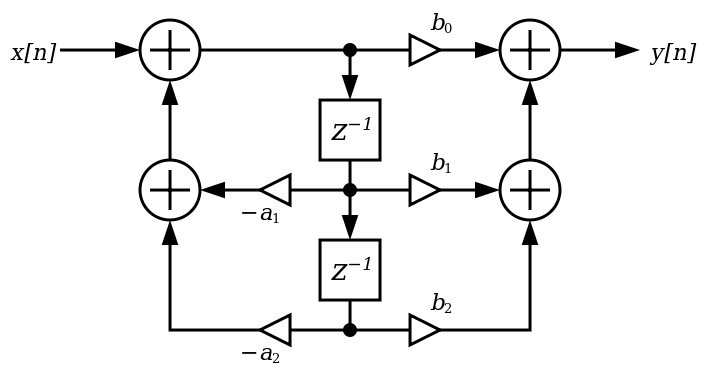
Az egyes direkt forma a legnyilvánvalóbban felírható megvalósítási mód. Megvalósítása négy késleltető tagot, öt szorzót, és egy akkumulátort igényel. A struktúra érdekessége, hogy kettes komplemens számábrázolás esetén, az egy darab akkumulátor miatt, nincs belső túlcsordulás, elég arra ügyelni, hogy a végső kimenetnél szaturáljuk a jelet, ha túlcsordulás lépne fel. A struktúra jelfolyamgráfja a 2‑12. ábraán látható.



2‑12. ábra. Direkt I-es forma jelfolyamgráfja

**Direkt II.**

A kettes direkt forma az átviteli függvény kanonikus (minimális késleltetőt tartalmazó) megvalósítási módja. A struktúrához tartozó jelfolyamgráf a 2‑13. ábraán látható.

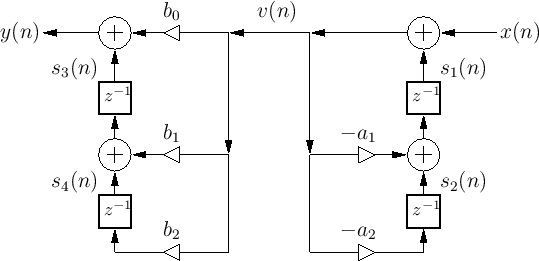


2‑13. ábra. Direkt II-es forma jelfolyamgráfja

Két késleltető tagot, két akkumulátort, és öt szorzót tartalmaz. Az első késleltető bemeneténél, és a kimenetnél túlcsordulás léphet fel, ez ellen védekezni kell (pl.: szaturációval). A struktúrát leíró időfüggvények az alábbiak:

**Transzponált Direkt I. [13]**

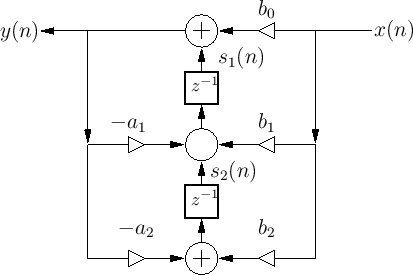
Ritkán használt struktúra, mivel sok késleltetőt és összeadót tartalmaz, és ezek több zajforrást jelentenek a rendszerben. Jelfolyamgráfja a 2‑14. ábraán látható.



2‑14. ábra. Transzponált Direkt I-es forma jelfolyamgráfja

**Transzponált Direkt II. [13]**

A kettes direkt forma transzponált alakja. Előnye az előbbivel szemben, hogy ha felbontanánk egy-egy csak zérusokból és csak pólusokból álló tagra, akkor a jel előbb haladna át a tisztán zérusokból álló tagon, ami a gyakran használt szűrőknél általában kedvezőbb. Jelfolyamgráfja a 2‑15. ábrán látható.

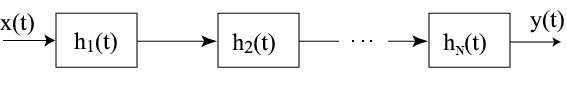


2‑15. ábra. Transzponált Direkt II-es forma jelfolyamgráfja

#### Soros és párhuzamos felbontás

A soros és párhuzamos felbontás során úgy próbáljuk meg kisebb egységekre bontani a szűrőt, hogy ezek az alaptagok párhuzamosan, vagy sorosan kapcsolva az eredeti szűrő karakterisztikáját adják. Leggyakrabban másodfokú (biquad) tagokra történik a felbontás. Soros kapcsolás esetén egyszerűbb a helyzet, csak ki kell választani a megfelelő pólus- és zérus párokat:

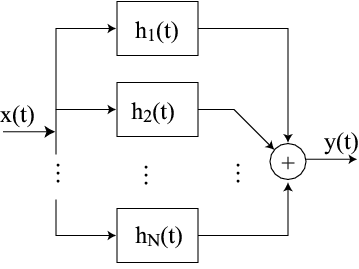
ahol , a *k*-adik zéruspár, , a *k*-adik póluspár, és *K* az eredő erősítés. Megoldandó feladat a pólusok és zérusok megfelelő összerendelése, illetve az alaptagok sorrendjének megállapítása. Az előbbi megoldására általában úgy rendelik egymáshoz a pólusokat és zérusokat, hogy azok minél közelebb essenek egymáshoz, hogy a karakterisztikában ne legyenek nagy kiemelések vagy elnyomások. A másik feladat megoldása bonyolultabb, általában számítógépes analízissel történik. Fő célja, hogy a bemenet, és a kaszkád tagok közötti átvitelben fellépő kiemeléseket és elnyomásokat minimalizálja. A kaszkád kapcsolás jelfolyamgráfja a 2‑16. ábraán látható.



2‑16. ábra. Alaptagok kaszkád (soros) kapcsolása

Párhuzamos megvalósítás esetén, az eredeti átviteli függvényt a részlettörtekre bontás módszerével bontjuk fel:

Ebben az esetben nem probléma az alaptagok kapcsolási sorrendje, azonban ez a fajta megvalósítás nagyobb bitszámot, és több szorzót igényel. A párhuzamos kapcsolás jelfolyamgráfja a 2‑17. ábraán látható.



2‑17. ábra. Alaptagok párhuzamos kapcsolása

# Fejlesztői környezet

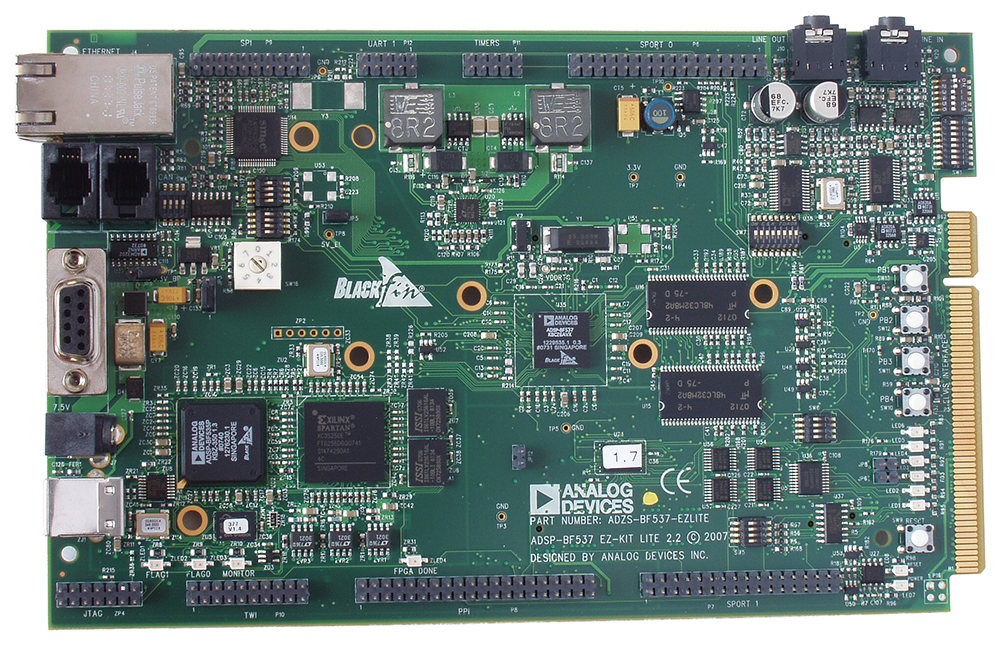
## ADSP-BF537 EZ-KIT Lite

Az ADSP-BF537 EZ-KIT Lite fejlesztői kártya (3‑1. ábra) segítségével még a célhardver megépítése előtt kipróbálhatjuk az ADSP-BF537 Blackfin processzort. Általános célú felhasználásra készült, hogy a processzor minél több funkcióját megismerhessük. A fejlesztői kártya igen gazdag perifériakészlettel rendelkezik. A teljesség igénye nélkül, néhány főbb jellemzője [14]:

* IEEE 802.3 10/100 Ethernet
* CAN 2.0B
* RS-232 UART meghajtó
* 96 kHz sztereó ADC/DAC
* 64MB SDRAM
* 4MB FLASH memória

A fejlesztői kártya tervezésekor fontos szempont volt annak bővíthetősége. A kártya több csatlakozóval rendelkezik, amik a feladattól függően, többféle funkcióra programozhatóak. Többek között található rajta soros-, párhuzamos-, és USB port, illetve UART-, TWI-, SPI-, JTAG-, és ELVIS interfész is.

A közvetlen kapcsolat érdekében a kártya rendelkezik négy (+1 RESET) gombbal, illetve hat, általánosan felhasználható LED-del is.



3‑1. ábra. ADSP-BF537 EZ-KIT Lite fejlesztői kártya

### ADSP-BF537 Blackfin processzor

Az ADSP-BF537 egy, az Analog Devices által kifejlesztett jelfeldolgozó processzorok Blackfin családjába tartozó, nagy teljesítményű DSP [15]. A processzor, az utasításkészletet tekintve, csökkentett utasításkészletű (RISC), és egy utasítással több adaton is tud dolgozni (SIMD). Az aritmetikai egységek 16-/32-bitesek, két akkumulátora 40 bites. Az aritmetikai utasítások bemenetei alapvetően 16 bitesek, míg kimeneteik 32 bitesek. A BF537 jelfeldolgozó processzor néhány egyéb tulajdonsága:

* 600MHz-es órajel (maximum)
* 132KB beépített SRAM
* állítható cache
* Ethernet MAC
* CAN 2.0B interfész

### ADC és DAC

A BF537 Blackfin processzor audió interfésze felől érkező jelek digitális-analóg átalakításáról az Analog Devices AD1854 digitális-analóg konvertere gondoskodik. Az AD1858 egy 96kHz-es, multibites szigma-delta elven működő, 24 bites, sztereó átalakító, tipikus dinamikatartománya körülbelül 115dB [16]. A fejlesztői kártya 24 bites módban, 48kHz-es mintavételi frekvenciával használja a konvertert.

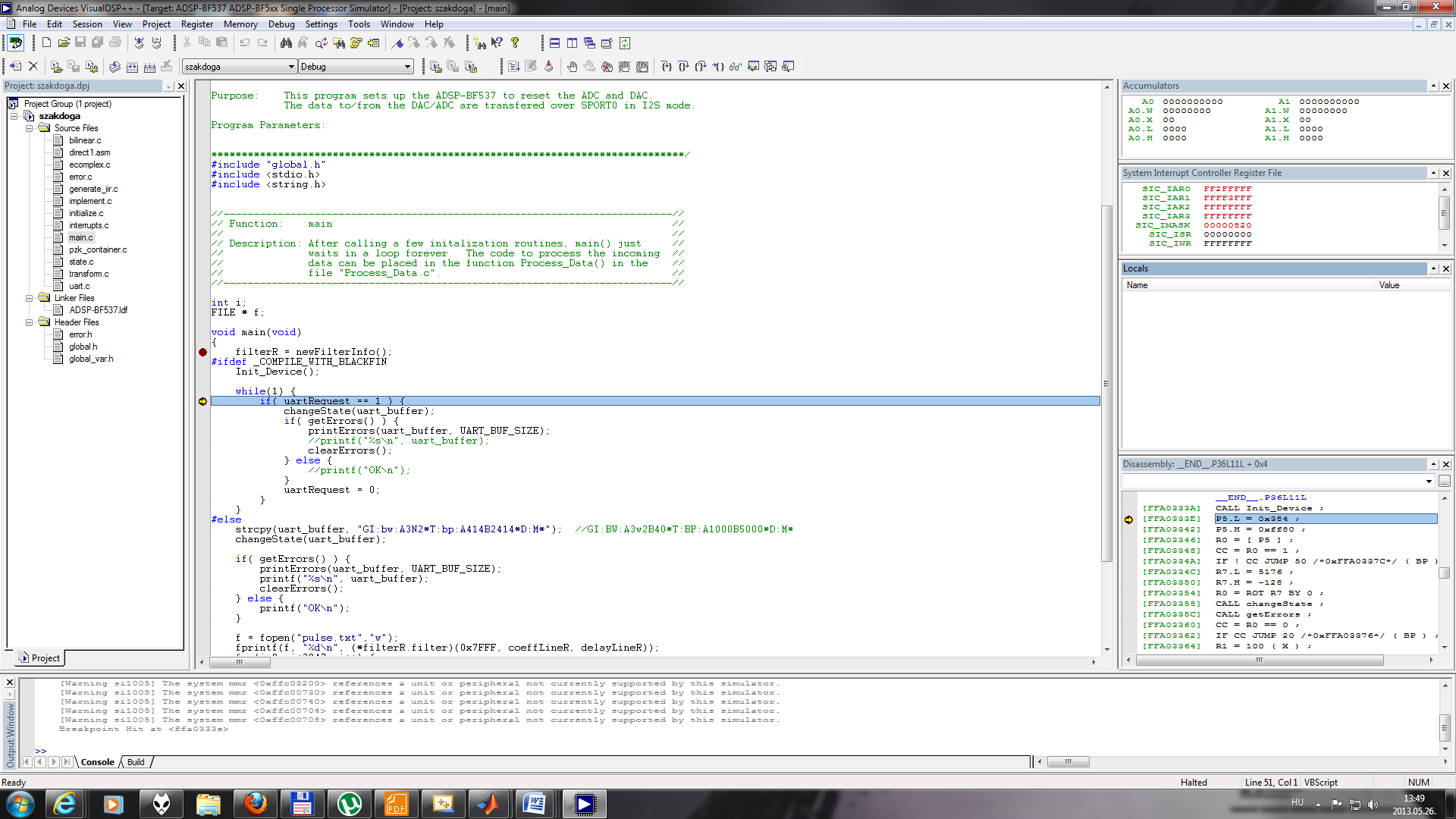
Az audió jelek digitalizálásáért az AD1871 analóg-digitális átalakító felel. Az AD1854-hez hasonlóan ez is 96kHz-es, multibites szigma-delta elven működik, szintén 24 bites, és két csatornás. Tipikus dinamikatartománya körülbelül 105dB [17]. A fejlesztői kártya 24 bites módban használja. Átalakítás akkor történik, ha adat érkezik a DSP felől.

Az ADC és a DAC a processzor SPORT0 portjára csatlakozik, két vezetékes (TWI) módon, de megfelelő kapcsolóállás mellett leválaszthatóak róla.

## Visual DSP++ 5.0

A Visual DSP++ az Analog Devices által, a saját termékeihez kifejlesztett fejlesztő szoftver [18]. Saját C/C++ és assembly fordítóval, és linkerrel rendelkezik, amikhez tartozik grafikus felület is, így nem kell bonyolult parancssori utasítások kiadásával törődni. A program több előre elkészített könyvtárat tartalmaz, rengeteg kész szubrutin áll a fejlesztő rendelkezésére, legtöbbjük kifejezetten a jelfeldolgozó processzorra optimalizáltan.

A fejlesztői környezet erős hibakereső eszközöket is tartalmaz. Többek között lépésről lépésre vizsgálhatjuk a kész assembly kódot, megtekinthetjük a regiszterek és a memória tartalmát, és akár módosíthatjuk is azokat. A 3‑2. ábraán egy hibakeresés közben készített képet láthatunk.



3‑2. ábra. VisualDSP++ 5.0 fejlesztői környezet

# A feladat megvalósítása

## Specifikáció és rendszerterv

### A feladat ismertetése

A szakdolgozat célja egy olyan konfigurálható rendszer létrehozása, amelynek segítségével, speciális külső programok használata nélkül, többféle szűrőkarakterisztika valósítható meg. Feladatom kifejezetten a végtelen impulzusválaszú (IIR) szűrők megvalósításával foglalkozik, mivel tervezésük változatosabb, és megvalósításuk során több probléma merülhet fel, mint a véges impulzusválaszú (FIR) szűrők esetében. Ugyanakkor a tervezés során végig szempont volt, hogy későbbiekben FIR szűrők tervezése is implementálható legyen a programba.

A feladat elvégzéséhez az Analog Devices ADSP-BF537 EZ-KIT Lite fejlesztői kártyája és a Visual DSP++ 5.0 fejlesztői környezet (lásd: 3. fejezet) állt a rendelkezésemre. A feladat végrehajtásához a fejlesztői kártya következő egységeire van szükség:

* DSP
* audió interfész
* UART (RS-232)

A megvalósítandó program a DSP-n fut. Alapvetően két részre bontható: a szűrőtervezésre, és magára a szűrési folyamatra. A szűrő bemeneti jele az audió interfész bemeneti oldalán érkezik, a szűrt jel az interfész kimeneti felére van kivezetve. A szűrőtervezéshez szükségesek a szűrő paraméterei. Ezeket a felhasználónak kell megadnia, ezért szükség, hogy tudjon kommunikálni az eszközzel.

**A program feladatai:**

bemenő jel szűrése

kommunikáció a felhasználóval

szűrőtervezés

A szűrőtervezés során elvárás, hogy a következő tulajdonságokat a felhasználó határozhassa meg:

* jelleg
* Butterworth
* Csebisev
* Inverz Csebisev
* frekvencia
* alak
* aluláteresztő
* felüláteresztő
* sáváteresztő
* sávzáró
* digitalizálás módszere
* realizálás

### Rendszerterv

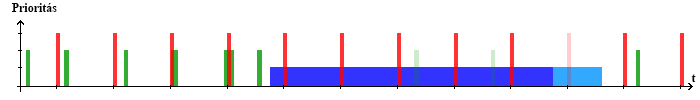
A rendszerterv elkészítése során figyelembe kell venni a program által végrehajtott feladatok időtartamát és fontossági sorrendjét.

A legidőkritikusabb feladat maga a szűrés, mivel ha nem áll időben rendelkezésre a következő kimeneti adat, az a jel torzulásához vezethet. Szerencsére a szűrés végrehajtása jóval kevesebb időt vesz igénybe, a többi feladathoz képest. Ezen okokból kifolyólag célszerű a szűrést egy megszakítási szubrutinban lekezelni úgy, hogy közben más megszakítás ne juthasson érvényre.

A következő feladat a kommunikáció. Ez kevésbé kritikusabb, mint a szűrési folyamat, ráadásul mindig meg kell előznie a szűrőtervezést, mivel a kommunikáció nyugtázásáig nem állnak rendelkezésre a tervezési paraméterek. A kommunikáció során karakterenként érkezik a tervezési paramétereket tartalmazó kódszó. Ez azt jelenti, hogy a kommunikáció időben elhúzódhat, de egy-egy karakter beolvasása rövid ideig tart. Szűrőtervezés közben nem elvárás, hogy működjön a kommunikáció, ráadásul ez akár felül is írhatná az előző kódszót, ami hibás működéshez is vezethetne, ezért a tervezési folyamat idejére le kell tiltani. A fenti okok miatt, a karakterek beolvasását egy a szűrésnél kisebb prioritású megszakítás kezelő rutinban célszerű megoldani, majd a nyugtázás után jelezni a főprogramban, hogy új üzenet érkezett.

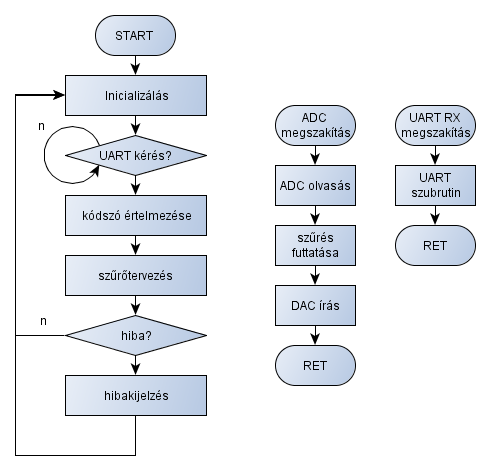
A harmadik feladat maga a szűrőtervezés. Végrehajtása nem időkritikus, ugyanakkor nagy a számításigénye, és sokáig tart. A végrehajtás egy új üzenet érkezésekor kezdődik, és befejezéséig nem is fogad újabbakat. Természetesen a szűrés megszakíthatja a tervezést. A tervezés során lebegőpontos számokkal dolgozunk, mivel a felhasznált változók nagyságrendje nem ismert. A folyamat végén ideiglenesen letiltásra kerül a szűrés, amíg az új szűrő implementálása megtörténik, majd az újraengedélyezés után már az új szűrő fog futni, és a kommunikáció is újra aktív lesz.

A 4‑1. ábrán a program idődiagramját láthatjuk. Pirossal jelölve a szűrés megszakításai, zölddel a karakterolvasások, sötét kékkel a szűrőtervezés, világos kékkel az implementálás. A halványabb színek a letiltott megszakításokat jelentik.



4‑1. ábra. A program idődiagramja

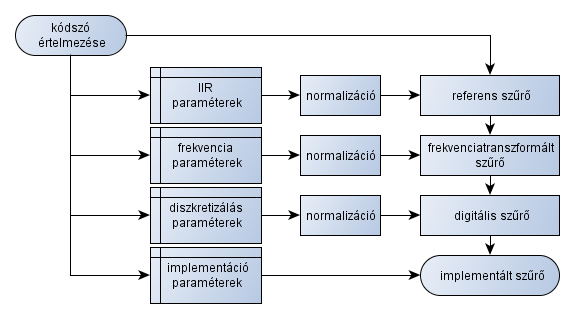
A fenti megfontolások alapján felrajzolható a program működésének folyamatábrája. A 4‑2. ábráról leolvasható, hogy az időkritikus feladatok végrehajtása a megszakítási folyamaton belül történik, míg a nem kritikus részek a programtörzsben foglalnak helyet.



4‑2. ábra. A program működésének folyamatábrája

## Szűrőtervezés

A szűrőtervezés a 2.2.1. fejezetben ismertetett módszerek alapján történik. A következőkben ezeknek a módszereknek a tényleges megvalósítását ismertetem. A tervezés folyamata a 4‑3. ábrán látható.



4‑3. ábra. A szűrőtervezés folyamatábrája

### A szűrő reprezentációja

A program a szűrők pólusait és zérusait használja a különböző számítások során, ezért célszerű ezeket, és az erősítést tárolni. Erre szolgál a *pzkContainer* elnevezésű struktúra:

real amp, // erősítés

wz; // tisztán képzetes zérus/pólus

complex \* poles; // pólusok

complex \* zeros; // zérusok

uint pSize, // pólustömb mérete

zSize, // zérustömb mérete

nextPole, // következő pólus helye

nextZero; // következő zérus helye

int no\_wz; // a tisztán képzetes zérusok száma (negatív érték esetén pólusok)

A pólusok és zérusok komplex számként vannak elmentve, és a konjugált párok közül csak a pozitív képzetes részű van tárolva, mivel a program azzal a feltételezéssel él, hogy a komplex számok mindig párban vannak, míg a tisztán valósak sosem. Ennek oka a felhasznált memória csökkentése. Szintén a memória-felhasználás csökkentése miatt vannak elmentve a tisztán képzetes frekvenciák (beleértve a differenciátor- és integrátor tagot is), mivel az aluláteresztő szűrőn kívül minden egyéb szűrő sok azonos ilyen jellegű zérust (vagy pólust) tartalmaz.

A tervezés paraméterei, és a folyamat során kapott szűrők a *filterInfo* struktúrában vannak tárolva:

iirParameters iirP; // a referens szűrő paraméterei

transformParameters transformP; // a transzformál szűrő paraméterei

pzkContainer \*iFilter, // a referens szűrő

\*tFilter, // a transzformált szűrő

\*dFilter; // a digitális szűrő

// a szűrést végző függvény:

fract32 (\*filter)(fract16 input, fract16 \* coeffs, fract16 \* delays);

filterType type, // a frekvenciamenet alakja

subtype, // a szűrő jellege (pl.: Butterworth)

supertype; // iir, fir

### Referens szűrő készítése

A referens szűrők tervezése a 2.2.1.1. fejezetben ismertetett algoritmusok alapján történik. A kódszóban megadott specifikáció alapján a normalizáló függvények kiszámítják a tervezéshez szükséges paramétereket, majd azokat átadják a generátor függvényeknek.

A normalizáló függvények:

int convertParametersForButterworth( iirParameters \* ip );

int convertParametersForChebyshev1( iirParameters \* ip );

int convertParametersForChebyshev2( iirParameters \* ip );

A generátor függvények:

pzkContainer \* createButterworth(uint n, real e0);

pzkContainer \* createChebyshev1(uint n, real e0);

pzkContainer \* createChebyshev2(uint n, real Os, real d2);

### Frekvencia transzformáció

A frekvencia transzformáció során, a referens szűrőt a megfelelő frekvenciára toljuk el, és a kívánt alakúra módosítjuk. A felhasznált algoritmusok a 2.2.1.2. fejezetben találhatóak. Első lépésként a specifikációban megadott értékeket normalizálja egy függvény, majd a megfelelő transzformáló függvénynek adja a kapott paramétereket, és a referens szűrőt. A normalizáló függvény:

int normalizeTransformParameters(transformParameters \*tp);

A transzformáló függvények:

pzkContainer \* t2lp(pzkContainer \* pzk, real w0);

pzkContainer \* t2hp(pzkContainer \* pzk, real w0);

pzkContainer \* t2bp(pzkContainer \* pzk, real w0, real dw);

pzkContainer \* t2bs(pzkContainer \* pzk, real w0, real dw);

### Áttérés a diszkrét időtartományra

A transzformált szűrő átalakítása a 2.2.1.3. fejezetben ismertetett bilineáris transzformáció alapján történik. A specifikáció során megadható, hogy milyen frekvencián kívánjuk pontosan tartani az amplitúdó karakterisztikát, vagy ráhagyhatjuk ezt a programra, amely esetben a törésponti frekvenciát tekinti iránymutatónak. A bilineáris leképzést végrehajtó függvény:

pzkContainer \* bilinear(pzkContainer \* pzk, real pwf);

### A digitális függvény implementálása

A szűrőtervezés utolsó része a megtervezett digitális szűrő megvalósítása. A 2.2.2. fejezetben bemutatott megvalósítási módok közül az egyes direkt formájú biquadokból felépített kaszkád struktúrát valósítottam meg.

#### Módosított Direkt I-es forma

Az elméleti részben bemutatott egyes direkt forma (2.2.2. fejezet) közvetlen nem alkalmazható. Az okok felderítéséhez fel kell írni a gyöktényezős-, és a felbontott alakú átvitelek közötti összefüggést:

A DSP az együtthatókat a [-1;1) tartományban tudja tárolni. A pólusokról tudjuk, hogy abszolút értékük kisebb, mint egy, mivel stabil a rendszer. Ez esetben a két pólus összegéről csak annyit tudunk, hogy abszolút értéke kisebb, mint kettő. Célszerűnek tűnik ennek a felét tárolni, és a szűrés során kétszer venni. Igaz ez a zérusok esetén is, de csak akkor, ha *K* segítségével úgy skálázzuk őket, hogy abszolút értékük kisebb legyen, mint egy. Az együtthatók pontosságának érdekében célszerű, úgy.

## Vezérlés

A rendszer vezérlése a 4‑2. ábra és 4‑3. ábra alapján történik. K

## Kommunikáció

A rendszer soros porton keresztül kommunikál a felhasználóval. A szűrőtervezéshez szükséges paramétereket egy tömör kódszó segítségével lehet megadni. A

# Mérési eredmények

Irodalomjegyzék

1. Dr. Simonyi Ernő: Digitális szűrők (Műszaki Könyvkiadó, 1984)  
   1.4.1. A FIR és az IIR szűrők osztálya
2. Dr. Simonyi Ernő: Digitális szűrők (Műszaki Könyvkiadó, 1984)  
   5. Véges impulzusválaszú (FIR) szűrők tervezése és alkalmazása
3. Dr. Simonyi Ernő: Digitális szűrők (Műszaki Könyvkiadó, 1984)  
   5.2.Tervezés ablakozási módszerekkel
4. Dr. Simonyi Ernő: Digitális szűrők (Műszaki Könyvkiadó, 1984)  
   5.4. Optimális FIR szűrők tervezése
5. Dr. Simonyi Ernő: Digitális szűrők (Műszaki Könyvkiadó, 1984)  
   6. Végtelen impulzusválaszú (IIR) szűrők tervezése és alkalmazása
6. Dr. Simonyi Ernő: Digitális szűrők (Műszaki Könyvkiadó, 1984)  
   6.1. IIR szűrők tervezése analóg szűrők transzformációjával
7. John Stensby: Chebyshev filters, EE648 08/31/11  
   <http://www.ece.uah.edu/courses/ee426/Chebyshev.pdf>
8. BME MIT Tanszéki Munkaközösség: Digitális jelfeldolgozás  
   (Műegyetemi Kiadó, 2004)  
   6.2. IIR szűrők tervezése
9. Dr. Simonyi Ernő: Digitális szűrők (Műszaki Könyvkiadó, 1984)  
   6.1.1. Az impulzus invariáns transzformáció
10. Dr. Simonyi Ernő: Digitális szűrők (Műszaki Könyvkiadó, 1984)  
    6.1.2. Az illesztett z-transzformáció
11. Dr. Simonyi Ernő: Digitális szűrők (Műszaki Könyvkiadó, 1984)  
    6.2. Tervezés a bilineáris transzformációval
12. Dr. Simonyi Ernő: Digitális szűrők (Műszaki Könyvkiadó, 1984)  
    6.2.2. A H(z) függvény előállítása és megvalósítása
13. Transposed Direct-Forms  
    <https://ccrma.stanford.edu/~jos/fp/Transposed_Direct_Forms.html>
14. ADSP-BF537 EZ-KIT Lite Manual  
    <http://www.analog.com/static/imported-files/eval_kit_manuals/ADSP-BF537_EZ-KIT_Lite_Manual_Rev_2_4.pdf>
15. ADSP- BF537 Blackfin DSP datasheet  
    <http://www.analog.com/static/imported-files/data_sheets/ADSP-BF534_BF536_BF537.pdf>
16. AD1854 DAC datasheet  
    <http://www.analog.com/static/imported-files/data_sheets/AD1854.pdf>
17. AD1871 ADC datasheet  
    <http://www.analog.com/static/imported-files/data_sheets/AD1871.pdf>
18. Visual DSP++ Manual  
    <http://www.analog.com/static/imported-files/software_manuals/10910966550_gs_guide.pdf>

Függelék