TARTU ÜLIKOOL

LOODUS- JA TEHNOLOOGIATEADUSKOND

Füüsika Instituut

Karl Tiirik

# PRAKTILISED ÜLESANDED DIGITAALSE SIGNAALITÖÖTLUSE MEETODITEGA TUTVUMISEKS

Bakalaureusetöö (12 EAP)

Juhendaja: Fred Valk, MSc

Luban töö kaitsmisele:

Juhendaja .....

Programmijuht .....

*allkiri, kuupäev*

Tartu 2012

# SISUKORD

SISSEJUHATUS ..... 4

KASUTATUD LÜHENDID JA DEFINITSIOONID ..... 5

1. ÜLEVAADE DIGITAALSE SIGNAALITÖÖTLUSE ARENGUST JA RIISTVARAST ..... 8

1.1 Digitaalse signaalitöötluse teke ja areng ..... 8

1.2 Moodne digitaalse signaalitöötluse riistvara ..... 10

2. TI TMS320VC5510 PROTSESSORI ÜLEVAADE ..... 13

2.1 TMS320VC55x DSP ..... 13

2.2 TMS320VC55x protsessori mälu ..... 13

2.3 Kiibil paiknevad välisseadmed ..... 15

2.4 TMS320VC5510 DSK ..... 16

2.5 AIC23 koodek ..... 17

3. PRAKTILISED ÜLESANDED DIGITAALSE SIGNAALITÖÖTLUSE MEETODITEGA TUTVUMISEKS ..... 18

3.1 Signaalijagur ..... 18

3.2 Praktikumitöö katsestend ja selle ühendamine ..... 19

3.3 Signaaligeneraator ..... 20

3.4 Praktiline töö: Diskreetimisteoreemi praktiline kontrollimine ..... 20

3.5 Praktiline töö: Filtri tüübi ja parameetrite eksperimentaalne määramine ..... 21

3.6 Praktiline töö: FIR filtri loomine ..... 24

KOKKUVÕTE ..... 26

ABSTRACT ..... 27

LISAD ..... 30

Lisa 1. TMS320VC5510 DSP funktsionaalsuse plokkskeem ..... 30
Lisa 2. TMS320VC5510 DSK arendusplaat ..... 31
Lisa 3. Signaalijagur ..... 32
Lisa 4. Signaaligeneraator ..... 33
Lisa 5. Katsestendi kooste foto ..... 34
Lisa 7. Esimese praktikumi juhend ..... 35
Lisa 8. Teise praktikumi juhend ..... 38
Lisa 9. Kolmanda praktikumi juhend ..... 41
Lisa 10. Praktikumijuhendite ja tarkvara DVD ..... 44

# SISSEJUHATUS

Digitaalne signaalitöötlus (LOFY.03.010) koos kaasnevate praktiliste töödega on olnud arvutitehnika eriala kohustuslikus aineks juba neli aastat. Õppeaine eesmärgiks on anda ülevaade digitaalse signaalitöötluse põhitõdedest ja -printsiipidest reaalsete süsteemidega töötamisest tuleneva praktilise kogemuse kaudu. Kahjuks puuduvad jõukohased ja detailsete juhenditega praktikumitööd. Seetõttu jääb aine olemuselt teoreetiliseks ja ei ole võib-olla nii kergesti omandatav.

Käesoleva bakalaureusetöö eesmärgiks on koostada praktikumitööd ning -juhendid, mis põhinevad TMS320VC5510 digitaalsel signaalitöötlusprotsessoril ja võimaldaks tudengitel tutvuda digitaalse signaalitöötlusega inimkõrvaga kuuldavas sagedusvahemikus. Praktilisi ülesandeid peab olema võimeline lahendama ilma eelteadmiseta protsessorite programmeerimisest. Protsessorite programmeerimise õppimiseks on Tartu Ülikoolis olemas eraldi kursused. Töö eesmärgi saavutamiseks tuleb programmeerida vajalik tarkvara ja näitefailid ning luua katsestend, millel oleks võimalik praktilisi töid teha. Kokku peaks moodustuma terviklik lahendus.

Bakalaureusetöö algab töös kasutatud lühendite vastete ja lühikese seletusega. Esimeses peatükis tutvustatakse digitaalse signaalitöötluse ja selleks kasutatava riistvara arengut. Töö teises osas tutvustatakse TI TMS320VC5510 digitaalset signaalitöötlusprotsessorit (DSP), selle arhitektuuri, võimalusi ja omadusi. Samuti kirjeldatakse lühidalt konkreetset arendusplaati, mida praktikumi kasutatakse ja millel paikneb TI TMS320VC5510 DSP. Töö viimases osas kirjeldatakse loodud vahendeid ning katsestendi ja pakutakse välja konkreetseid praktilisi töid. Detailsed juhendid ja näitefailid on paigutatud lisadesse, et vältida põhiteksti koormamist ja pikaks venimist. Lisades olev DVD sisaldab vajalikku tarkvara, juhendeid ja näitefaile.

# KASUTATUD LÜHENDID JA DEFINITSIOONID

ADC (ing *analog-to-digital converter*) – analoog-digitaalmuundur. Seade, mis muundab analoogsignaali digitaalsignaaliks.

ALU (ing *arithmetic and logic unit*) – aritmeetika-loogika üksus. Digitaallülitus, mis teostab aritmeetika ja loogika operatsioone.

ARM (ing *advanced RISC machine*) – arenenud RISC masin on 32-bitine vähendatud käsustikuga arvutiarhitektuur (RISC), mida arendab ARM Holdings. ARM protsessorite lihtsus on teinud selle sobivaks madala voolutarbega rakenduste jaoks nagu mobiiltelefonid ja muud väikesed seadmed.

CPLD (ing *complex programmable logic device*) – keerukas programmeeritav loogikaseade. Koosneb makropesikutest, mis teostavad loogikatehteid.

DAC (ing *digital-to-analog converter*) – digitaal-analoogmuundur. Seade, mis muundab digitaalse signaali pidevaks signaaliks.

DFT (ing *discrete Fourier transform*) – diskreetne Fourier' teisendus. Integraalne teisendus diskreetsete signaalide jaoks, mida tehes võime leida mistahes signaali spektri.

DIP (ing *dual in-line package*) – kahe viigureaga pakend. Kasutatakse enamasti mikroskeemide puhul, aga ka näiteks lülitite ja elektromehaaniliste releede jaoks.

DMA (ing *direct memory access*) – mälu otsepöördumine. Võimaldab mälu poole pöörduda kesksest protsessorist sõltumatult.

DPLL (ing *digital phase-locked loop*) – faasilukustusega tagasisidestatud süsteem. Võrdleb sisendsignaali faasi enda genereeritud signaali faasiga ja kontrollib tagasisideahelaga sisemise ostsillaatori sagedust, et hoida signaale faasis.

DSK (ing *DSP starter kit*) – digitaalse signaalitöötlusprotsessori, erinevate sisendite ja väljunditega ning lisakiipidega arendusplaat. Mõeldud katsetamiseks ja arendustööks.

DSP (ing *digital signal processor)* – digitaalne signaalitöötlusprotsessor. Mikroprotsessor, mille arhitektuur on optimeeritud digitaalse signaalitöötluse eripärade jaoks.

EHPI (ing *enhanced host-port interface*) – täiustatud peremees-pordi liides. Paralleelne port, mille kaudu saab peremees-protsessor pöörduda otse DSP mälu poole.

EMIF (ing *external memory interface*) – väline mäluliides. Liidese külge saab ühendada lisamälu.

FFT (ing *fast Fourier transform*) – kiire Fourier' teisendus. Põhineb teisenduse teostamiseks vajalike arvutuste mahu vähendamises teatud (korduvate) tulemuste (korduva) ärakasutamise teel või mõne spetsiifilise vaheteisenduse kasutamise teel

FIR (ing *finite impulse response*) filter – lõpliku impulsskostega filter on signaalitöötluses kasutatav filter, mille väljundsignaal on arvutatav sidumina sisendsignaalist ja filtri impulsskostest.

GPIO (ing *general purpose input/output*) – üldotstarbeline sisend/väljund on viik, mida saab tarkvaralisel määrata sisendiks või väljundiks.

IIR (ing *infinite impulse response*) filter – lõpmatu impulsskostega filter ehk rekursiivne filter. Rekursiivsete filtrite impulsskoste koosneb eksponentsiaalselt kahanevate amplituudidega sinusoididest, mis teeb nende impulsskoste põhimõtteliselt lõpmatult pikaks.

JTAG (ing *joint test action group)* – IEEE 1149.1 standardile vastav liides. Algselt loodud tootmisel seadmete kontrollimiseks, kuid sobib ka programmide silumiseks ja tarkvara laadimiseks.

MAC (ing *multiply–accumulate*) – korruta-akumuleeri tehe, ka vastav register, kus hoitakse vahearvutusi. MAC tehtega liidetakse kahe arvu korrutis akumulaatoris oleva arvuga.

McBSP (ing *multi-channel buffered serial port*) – mitme kanaliga puhverdatud jadaport. Need on kahesuunalised ja võimaldavad otseliidest teiste seadmetega, nagu näiteks teised DSPd ja koodekid.

MMACS (ing *millions of multiply-accumulates per second*) – miljon MAC tehet sekundis. Seda ühikut kasutatakse tihti protsessori jõudluse mõõtmiseks.

MMR (ing *memory mapped registers)* – on registrid, mille aadressid on mälus kindlal kohal. MMR registris hoitakse viidet välisele seadmele ja tänu sellele on võimalik pöörduda välise seadme mälu poole samade käskudega, millega pöördutakse sisemise mälu poole.

SBSRAM (ing *synchronous burst SRAM*) – sünkroonne impulss SRAM. SRAM, millel on võimalik lühiajaliselt kirjutamiskiirust suurendada.

SDRAM (ing *synchronous DRAM*) – sünkroonne dünaamiline muutmälu. Mälu, mis ootab taktsignaali ära enne, kui vastab juhtsisenditele. Pärast voolu eemaldamist andmed hävivad.

SIMD (ing *single instruction, multiple data*) – üks käsk, erinevad andmed. SIMD käsud võimaldavad sama tehet sooritada erinevate andmetega korraga.

SRAM (ing *static random access memory*) – staatiline juhupöördusega mälu. Andmed säilivad ka pärast voolu eemaldamist.

VLIW (ing *very long instruction word*) – väga pikk käsu sõna. Üks käsk sisaldab mitu eraldiseisvat käsku.

# 1. ÜLEVAADE DIGITAALSE SIGNAALITÖÖTLUSE ARENGUST JA RIISTVARAST

## 1.1 Digitaalse signaalitöötluse teke ja areng

Alates Teisest maailmasõjast, kui mitte varem, on teadlased kaalunud võimalust rakendada digitaalseid lahendusi signaalitöötlusel. Näiteks 1940ndate lõpus arutlesid C. Shannon, H. Bode ja teised teadlased firmast Bell Telephone Laboratories võimaluse üle kasutada digitaalsete elektriahelate elemente filtreerimise eesmärgil. Kuid sel ajal polnud sobivat riistvara saadaval, sest hind, suurus ja vastupidavus soosis tavalisi analooglahendusi. [1]

1950ndate keskel arutles Massachusettsi Tehnoloogiainstituudi professor J. Linvill magistriseminarides digitaalse filtreerimise teemal. Selleks ajaks oli kontrolliteooria, osaliselt W. Hurewicz'i töödel põhinev, kehtestatud eraldiseisva teadusharuna ning diskreetimine ja selle mõju spektrile hästi mõistetud. Elektroonikainsenerid hakkasid kasutama mitmeid matemaatilisi meetodeid nagu näiteks z-teisendus. Kuid tol ajal suutis tehnoloogia töödelda ja kontrollida vaid madalsageduslikke signaale. Kuigi seismoloogid kasutasid probleemide lahendamiseks digitaalse filtri kontseptsioone, ilmus rangem digitaalse signaalitöötluse teooria alles 1960ndate keskel. Sel perioodil tegi ränipõhiste integraalskeemide tehnoloogia tulek võimalikuks terviklikud digitaalsed süsteemid, kuigi need olid kallid. [1]

Esimese suure panuse digitaalsete filtrite arendamisse andis J. Kaiser. Tema uurimistööd näitasid, kuidas luua kasulikke filtreid, kasutades bilineaarset teisendust. 1965. aastal avaldati J. W. Cooley ja J. Turkey tuntud teadustöö, mis käsitles kiiret Fourier' teisendust (FFT): viisi, kuidas efektiivselt ja kiiresti teha diskreetset Fourier' teisendust (DFT). Sel ajal arendati välja riistvara, mis sobis paremini digitaalsete filtrite rakendamiseks. Samuti ilmusid müüki taskukohased integraalskeemid. [1]

Digitaalse signaalitöötluse jaoks tehti teed rajavaid pingutusi neljas valdkonnas. Radar ja sonar, kus riiklik julgeolek on ohus; naftamaardlate otsimine, kus võimalik teenida suuri summasid; kosmoseuuringud, kus andmed on asendamatud ja meditsiiniline pildindus, kus on võimalik inimeste elusid päästa. Personaalarvutite revolutsioon 1980. ja 1990. aastate

alguses põhjustas digitaalse signaalitöötluse rakenduste kasvu. Arengut hakkas juhtima sõjaväe ja riigi vajaduste asemel kaubanduslik eesmärk. Paljud firmad nägid kiiresti arenevas valdkonnas võimalust teenida ja DSPd jõudsid inimesteni mobiiltelefonides, CD-mängijates ja elektroonilises kõnepostis. [2]

1970ndate alguses, enne eraldiseisvate DSP mikroskeemide ilmumist, kasutati digitaalseks signaalitöötluseks *bit-slice* tehnoloogiat: protsessorid konstrueeriti väiksema siini laiusega moodulitest, igaüks neist töötles vaid osa operandist. Sellised protsessorid koosnesid 1,2,4 või 8-bititistest aritmeetika-loogika seadmetest ja juhtliinidest. Mikrosekventser täitis programmi ning andis andme- ja juhtsignaali. [3] Antud tehnoloogia mooduleid oli Intel 3000, AMD Am2009 ja National Instruments IMP-16/IMP-8 seerias [4]. TRW poolt toodetud TDC1008 ja TDC1010 korrutamise kiipidel oli lisaks akumulaator ja seega oli võimalik sooritada MAC tehteid [5].

1978. aastal esitles Texas Instruments mänguasja Speak&Spell, mille keskmes oli maailma esimeseks DSPks peetav kiip TMS5100. Kiibi ülesandeks oli sünteesida kõne [6]. Sama aasta tutvustas American Microsystems Inc esimest spetsiaalselt digitaalse signaalitöötluse jaoks loodud kiipi S2811, mille ülesanne oli vähendada protsessori poolt tehtavaid matemaatilisi arvutusi. Kiip ei olnud eraldiseisev, kuna vajas initsieerimiseks ja konfiguratsiooniks protsessorit. Kättesaadavaks sai see alles 1979. aastal ja see ei leidnud laialdast kasutamist. [7]

Järgneval aastal esitles Intel „analoogsignaali protsessorit“ 2920, mille kiibi sees oli digitaalne protsessor ning analoog-digitaalmuundur (ADC) ja digitaal-analoogmuundur (DAC). Protsessoril puudus korrutamise moodul ja adresseerimine oli piiratud otsese adresseerimisega st programm ei saanud hargneda. Kuigi protsessor suutis mõningaid arvutusi kiiremini teha kui universaalsed mikroprotsessorid, jäi paindlikkusest siiski puudu. [8]

1980. aastal esitleti esimesi eraldiseisvaid ja tervikliku funktsionaalsusega DSPsid: NEC μPD7720 ja AT&T DSP1. Mõlemad protsessorid sisaldasid kõiki funktsionaalseid elemente, mis leiduvad ka tänapäeva DSPs: MAC üksust, adresseerimis- ja juhtmoodulit,

juht- ning andmemälu. [7] Kuna AT&T protsessori kasutus oli piiratud vaid tootja enda seadmetega, siis saavutas suuremat kommertsedu μPD7720 [9].

Texas Instruments esitles 1983. aastal oma esimest DSPd: 16-bitine TMS32010, mida saatis suur edu. See põhines Harvardi arhitektuuril, seega oli programmi ja andmete mälu ning siin eraldatud. Oli ka eraldi käskude kogum, mis võimaldas ühe käsuga kahte operatsiooni teostada. Protsessor suutis töötada 16-bitiste arvudega ja MAC tehte sooritamiseks kulus 390 ns. [5]

Ligi viis aastat hiljem ilmus teine generatsioon DSPsid. Need sisaldasid eraldi mälu kahe operandi saamaaegseks salvestamiseks ja riistvara, mis aitas iteratsioone kiirendada. Lisaks oli adresseerimismoodul võimeline ring-adresseerima. Mõned neist töötasid 24-bitiste muutujatega ja tüüpiline mudel vajas vaid 21 ns MAC tehte sooritamiseks. [5] Selle generatsiooni hulka kuuluvad näiteks 16-bitine AT&T DSP16A ja 24-bitine Motorola DSP56001. DSP56000 seeria kiipide peamine eesmärk oli heli töötlemine: 24-bitti pakkus vajaliku dünaamilise ulatuse (144 dB) ja täpsuse [10].

Kolmanda põlvkonna peamine edusamm oli rakendusspetsiifiliste üksuste ja käskude ilmumine. Need moodulid võimaldasid väga keeruliste matemaatiliste ülesannete, nagu Fourier' teisenduse või maatriksitega tehete, riistvaralist kiirendamist. Mõnedel kiipidel oli rohkem kui üks protsessori tuum, näiteks Motorola MC68356, mis võimaldas paralleelset arvutust [5]. Sel ajal ilmusid ka Texas Instruments TMS 320C80 ja TMS320C541 [5], mida kasutati 1990ndate lõpus paljudes Nokia ja Ericssoni mobiiltelefonides [11].

Neljandat generatsiooni iseloomustab kõige paremini muutused käsustikus ja käskude kodeerimises/dekodeerimises. Lisandusid VLIW ja SIMD käsulaiendused, mis suurendasid süsteemi paralleelsust. Ka taktkiirused suurenesid ja MAC tehte sooritamine võttis aega vaid 3 ns. Paralleelsust ja jõudlust suurendas veelgi superskalaarne protsessori arhitektuur, mis võimaldas takti jooksul täita mitu käsku. [5]

### 1.2 Moodne digitaalse signaalitöötluse riistvara

Tänapäeval pakutakse erinevat tüüpi DSPsid, suur osa neist on mõeldud mingit spetsiifilist ülesannet hästi lahendama. Seega varieeruvad DSPde jõudlus, mälu maht, voolutarve ja

hind. Levinud on ka kiibid, kus DSP kiip on kaasprotsessoriks. Näiteks TI OMAP seeria ühendab endas ARM arhitektuuriga protsessorit ja TI TM320 seeria DSPd. OMAP seeria leiab laialdast kasutust tahvelarvutites ja mobiiltelefonides [12].

Enamik müügil olevatest DSPdest on püsikomaprotsessorid, sest nende ehituse lihtsus annab kiiruse- ja hinnaeelise võrreldes ujukomaprotsessoritega ning dünaamiline ulatus on piisav, et töödelda pärismaailma signaale [5]. Hea näide püsikomaprotsessorite kasutusvaldkonna kohta on video: sekundi jooksul tuleb töödelda mitu megabitti andmeid.

Kui on tarvis rakendada keerukamaid algoritme või suuremat täpsust ning dünaamilist ulatust, siis on mõistlik kasutada ujukomaprotsessoreid. Programmeerimise lihtsuse tõttu kasutatakse neid prototüüpide loomisel ja väikeste tootmismahtudega rakenduste puhul, kus aeg ja tarkvara arenduse maksumus on tähtsamad, kui ühiku tootmishind. [13]

| **Tootja** | **Perekond** | **Protessori tüüp** | **Andmesiini Laius** | **Taktkiirus (kiireim)** | **BDTI mark2000™** | **BDTI Mem-Mark™** | **Mälu** |
|---|---|---|---|---|---|---|---|
| Texas Instruments | TMS320C55x | püsikoma | 16 bitti | 300 MHz | 1460 | 75 | 80-376 KB |
| | TMS320C66x | uju-ja püsikoma | 16/32 bitti | 1,5 GHz | 20030* | 62 | 5-8 MB |
| Analog devices | ADSP-213xx (SHARC) | uju-ja püsikoma | 32/40 bitti | 400 MHz | 2050 | 34 | 384 - 1024 KB |
| | ADSP-TS20x (TigerSHARC) | uju-ja püsikoma | 8/16/32/40 bitti | 600 MHz | 6400 | 52 | 512 KB - 3 MB |
| Freescale | MSC81xx (SC140 tuum) | püsikoma | 16 bitti | 500 MHz | 5610* | 67 | 1440 KB |
| | MSC815x/825x (SC3850 tuum) | püsikoma | 16 bitti | 1 GHz | 15420* | 67 | 1728-4608 KB |
| CEVA | CEVA-TeakLite III | püsikoma | 16/32 bitti | 335 MHz | 2140 | 69 | 4 GB |
| | CEVA-X1620 | püsikoma | 8/16 bitti | 330 MHz | 2660 | 67 | 4 GB |
| VeriSilicon | ZSP400 | püsikoma | 16/32 bitti | 165 MHz | 780 | 74 | 256 KB |
| | ZSP500 | püsikoma | 16/32 bitti | 205 MHz | 1620 | 68 | 64 MB |

* Tulemus on protsessori ühe tuuma kohta (toote perekonnas esineb mitmetuumalisi mudeleid)

**Tabel 1.1 Erinevate tootjate DSPde parameetrid [14].**

Tabelis 1.1 on näha valikut praegu turul pakutavatest DSPdest ja nende võrdlusi. BDTImark2000™ ja BDTI Memtest™ on mõõdupuud, mis põhinevad BDTI DSP Kernel Benchmarks™ jõudlustestidel. Antud jõudlustest koosneb 12 erineva digitaalse signaalitöötluse algoritmi (näiteks IIR filter ja FFT) testimisest. BDTImark2000™ annab

võimaluse võrrelda protsessori kiirust paremini kui lihtsamad mõõdupuud, näiteks MMAC. BDTI MemMark™ näitab protsessori mälukasutuse efektiivsust signaalitöötlusel. Mõlema mõõdupuu puhul on suurem tulemus parem (kiirem protsessor ja efektiivsem mälukasutus). [15]

# 2. TI TMS320VC5510 PROTSESSORI ÜLEVAADE

## 2.1 TMS320VC55x DSP

Texas Instrumentsi (TI) poolt toodetud TMS320VC5510 DSP on püsikomaprotsessor, mis põhineb TMS320C55x DSP generatsiooni CPU tuumal ja töötab taktsagedusel 160 MHz või 200 MHz. C55x DSP arhitektuuri puhul (DSP funktsionaalsuse plokkskeem on välja toodud lisas 1) on tähelepanu pööratud suurele jõudlusele, püüdes säilitada madalat voolutarvet. Protsessori sisemine siinide struktuur koosneb ühest programmisiinist, kolmest andmete lugemise siinist, kahest andmete kirjutamise siinist ning lisasiinidest, mis on mõeldud DMA ja välisseadmete jaoks. Need siinid võimaldavad ühe protsessori tsükli jooksul teha kolm andmete lugemist ja kaks kirjutamist. Lisaks suudab DMA kontroller CPU aktiivsusest sõltumatult teha paralleelselt kuni kaks andmete teisaldust tsükli jooksul. [16]

C55x CPU sisaldab endas kahte MAC üksust, mis on võimelised sooritama ühe tsükli jooksul 17 x 17 bitti korrutist. Kesksele 40-bitilisele aritmeetika-loogika üksusele (ALU) on lisaks 16-bitine ALU. ALU kasutus on käsustiku kontrolli all ja see annab võimaluse optimeerida paralleelset arvutust ja võimsustarvet. Neid ressursse hallatakse aadressi- ja andmeüksuses (AU ja DU lisas 1). [16]

C55x DSP generatsioon toetab varieeruva laiusega käske, mis annab parema koodi tiheduse. Lisaks on protsessoril 7-järguline käsukonveier, mis on kaitstud (käskude vaheliste konfliktide vältimiseks lisatakse automaatselt tsükleid). C5510 sisaldab käskude vahemälu, et vähendada välise mälu poole pöördumist, suurenda andmete läbilaskevõimet ja hoida süsteemi arvutusvõimsust. [16]

## 2.2 TMS320VC55x protsessori mälu

C55X arhitektuuri puhul on ühtne mäluvahemik programmikoodi ja andmete jaoks. Ajaloolistel ja tehnilistel põhjustel on programmikood adresseeritav 8 bitiga ja andmed 16 bitiga. [17] C55x protsessor suudab adresseerida 24 bitti mälu, mis teeb kokku 16 MB. Kiibil oleva mälu kogumaht on 352 KB (176 16-bitilist sõna). Väline aadressivahemik on

jaotatud EMIF poolt neljaks võrdseks CE (ing *chip enable*) plokiks. [16] Tabelis 2.1 on näha C55x perekonna ja 5510 DSK mälu jaotus.

<table>
<tr><th>Aadress<br>(sõna)</th><th>C55x perekonna<br>mälujaotus</th><th>5510 DSK</th></tr>
<tr><td>0x000000h</td><td>MMR</td><td>MMR</td></tr>
<tr><td>0x000030h</td><td>Sisemine mälu (DARAM)</td><td rowspan="2">Sisemine mälu</td></tr>
<tr><td>0x010000h</td><td>Sisemine mälu (SARAM)</td></tr>
<tr><td>0x028000h</td><td>Väline CE0</td><td>SDRAM</td></tr>
<tr><td rowspan="2">0x200000h</td><td rowspan="2">Väline CE1</td><td>Välkmälu</td></tr>
<tr><td>CPLD</td></tr>
<tr><td>0x400000h</td><td>Väline CE2</td><td rowspan="2">Lisakaart</td></tr>
<tr><td>0x600000h</td><td>Väline CE3</td></tr>
</table>

**Tabel 2.1 Üldine mälu paigutus C55x perekonna protsessoritel ja 5510 DSK-l [16, 17].**

Kiibil olevate väliste seadmete jaoks on mälu alguses MMR. Järgneb kaheksa 8 KBst plokki DARAMi ja 32 8 KBst plokki SARAMi. Ühes DARAM plokis on takti jooksul võimalik teha paralleelselt kaks operatsiooni, SARAM plokis aga üks. [17]

5510 DSK-l on esimese välise mälu ploki CE0 vahemikus SDRAM. CE1 ploki aadressivahemik on jagatud välkmälu ja CPLD vahel. Kaks viimast CE plokki on mõeldud tütarkaardi poole pöördumiseks.

ROM koosneb ühest 32KBsest plokist. ROM paikneb CE3 plokis aadressivahemikus 7FC000h−800000h kui ST3 staatuse registris paiknev bitt MPNMC = 0 algseadistuse ajal. Kui MPNMC = 1 algseadistuse ajal, siis on ROM välja lülitatud ja ei ole mälupaigutuses. Siis viitavad aadressid 7FC000h−800000h välisele mäluruumile. [16]

ROM sisaldab endas alglaadurit ja vektori tabelit ning teatmetabelit 256 siinusfunktsiooni väärtusega Q15 formaadis. Q15 on püsikoma numbriformaat, kus on üks märgibitt ja 15

bitti suuruse jaoks. Q15 on vahemikus -1 kuni 1- $2^{15}$. Vahemik on jagatud $2^{16}$ intervalliks, iga üks suurusega $2^{15}$. Ülejäänud ROMi komponente kasutatakse tehases testimise käigus. [18]

### 2.3 Kiibil paiknevad välisseadmed

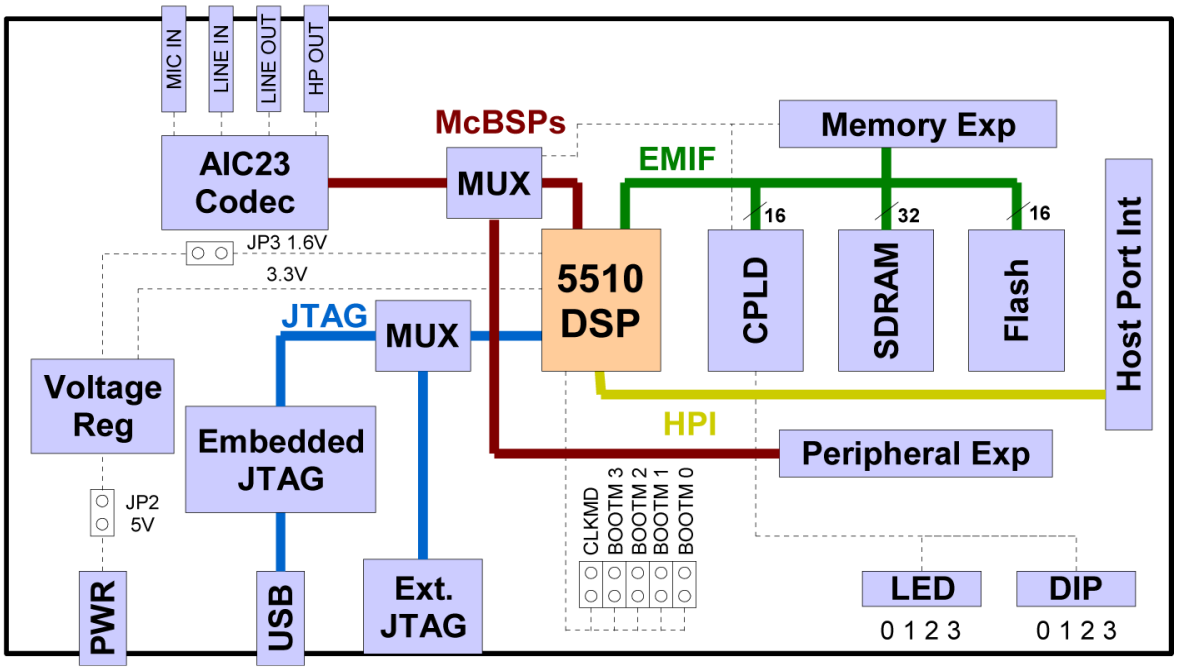
Välisseadmete kontrollregistrid kaardistatakse eraldi sisend-väljud vahemikku, mis on eraldi põhimälu vahemikust. Välisseadmete kaardistamine on seadmest sõltuv.

C5510 toetab järgnevaid välisseadmeid [19]:

- Välise mälu liides (EMIF), mis toetab järgnevaid mälutüüpe: asünkroonne SRAM, ROM, välkmälu ja sünkroonne impulss SRAM (SBSRAM) ning sünkroonne DRAM (SDRAM)
- 6-kanaliline DMA kontroller, mis võimaldab CPU sekkumiseta teha kuut eri operatsiooni mälus.
- 16-bitiline paralleelne EHPI, mille kaudu saab peremees (ing *host*) protsessor (PC, mikrokontroller, teine DSP) pöörduda otse DSP mälu poole.
- DPLL taktsignaali generaator, mis annab võimaluse jagada või kordistada välist taktsignaali.
- Kaks 16-bitilist taimerit 4-bitise loenduriga, mis kokku annab 20-bitilise dünaamilise ulatuse. Taimereid on võimalik seadistada töötama CPU või välisel taktsignaalil.
- Kolm mitme kanaliga puhverdatud jadaporti (McBSP). Need on täisdupleks tüüpi ja võimaldavad otseliidest teiste seadmetega, nagu teised DSPd ja koodekid.
- 8 tarkvaraliselt seadistatavat sisend-väljundviiku (GPIO)

## 2.4 TMS320VC5510 DSK

Praktiliste ülesannete lahendamiseks kasutatakse Digital Spectrumi poolt toodetud TMS320VC5510 DSK digitaalse signaalitöötluse arendusplaati (vt lisa 2). Joonisel 2.1 on näha arendusplaadi plokkskeem. [17]




**Joonis 2.1 TMS320VC5510 DSK arendusplaadi plokkskeem [17]**

Arendusplaadil on TI TMS320C5510 DSP, mis töötab 200 MHz taktsagedusel, AIC23 stereo koodek, 8 megabaiti SDRAMi, 512 kilobaiti välkmälu, 4 kasutaja kontrollitavat valgusdioodi ja DIP lülitit, neli 3,5 mm stereo pesa: mikrofoni sisend (MIC IN), sisendliin (LINE IN), väljundliin (LINE OUT) ja kõrvaklapi väljund (HEADPHONE). Alglaadimise konfiguratsiooni valimiseks on plaadil sillused ning tütarkaartide ühendamiseks standardsed laienduspesad. Arendusplaadi konfigureerimiseks kasutatakse CPLDs paiknevaid registreid. [17]

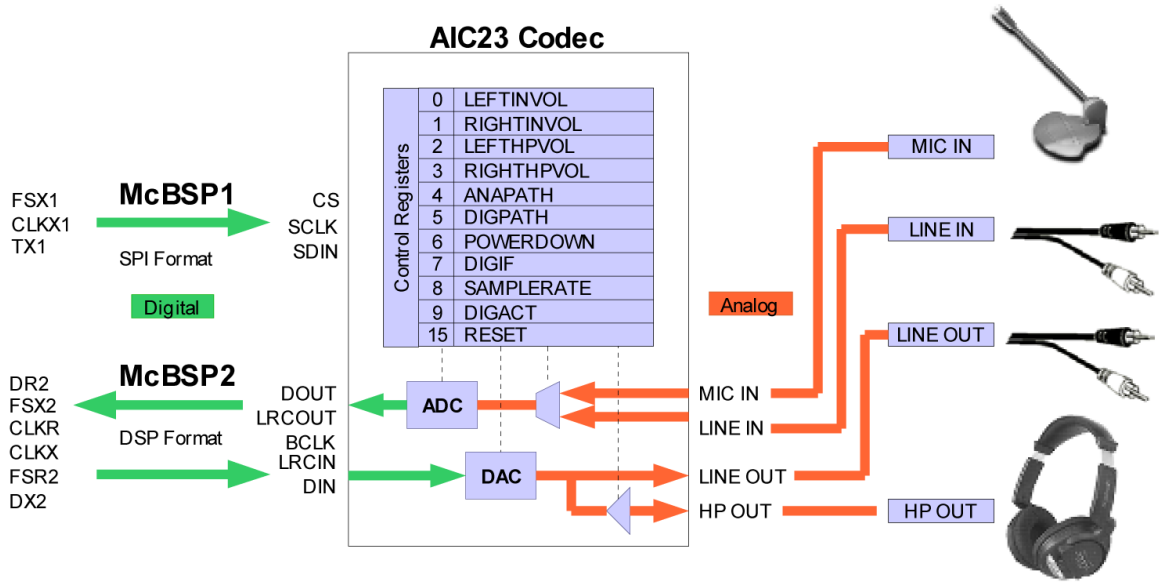
DSKga on kaasas TI tarkvara *Code Composer Studio*, mis suhtleb arendusplaadiga plaadil oleva JTAG emulaatori kaudu. Arendusplaadiga on võimalik ka suhelda kasutades välist JTAG emulaatorit. [17]

Toide (+5 V) saadakse välisest toiteadapterist. Plaadil olevad pingeregulaatorid varustavad DSP tuuma 1,6 Vga  ja ülejäänud arendusplaati 3,3 Vga. [17]

## 2.5 AIC23 koodek

Helisignaalide sisestamiseks ja väljastuseks on arendusplaadil AIC23 stereo koodek TI TLV320AIC23. Joonisel 2.2 on näha AIC23 koodeki toimimise põhimõte. Mikrofoni või sisendliini kaudu saadud pidev signaal muudetakse ADC abil digitaalkujule ja edastatakse protsessorile töötlemiseks. Kui DSP on andmed töödelnud, siis muundatakse digitaalne signaal DACiga tagasi pidevaks signaaliks ja väljastatakse väljundliini ja kõrvaklappide pessa. ADC ja DAC parameetrid määratakse AIC23 koodeki kontrollregistris.[17]

Koodek suhtleb kahe jadakanali kaudu: ühesuunaline juhtkanal McBSP1, mida kasutatakse juhtregistrite seadistamiseks ja kahesuunaline andmekanal McBSP2, kus kaudu liigub kogu heli informatsioon [17]. Vastavalt juhtregistri väärtusele liigutatakse andmeid 16-, 20-, 24- või 32-bitiste sõnadena, diskreetimissagedusel 8 – 96 kHz [20].

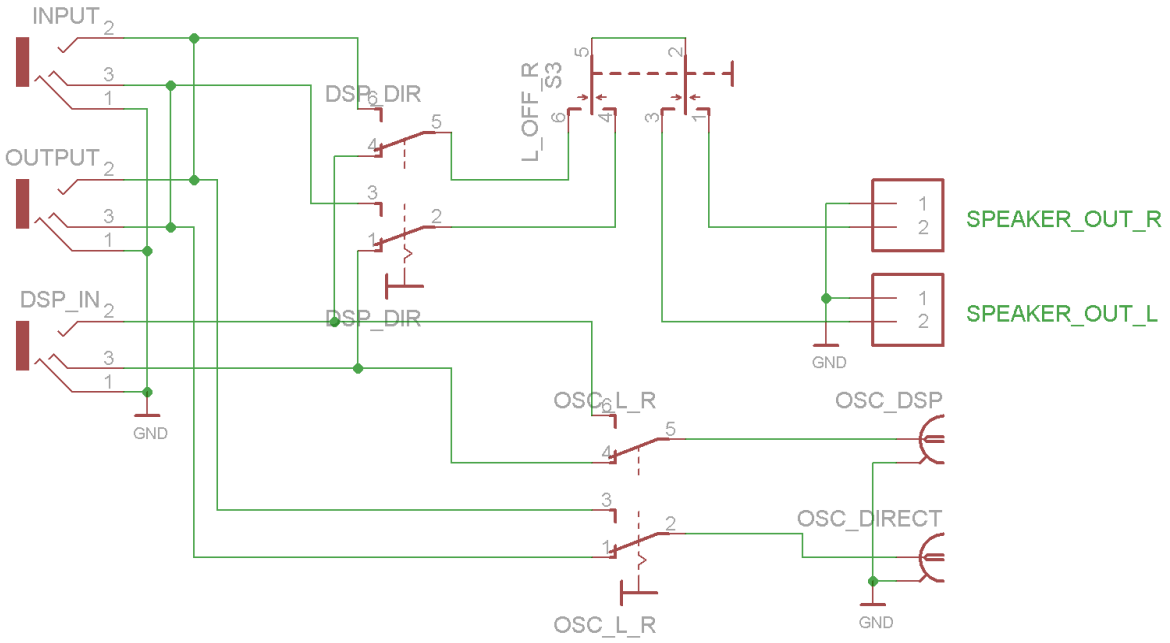



**Joonis 2.2 AIC23 koodeki plokkskeem [17]**

# 3. PRAKTILISED ÜLESANDED DIGITAALSE SIGNAALITÖÖTLUSE MEETODITEGA TUTVUMISEKS

## 3.1 Signaalijagur

Praktikumide tarbeks projekteeriti ja koostati passiivne signaalijagur, mille eesmärgiks on koondada sisendid ja väljundid ühte füüsilisse plokki. Jagur võimaldab lihtsalt ja kiiresti valida sisendit, väljundit ning kanalit (vasak, parem). Joonisel 3.1 on välja toodud jaguri skeem. Pilte koostatud signaalijagurist saab näha lisas 3.




**Joonis 3.1 Signaalijaguri elektriskeem**

Vasakul pool on kolm 3,5 mm stereo pesa: INPUT, OUTPUT ja DSP IN. INPUT pesasse tuleb signaaligeneraatorist signaal, mis suunatakse edasi DSPsse väljundi OUTPUT kaudu. DSPst väljastatav signaal tuleb sisendisse DSP IN.

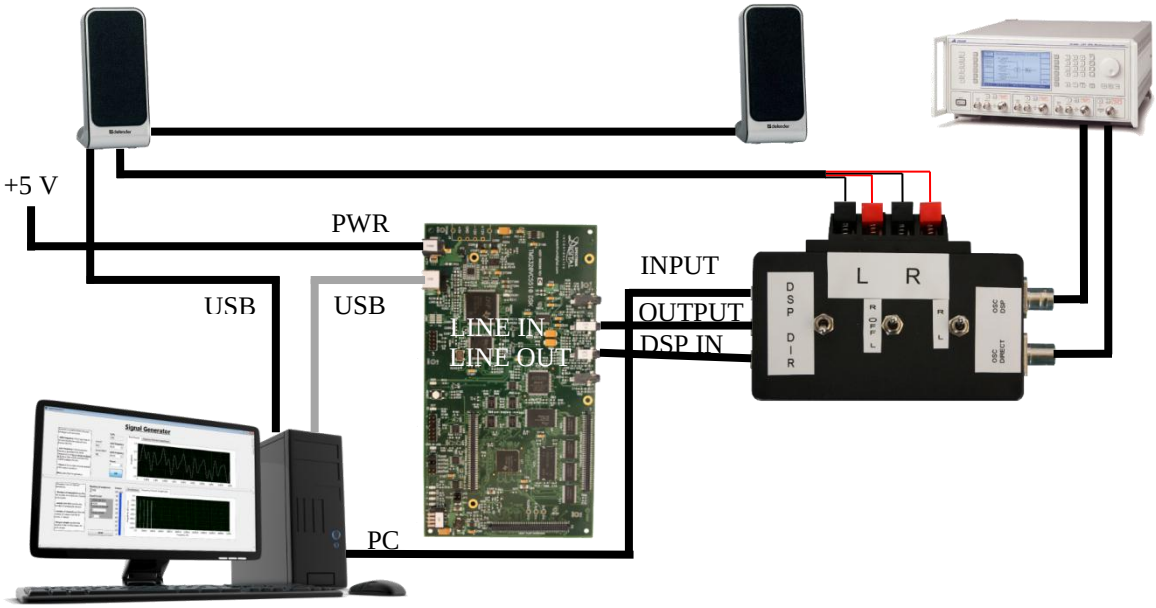
Jaguri paremal pool on kaks BNC pesa: OSC DSP ja OSC DIRECT, mille külge on saab ühenda ostsillaatori. Väljundist OSC DSP saab uurida DSPst tulnud signaali ja väljundist OSC DIRECT otse signaaligeneraatorist tulnud signaali. Lülitiga on võimalik valida parema ja vasaku kanali vahel.

Jaguri esiküljel on kaks kõlariklemmide paari (signaal ja maa), mille kaudu saadetakse signaal aktiivkõlaritesse. Lüliti abil on võimalik valida signaaligeneraatorist ja DSPst tuleva signaali vahel. Teise lülitiga on võimalik saata parema kanali signaali paremasse kõlarisse ja vasaku kanali signaali vasakusse kõlarisse või vaigistada mõlemad kõlarid.

### 3.2 Praktikumitöö katsestend ja selle ühendamine

Enne praktikumitöö juurde asumist tuleb üles seada ja ühendada vajalikud seadmed. Selleks, et vältida DSK arendusplaadil olevate komponentide kahjustamist, asetatakse see antistaatilise mati peale. Juhul kui praktikumitöö tegija ei ole antistaatilise randmepaela kaudu maandatud tuleb enne arendusplaadi puutumist ennast maandada.

Katsestend koostatakse joonise 3.2 järgi. Signaalijaguri ja arendusplaadi ning arvuti vahele ühenduvad juhtmed on märgistatud mõlemas otsas sildiga, mis näitab, kuhu pesasse tuleb need ühendada. Enne arendusplaadi toite ühendamist ja praktikumitöö juurde asumist näidatakse skeem praktikumijuhendajale ette. Katsestendi koostefoto on toodud lisas 5.




**Joonis 3.2 Katsestendi koostamise skeem**

### 3.3 Signaaligeneraator

Selleks, et signaali töödelda on esmalt tarvis sisendsignaali. Praktikumide tarbeks on vaja genereerida võrdlemisi keerulisi sisendsignaale (müra, liitsignaale). Kuna praktikumis teeb katseid korraga mitu inimest, siis on ebaotstarbekas kõigile kalleid signaaligeneraatoreid osta. Et praktikumis tegeletakse inimkõrvale kuuldava sagedusvahemikuga, siis sobib signaalide genereerimiseks arvuti helikaart.

Praktikumi tarbeks loodi LabVIEW programmeerimiskeskkonnas spetsiaalne tarkvara: *Signal Generator*. Loodud signaaligeneraator võimaldab genereerida:

- Siinustest koosnevat liitsignaali. Tuleb valida baassagedus, sageduse samm ja signaalide arv. Baassagedus ja sageduse samm peavad olema jagatise $\frac{diskreetimissagedus}{lugemite\ arv}$ täisarvkordne. Juhul kui sagedused ei vasta sellele tingimusele, kasutatakse lähimaid sagedusi, mis vastavad nõuetele.
- Siinus-, nelinurk-, hammas- või kolmnurksignaali soovitud sagedusega.
- $\frac{1}{f}$ müra valikulise müratihedusega ($\frac{V}{\sqrt{Hz}}$ sagedusel 100 Hz).

Programm liidab kasutaja valitud signaalid kokku ja saadab helikaardi väljundisse. Kasutajal on võimalus valida lugemite arvu, diskreetimissagedust, kanalite arvu, lugemi bittide arvu ja väljundi helitugevust. Vaikimisi kuvatakse signaalide käitumist ajas, kuid kasutajal on võimalik kuvada ka signaalide spektreid.

Lisas 4 on signaaligeneraatori kasutajaliidese kuvatõmmis. Tarkvara paigaldaja, mis sisaldab ka programmi käivitamiseks vajalikku keskkonda LabVIEW Run-Time Engine 2011, on lisas 10 oleval DVDl kaustas *Signaaligeneraator\Installer*.

### 3.4 Praktiline töö: Diskreetimisteoreemi praktiline kontrollimine

Esimeses praktikumis tuleb katsete teel kontrollida Nyquist–Shannoni diskreetimisteoreemi kehtivust (valem 1). Vastavalt diskreetimisteoreemile saame me võetud lugemitest algse pideva signaali õigesti taastada, kui pideva signaali ribalaiuse $B$ ja diskreetimissageduse $f_s$ vahel kehtib seos:

$$B < \frac{f_s}{2} \quad (1)$$

Seega peab diskreetimissagedus $f_s$olema üle kahe korra suurem, kui pidavas signaalis esinev kõige kõrgema sageduse komponent. [18]

Praktikumitöö algab sellega, et tudeng seab katsestendi üles ja käivitab *Code Composer Studio*, kus avatakse eelnevalt valmis tehtud projekt *Praktikum_1.pjt*, mis laetakse arendusplaadile. Projektis olev kood kasutab AIC23 koodekit (joonis 2.2) ning võtab lugemeid sisendliinist ja saadab need väljundliini. Algselt on diskreetimissageduseks valitud 48 kHz ja signaaligeneraatorist tulevad sagedused väljastatakse moonutusteta. Siis vähendatakse diskreetimissagedust 24, 16 ja siis 8 kHz-ni ning katsetatakse, mis saab sisendsignaalidega, mis ei vasta valemile 1. Detailsem juhend, mille järgi praktikumitööd tehakse, on lisas 7 ning sisaldab ka kontrollküsimusi ning viiteid teema kohta lugemiseks. Praktilise töö juures olevate küsimuste vastatakse hiljem kirjalikus vormis. Lisas 10 oleval DVD-l on fail *Praktikum_1.pjt* kaustas *DSP praktilised tööd\Praktiline töö nr 1.*

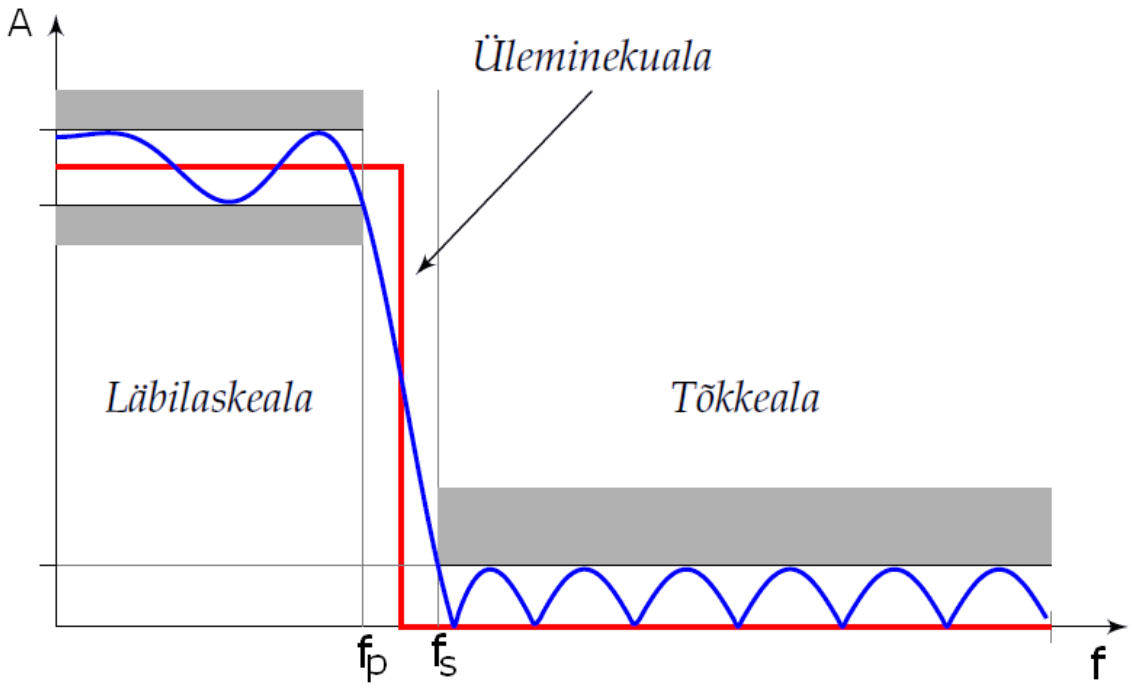
Esimene praktikumitöö on sisuliselt väikse mahuga, kuid arvestada tuleb seda, et suur osa ajast kulub katsestendi ühendamiseks ja programmeerimiskeskkonnaga *Code Composer Studio* tutvumiseks.

### 3.5 Praktiline töö: Filtri tüübi ja parameetrite eksperimentaalne määramine

Teine praktiline töö seisneb katsetega filtri tüübi (madal-, kõrg-, riba- või tõkkefiltri) ja filtri parameetrite (joonis 2) kindlakstegemisel.

Filtri parameetritest on vaja leida:

- Läbilaskeala – selles vahemikus olevaid sagedusi ei summutata (va võngetest tulnud muutused). Määratud sagedusega $f_p$.
- Tõkkeala – selles vahemikus olevad sagedused summutatakse. Määratud sagedusega $f_s$.
- Üleminekuala – vahemik läbilaskealast tõkkealaks ja vastupidi (nt. kõrgpääs filtri puhul). Vahemik $f_p$ kuni $f_s$.

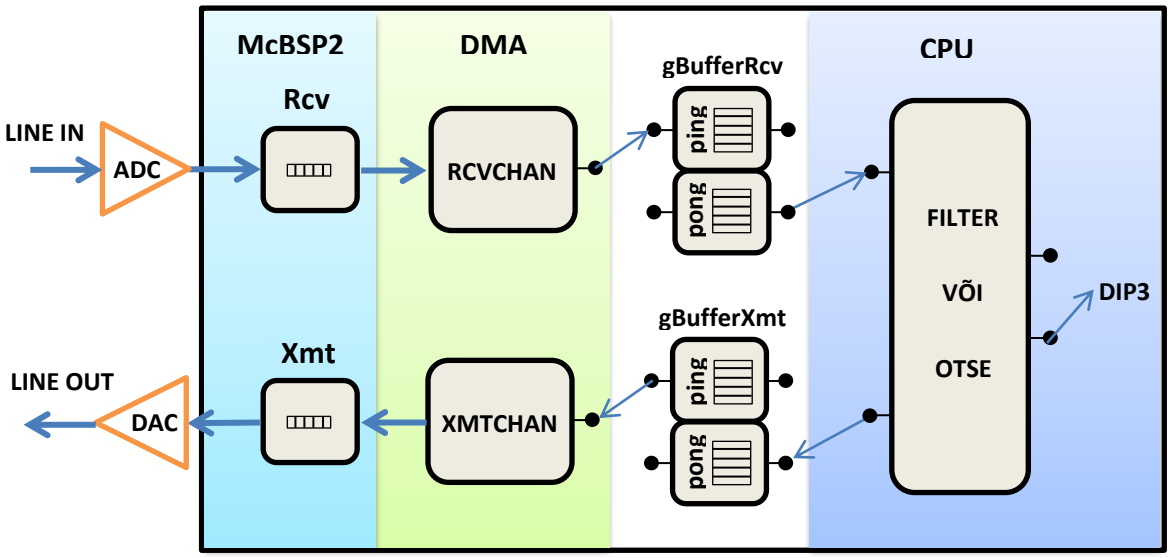



**Joonis 3.3 Filtri parameetrid madalpääs filtri näitel. Punane joon on ideaalne filter, sinine reaalne filter.**

Praktilise töö jaoks kasutatakse projekti *Praktikum_2.pjt* (lisas 10 DVD-l kaustas *DSP praktilised töö/Praktiline töö nr 2*).

Projektis olev kood kasutab AIC23 koodekit (joonis 2.2) ja lugemeid liigutatakse McBSP2 jadaliidese kaudu. DMA kanalit 0 kasutatakse koodekile andmete edastamiseks ja kanalit 1 andmete vastuvõtmiseks. Lugemite hoidmiseks kasutatakse kahte puhvrit (nimedega ping ja pong). DMA on algselt seadistatud kasutama ping puhvrit. Kui see täis saab, siis seadistatakse DMA ümber kasutama pong puhvrit. Sel ajal on võimalik ping puhvrit töödelda lugemite ülekirjutamist kartmata. Iga kord, kui puhver täis vilgutatakse LEDi: ping puhul LED3-e ja pong puhul LED2-e. Programm töötab katkestustepõhiselt ja on seetõttu väikse ressursinõudlikkusega.

Programm kontrollib enne sisendlugemite väljundisse saatmist lüliti DIP3 seisu. Kui lüliti on vajutamata, siis läbib signaal DSPd ilma töötluseta. Kui lüliti on vajutatud, siis töödeldakse lugemeid filtri koefitsientidega. Programmi tööpõhimõte on toodud joonisel 3.4.




**Joonis 3.4 Progammi tööpõhimõte**

Filtreerimiseks kasutatakse eelnevalt loodud FIR filtrit. Väljundsignaal leitakse konvulsioonisumma abil:

$$y[n] = \sum_{k=0}^{N} h[k] \cdot x[n-k], \tag{2}$$

kus $y[n]$ on väljunsignaal, $h[k]$ filtri koefitsient, $x[n-k]$ käesoleva ajahetke $n$ suhtes $k$ lugemit hilistatud sisendsignaali väärtus ja $N$ on filtri pikkus.

Praktikumitöö algab sellega, et tudeng seab katsestendi üles ja käivitab *Code Composer Studio*, kus avatakse eelnevalt valmis tehtud projekt *Praktikum_2.pjt*. Siis valitakse vastavalt juhendaja korraldusele üks 10-st filtri impulsskostest kaustast *DSP praktilised tööd\Praktiline töö nr 2\Filtrid* ning laetakse kood arendusplaadile.

| Faili nimi | Filtri tüüp | Läbilaskeala (Hz) | Tõkkeala (Hz) | Üleminekuala (Hz) |
|---|---|---|---|---|
| *filter1.h* | madalpääs | 0-5000 | 6000-… | 5000-6000 |
| *filter2.h* | madalpääs | 0-2000 | 2500-… | 2000-2500 |
| *filter3.h* | kõrgpääs | 9000-… | 0-8000 | 8000-9000 |
| *filter4.h* | kõrgpääs | 1000-… | 0-100 | 100-1000 |
| *filter5.h* | riba | 900-4800 | 0-450,6000-0 | 450-900, 4800-6000 |
| *filter6.h* | riba | 1200-1500 | 0-1000, 2000-… | 1000-1200, 1500-2000 |
| *filter7.h* | riba | 300-600 | 0-100, 900-… | 100-300, 600-900 |
| *filter8.h* | tõkke | 0-10, 900-… | 500-700 | 10-500, 700-900 |
| *filter9.h* | tõkke | 0-100, 7000-… | 1000-6000 | 100-1000, 6000-7000 |
| *filter10.h* | tõkke | 0-3000, 4500-… | 3500-4000 | 3000-3500, 4000-4500 |

**Tabel 3.1 10 Genereeritud filtrite tüübid ja parameetrid**

Filtrid on loodud MATLABi Filter Design & Analysis Tool-i kasutades. Diskreetimissageduseks valiti 48 kHz. Filtrid koosnevad 201 koefitsendist ning tõkkeala sumbumiseks on valitud 80 dB. Tabeli järgi on juhendajal võimalik kontrollida, kas tudeng sai õiged tulemused. Erinevate filtrite olemasolu tagab selle, et vastuseid ei kirjutata maha.

Järgnevad eksperimendid, kus proovitakse erinevaid (liit)signaale süsteemist läbi lasta ning võrreldakse sisend- ja väljundsignaali erinevusi kasutades kõlareid ja spektrianalüsaatoriga ostsilloskoopi.

Detailsem juhend, mille järgi praktikumitööd tehakse, on lisas 8 ning sisaldab ka kontrollküsimusi ning viiteid teema kohta lugemiseks. Praktilise töö juures olevate küsimuste vastatakse hiljem kirjalikus vormis. Lisas 10 oleval DVD-l on fail *Praktikum_2.pjt* kaustas *DSP praktilised tööd\Praktiline töö nr 2*.

### 3.6 Praktiline töö: FIR filtri loomine

Kolmandaks praktiliseks tööks on vastavalt etteantud nõuetele FIR filtri projekteerimine. Tudengil on valida kahe etteantud ülesande vahel:

- Müra eemaldamine 440 Hz sagedusega (A noot) signaalist.

- 150 Hz siinuse eemaldamine liitsignaalist (nt. mõnest laulust).

Kuna digitaalse signaalitöötluse aine seminarides arvutatakse välja filtri impulsskoste ja rakendatakse sellele aknafunktsiooni käsitsi, siis praktikumides nad seda enam tegema ei pea. Filtri impulsskoste loomiseks kasutatakse MATLABi Filter Design & Analysis Tool-i, mis pakub väga palju erinevaid võimalusi filtri disainiks.

Praktikumi eesmärk on eelnevate teadmiste rakendamine püsititatud probleemi lahendamiseks. Tudengil tuleb valida filtri tüüp, lõikesagedused, pikkus ja rakendada sobivat aknafunktsiooni, et saada soovitud väljundsignaal

Probleemi lahendamiseks kasutatakse kahte erinevat filtrit, millest ühele rakendatakse aknafunktsiooni ja teisele mitte. Filtrite amplituudkosted salvestatakse pildina ning pärast praktikumi analüüsitakse loodud filtreid. Praktikum loetakse sooritatuks, kui kuuldavat müra väljundsignaalis enam ei ole.

Praktikumitöö algab sellega, et tudeng seab katsestendi üles ja käivitab *Code Composer Studio,* kus avatakse eelnevalt valmis tehtud projekt *Praktikum_3.pjt*. Vastavalt valitud ülesandele luuakse kaks filtrit erineval meetodil ja katsetatakse, kas loodud filter lahendab probleemi.

Detailsem juhend, mille järgi praktikumitööd tehakse, on lisas 9 ning sisaldab ka kontrollküsimusi ning viiteid teema kohta lugemiseks. Praktilise töö juures olevate küsimuste vastatakse hiljem kirjalikus vormis. Lisas 10 oleval DVD-l on fail *Praktikum_3.pjt* kaustas *DSP praktilised tööd\Praktiline töö nr 3.*

# KOKKUVÕTE

Bakalaureusetöös kirjeldati digitaalse signaalitöötluse ja selleks kasutatava riistvara arengut ning toodi välja näiteid tänapäeval kasutatavatest DSPdest. Lisaks kirjeldati kasutatava DSP arhitektuuri ning arendusplaati, mida praktiliste tööde lahendamiseks kasutatakse. Erilist tähelepanu pöörati arendusplaadil olevale AIC23 koodekile, mida kasutakse helisignaalide sisestamiseks ja väljastamiseks.

Praktikumi tarbeks disainiti ja ehitati valmis passiivne signaalijagur, mille eesmärgiks on koondada sisendid ja väljundid ühte füüsilisse plokki, mis võimaldaks lihtsalt signaale ümber lülitada. Erinevate sisendsignaalide loomiseks programmeeriti LabVIEW keskkonnas signaaligeneraator Signal Generator, mille funktsionaalsus valiti spetsiaalselt praktikume silmas pidades.

Käesoleva töö raames mõeldi välja kolm praktilist tööd ja koostati neile praktikumijuhendid. Et praktikumide põhirõhk on loengus kuuldu rakendamine praktiliste probleemide lahendamiseks, siis programmeeriti ka vajalikud alusfailid, mida kasutatakse praktikumides DSP programmeerimiseks.

# ABSTRACT

Practical assignments based on digital signal processing techniques

Karl Tiirik

Digital signal processing and the methods involved are often complicated and presented in a mathematical form. Without enlightening practical examples or experiments it can be difficult to understand them.

The objective of this thesis is to put together practical assignments based on Texas Instruments TMS320VC5510 digital signal processor and create practical guide for those labs. All the assignments need the implementation of digital signal processing techniques and the frequency range is confined to the human hearing range. No prior knowledge of processor programming is needed.

A signal selector/divider was designed and built for the labs. The purpose of the designed signal divider is to group all the inputs and outputs to a single physical box that can be used for quick signal selection. To generate complex input signals for processing, a special signal generator was programmed in LabVIEW. The signal generator uses PC's sound card for signal generation.

Finally, three labs were developed and practical guides written for them. Because programming digital signal processors needs a lot of knowledge about processor architecture, all the programs used in the labs were prewritten to some extent to get past the first complicated steps in DSP programming.

VIITED


[1] Dag Stranneby, William Walker, Digital Signal Processing and Applications 2nd edition, *Elsevier/Newnes*, 2004.

[2] Steven W. Smith, The Scientist and Engineer's Guide to Digital Signal Processing [Võrgumaterjal] [Tsiteeritud 5. jaanuar 2012] http://www.dspguide.com/pdfbook.htm

[3] Bit slice, *Free Online Dictionary of Computing* [Võrgumaterjal] [Tsiteeritud 5. jaanuar 2012] http://foldoc.org/bit-slice

[4] Bit slicing, *Wikipedia, the free encyclopedia* [Võrgumaterjal] [Tsiteeritud 5. jaanuar 2012] http://en.wikipedia.org/wiki/Bit_slicing

[5] Digital signal processor, *Wikipedia, the free encyclopedia* [Võrgumaterjal] [Tsiteeritud 5. jaanuar 2012] http://en.wikipedia.org/wiki/Digital_signal_processor

[6] Milestones:Speak & Spell, the First Use of a Digital Signal Processing IC for Speech Generation, *IEEE Global History Network* [Võrgumaterjal] [Tsiteeritud 9. mai 2012] http://www.ieeeghn.org/wiki/index.php/Milestones:Speak_%26_Spell,_the_First_Use_of_a_Digital_Signal_Processing_IC_for_Speech_Generation,_1978

[7] Daniel Mlynek, Yusuf Leblebici, Digital Signal Processing Architectures [Võrgumaterjal] [Tsiteeritud 5. jaanuar 2012] http://lsmwww.epfl.ch/Education/former/2002-2003/VLSIDesign/ch12/DSParch.htm

[8] 2920 Analog Signal Processor Design Handbook, *Intel*, 1980 [PDF] [Alla laaditud 5. jaanuar 2012] http://ebookbrowse.com/1980-2920-analog-signal-processor-design-handbook-pdf-d87844534

[9] Will Strauss, Early history of the DSP chip, *EE Times Asia* [Võrgumaterjal] [Tsiteeritud 5. jaanuar 2012] http://www.eetasia.com/ART_8800310261_499489_NT_a653224d.HTM

[10] Motorola 56000, *Wikipedia, the free encyclopedia* [Võrgumaterjal] [Tsiteeritud 6. jaanuar 2012] http://en.wikipedia.org/wiki/Motorola_56000

[11] Texas Instruments TMS320, *Wikipedia, the free encyclopedia* [Võrgumaterjal] [Tsiteeritud 9. mai 2012] http://en.wikipedia.org/wiki/Texas_Instruments_TMS320

[12] OMAP, *Wikipedia, the free encyclopedia* [Võrgumaterjal] [Tsiteeritud 9. mai 2012] http://en.wikipedia.org/wiki/Texas_Instruments_OMAP

[13] Gene Frantz, Ray Simar, Comparing Fixed- and Floating-Point DSPs, *Texas Instruments* , 2004 [PDF] [Alla laaditud 10. mai 2012] http://www.ti.com/lit/wp/spry061/spry061.pdf

[14] BDTI's Pocket Guide to Processing Engines for Embedded Applications, *BDTI*, 2012 [PDF] [Alla laaditud 12. mai 2012]

[15] BDTI DSP Kernel Benchmarks™ (BDTImark2000™) Certified Results, *BDTI* [PDF] [Alla laaditud 12. mai 2012]

[16] TMS320VC5510/5510A Fixed-Point Digital Signal processors [PDF] [Alla laaditud 7. jaanuar 2012] http://www.ti.com/lit/ds/sprs076o/sprs076o.pdf

[17] TMS320VC5510 DSK Technical reference [PDF] [Alla laaditud 10.mai 2012] http://c5000.spectrumdigital.com/dsk5510/docs/dsk5510_techref.pdf

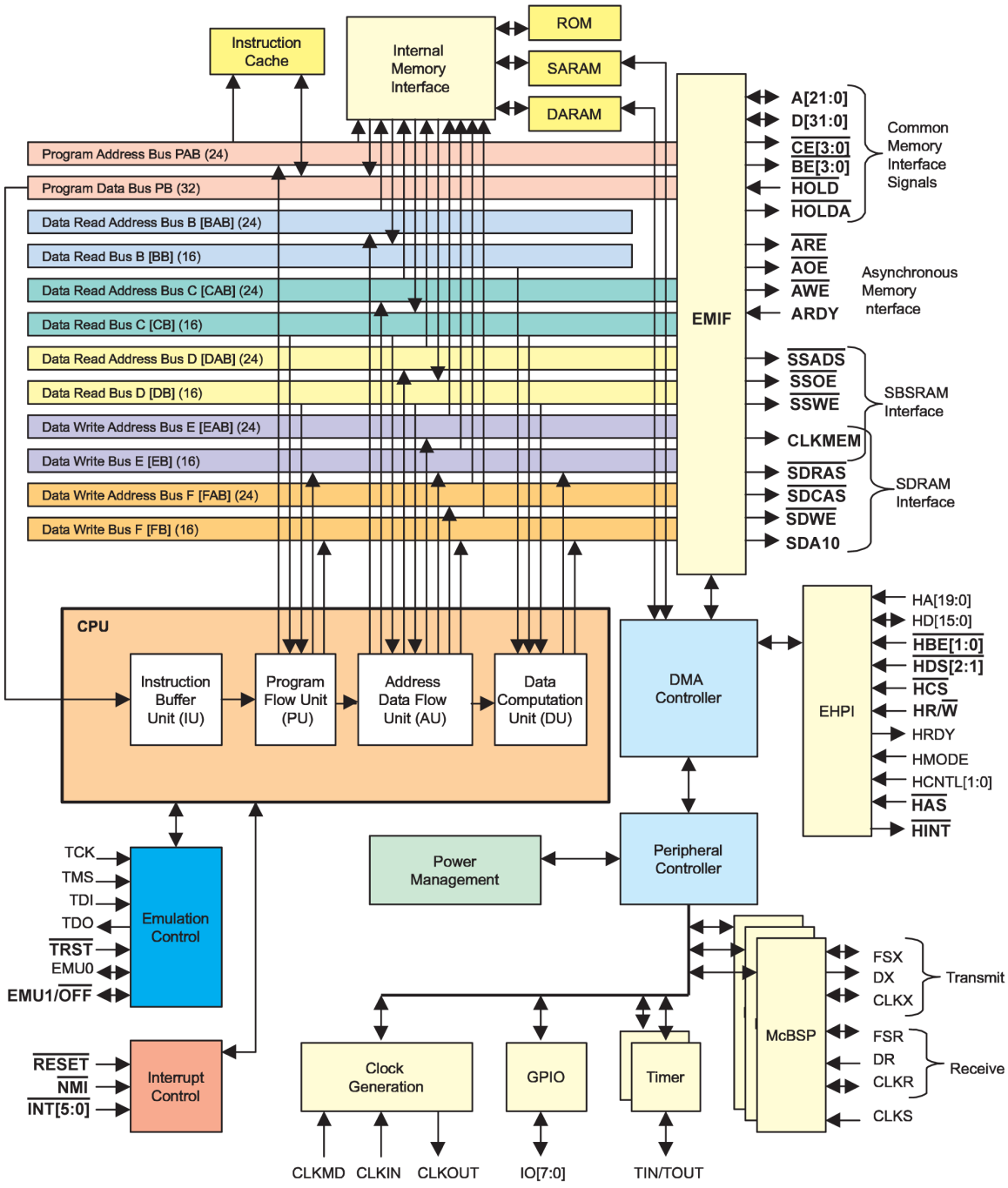
[18] Li Tan, Digital Signal Processing, *Elsevier*, 2008

[19] TMS320C55x DSP Functional Overview [PDF] [Alla laaditud 7. jaanuar 2012] http://www.ti.com/lit/ug/spru312/spru312.pdf

[20] TLV320AIC23 Data manual [PDF] [Alla laaditud 7. jaanuar 2012] http://www.ti.com/lit/ds/symlink/tlv320aic23.pdf

# LISAD

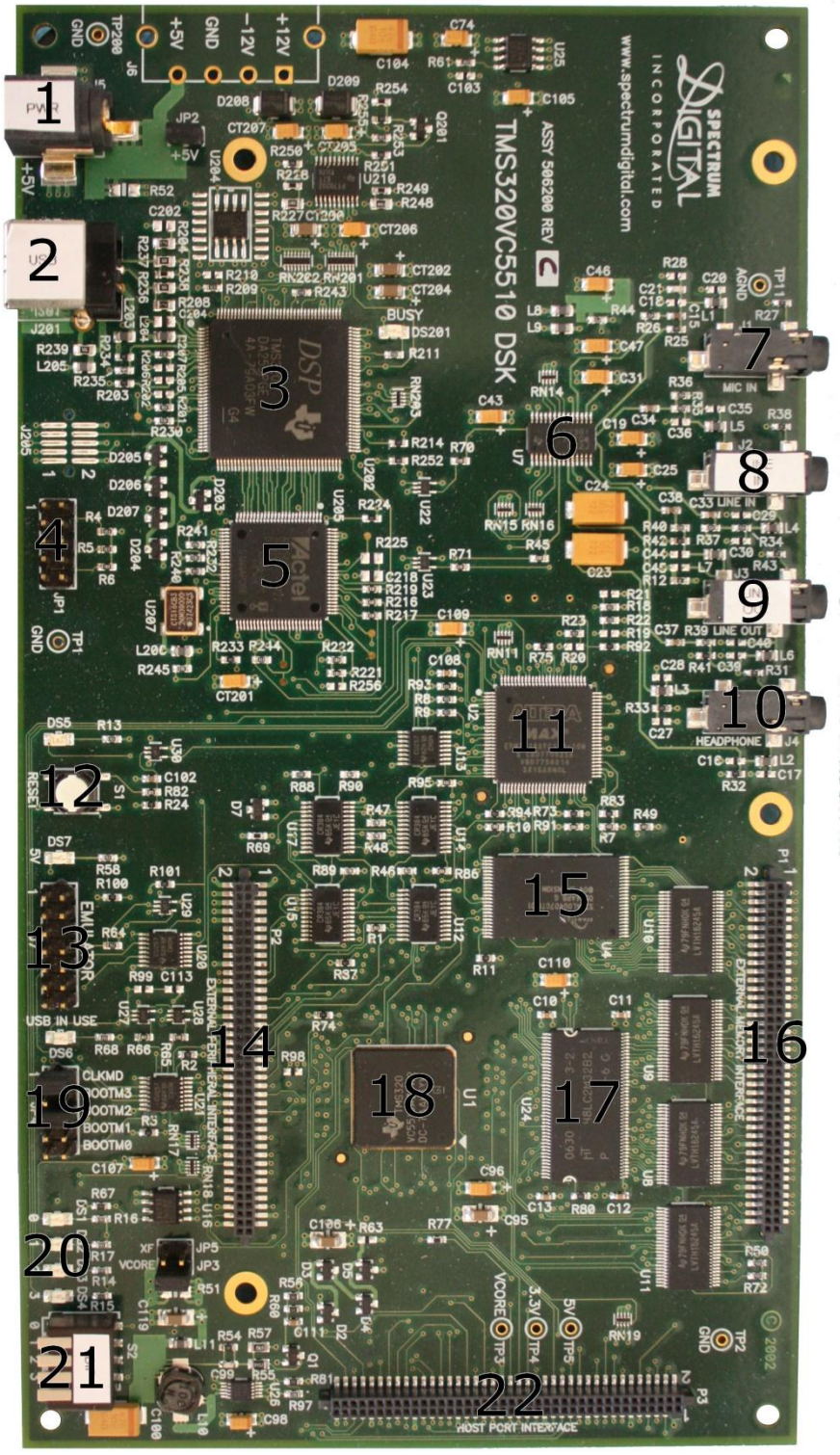
## Lisa 1. TMS320VC5510 DSP funktsionaalsuse plokkskeem




Allikas: [16]

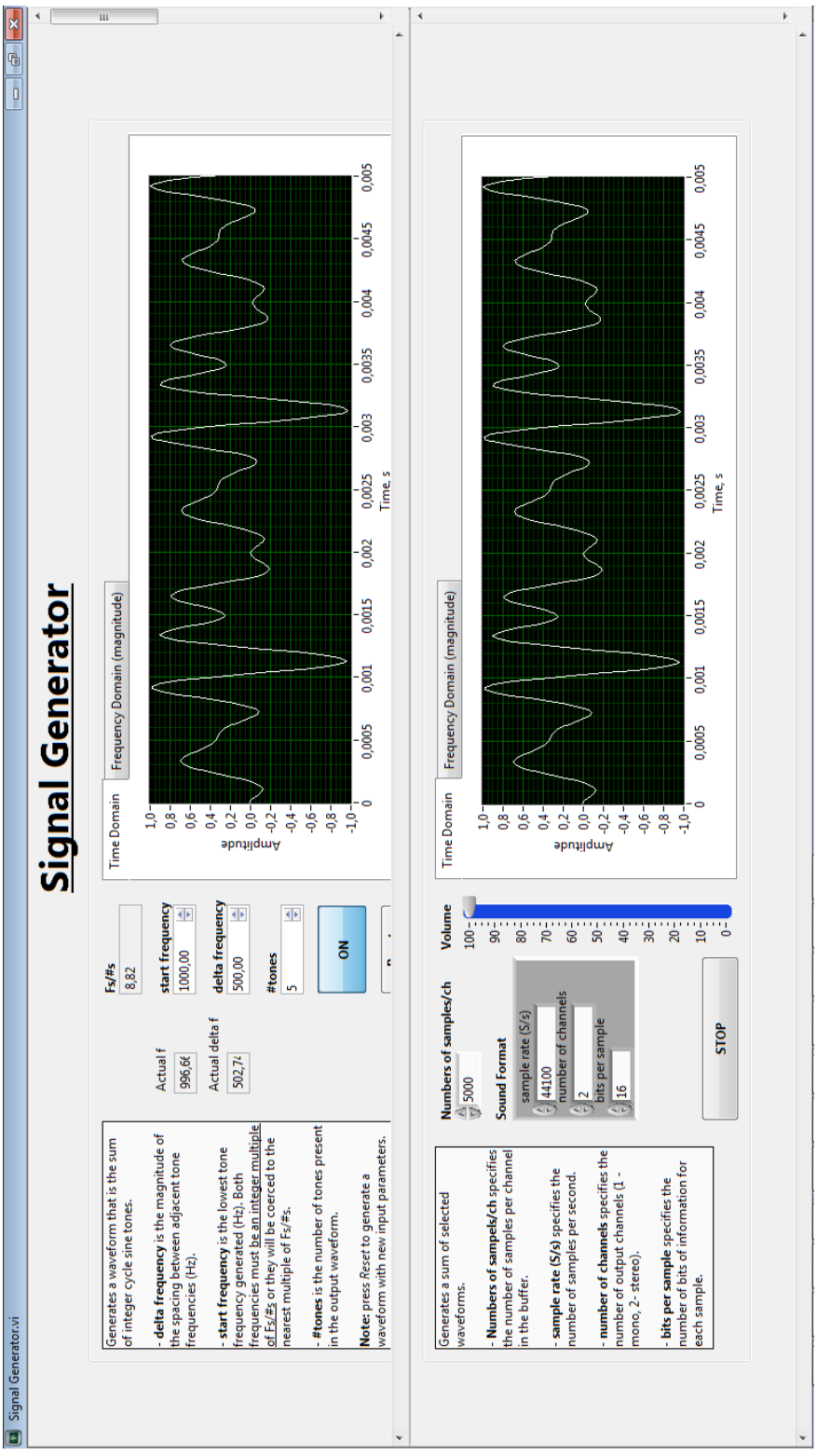
**Lisa 2. TMS320VC5510 DSK arendusplaat**

1. +5 V toide
2. USB
3. JTAG emuleerimine
4. CPLD programmeerimine
5. FPGA
6. AIC23
7. Mikrofoni sisend
8. Sisendliin
9. Väljundliin
10. Kõrvaklapi väljund
11. CPLD
12. Algseadistus
13. Väline JTAG
14. Perifeeria liides
15. Välkmälu
16. Välise mälu liides
17. SDRAM
18. TMS320VC 5510 DSP
19. Sillused
20. LEDid
21. DIP lüliti
22. EHPI

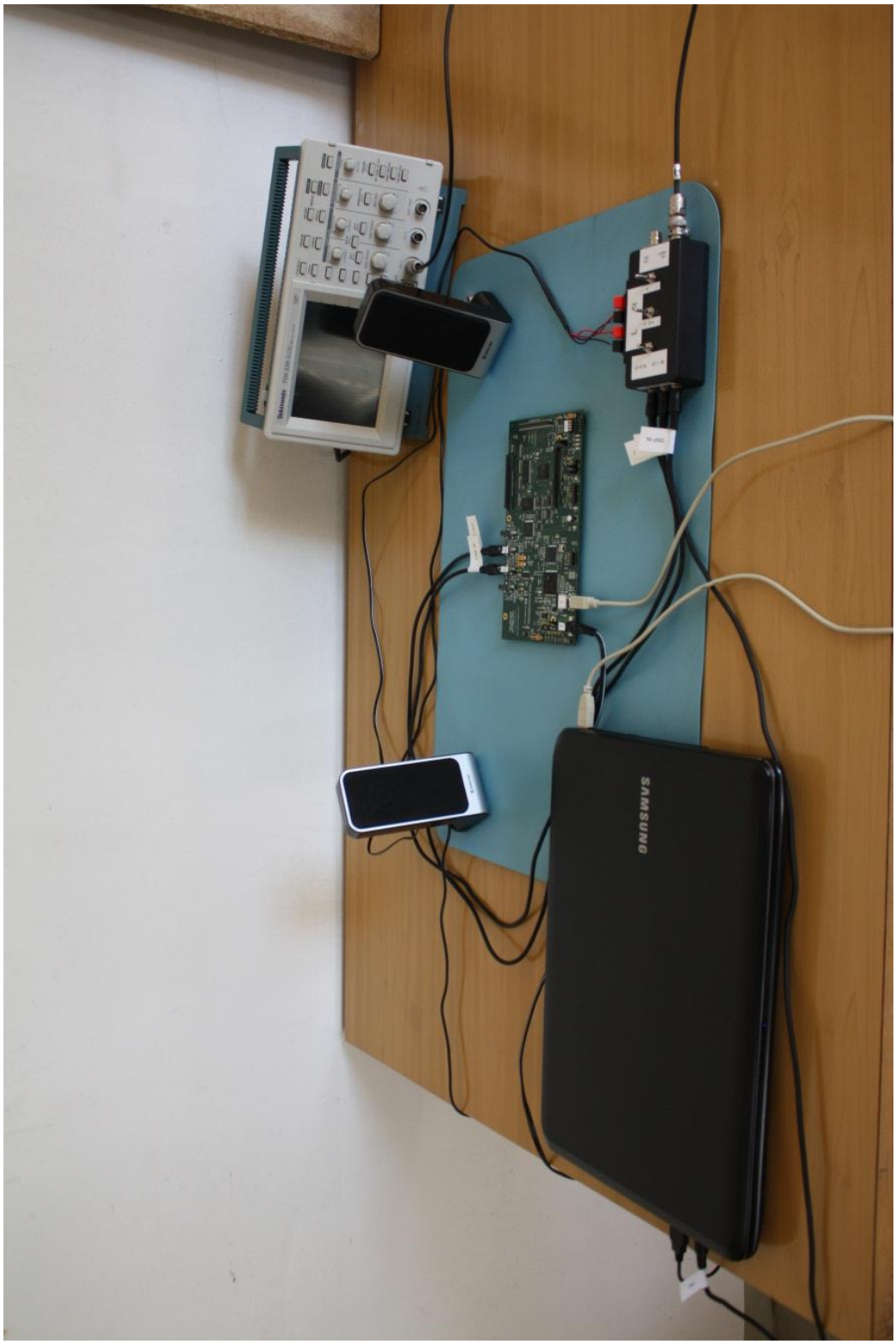


**Lisa 3. Signaalijagur**


## Lisa 4. Signaaligeneraator



**Lisa 5. Katsestendi kooste foto**



## Lisa 7. Esimese praktikumi juhend

# 1. Diskreetimisteoreemi praktiline kontrollimine

Vastavalt Nyquist–Shannoni diskreetimisteoreemile saame me võetud lugemitest algse pideva signaali õigesti taastada, kui pideva signaali ribalaiuse *B* ja diskreetimissageduse $f_s$ vahel kehtib seos:

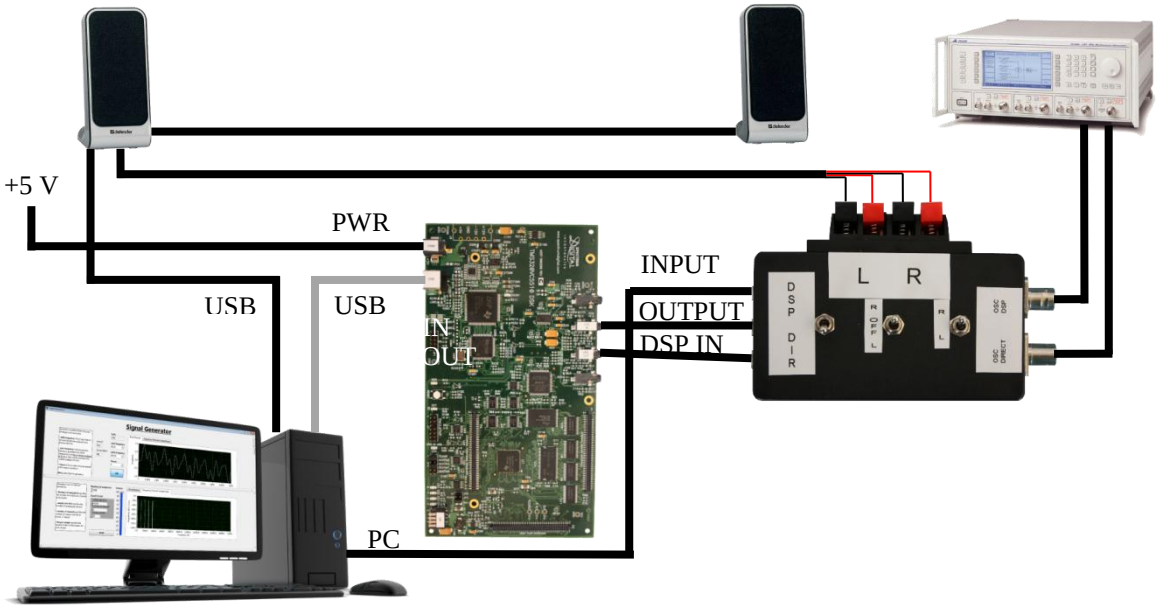
$$B < \frac{f_s}{2} \qquad (1)$$

Seega peab diskreetimissagedus $f_s$ olema üle kahe korra suurem, kui pidavas signaalis esinev kõige kõrgema sageduse komponent. Kuna diskreetimissagedus $f_s$ ei saa olla lõpmata suur, tuleb valida vastavalt sisendsignaalile sobiv diskreetimissagedus.

**Töö käik**

**NB!** Kui sa ei ole randmepaela kaudu maandatud, siis maanda end (nt puutu radiaatorit) enne arendusplaadi puutumist. Vastasel juhul võib sinu kehale kogunenud laeng kahjustada plaadil olevaid komponente

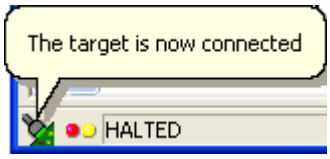
- Koosta skeem nagu on näidatud joonisel 1. Signaalijaguri ja arvuti/arendusplaadi vahele käivatel kaablitel on mõlemas otsas märgitud lipiku külge pesa nimi, kuhu see tuleb ühendada. Kui skeem on koos, siis enne süsteemi toitevõrku ühendamist näita koostatud skeem ette praktikumijuhendajale.




**Joonis 1. Katsestendi koostamise skeem**

- Ava *Code Composer Studio (CCS)* 


- Loo ühendus CCS ja DSK vahel: **Debug** → **Connect** (**Alt+C**). Kui ühendus õnnestus, kuvatakse all nurgas teadaanne.




- Ava projektifail *Praktikum_1.pjt*: **Project** → **Open.** Fail asub töölaual kaustas *DSP praktilised tööd\Praktiline töö nr 1.*
- Mine **Option → Customize** ja vali **Program Load Options** sakk. Seal pane linnuke **Load Program After Build** ette. Siis laetakse programm kohe pärast edukat kompileerimist arendusplaadile.
- Kompileeri vastav programm: **Project → Rebuild All** või  . Edukast kompileerimisest antakse teada **Build** aknas.
- Pane programm käima: **Debug → Run, F5** või  . Plaadil süttib LED0.
- Seejärel käivita signaaligeneraator *Signal Generator*  Kontrolli, et kõik helisagedused kostuksid kõlaritest. Vaatle signaali ja selle spektrit ostsilloskoobiga.
- Seiska programm: **Debug → Halt, Shift + F5 või** 
- Kuva projektifaili sisu vajutades selle kõrval olevale plussmärgile. Ava *Source* kasutas olev fail *Praktikum_1.c*, kus asub programmi põhikood. Püüa aru saada, kuidas see töötab.
- Leia koht, kus määratakse diskreetimissagedus ja muuda seda 24 kHz-ks. Kompileeri ja käivita programm uuesti. Katseta erinevate sagedustega signaaligeneraatoris. Vaatle signaali ja selle spektrit ostsilloskoobiga.
- Korda seda sama diskreetimissagedusega 16 kHz.
- Korda seda sama diskreetimissagedusega 8 kHz.

**Vastata järgmistele küsimustele**

- Mida täheldasid, kui muutsid diskreetimissagedust madalamaks? Millest on see tingitud?
- Mis asi on spektraalne kattuvus ja millal see tekib?
- Miks diskreeditakse heli üldjuhul (nt. CD puhul) sagedusel 44,1 kHz. ?

- Mis kasu saadakse ülediskreetimisest (diskreetimissagedus $f_s$ on üle 2 korra suurem kui B) (valem 1)?

**Kirjandus**


- **Li Tan**, *Digital signal processing: fundamentals and applications,* Amsterdam [etc.] : Elsevier/Academic Press, 2008. Lk 9-10, 13–19.
- **Dag Stranneby, William Walker**, *Digital Signal Processing and Applications*, Amsterdam [etc.] : Elsevier/Newnes, 2004. Lk 7-9
- **Steven W. Smith**, *The Scientist and Engineer's Guide to Digital Signal Processing,* pt. 3. http://www.dspguide.com/ch3/2.htm
- **Saeed V. Vaseghi**, Advanced Digital Signal Processing and Noise Reduction, *Wiley*, 2008. Lk 27-28.

**Lisa 8. Teise praktikumi juhend**

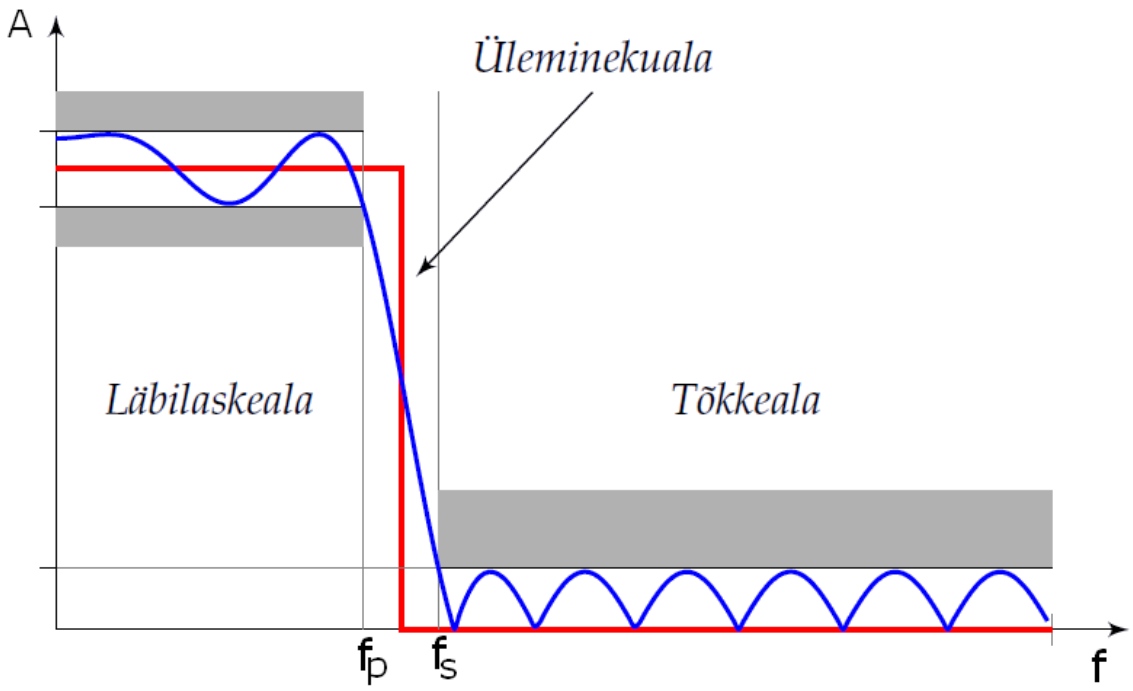
# 2. Filtri tüübi ja parameetrite eksperimentaalne määramine

Filtreid kasutatakse selleks, et eemaldada signaalist soovimatuid sagedusi. Filtri funktsioonist saab tavainimene kõige paremini aimu filtri sageduskarakteristiku järgi, kus on ühel teljel sagedused hertsides ja teisel amplituud detsibellides (dB). Filtri karakteristikuid on võimalik kuvada ka mitmel teisel moel, mida antud töös ei käsitleta.

Põhilised filtrid on:

- Madalpääs – summutatakse kõrgsageduslikud komponendid.
- Kõrgpääs – summutatakse madalsageduslikud komponendid.
- Ribafilter – määratud sagedusvahemik lastakse läbi, teised sagedused summutatakse.
- Tõkkefilter – summutatakse määratud sagedusvahemik.

Filtri parameetrid (joonis 2)




**Joonis 2. Filtri parameetrid madalpääs filtri näitel. Punane joon on ideaalne filter, sinine reaalne filter.**

- Läbilaskeala – selles vahemikus olevaid sagedusi ei summutata (va võngetest tulnud muutused). Määratud sagedusega $f_p$.
- Tõkkeala – selles vahemikus olevad sagedused summutatakse. Määratud sagedusega $f_s$.
- Üleminekuala – vahemik läbilaskealast tõkkealaks ja vastupidi (nt. kõrgpääs filtri puhul). Vahemik $f_p$ kuni $f_s$.

**Töö käik**

**NB!** Kui sa ei ole randmepaela kaudu maandatud, siis maanda end (nt puutu radiaatorit) enne arendusplaadi puutumist. Vastasel juhul võib sinu kehale kogunenud laeng kahjustada plaadil olevaid komponente.

- Sea üles katsestend (meeldetuletuseks joonis 1 esimese praktikumi juhendist). Kui skeem on koos, siis enne süsteemi toitevõrku ühendamist näita koostatud skeem ette praktikumijuhendajale.
- Avada ***Code Composer Studio***, ühendada keskkond arendusplaadiga ja avada projektifail *Praktikum_2.pjt*. Fail asub töölaual kaustas *DSP praktilised tööd**Praktiline töö nr* 2.
- Püüa ilma käivitamata aru saada mida kood teeb. Põhikood on ***Source*** kaustas *Praktikum_2.c* failis.
- Lisa projektile vastavalt juhendaja korraldusle ***filter%n.h***, kus ***%n*** on filtri number. Fail sisaldab filtri impulsskostet.
- Muuda koodis ridu vastavalt lisatud filtrile.

```
//Filtri koefitsendid
#include   "filter%n.h"

//Filtri järk, koefitsientide arv
#define    ORDER      191
```

- Kompileeri programm (kui see on edukas laetakse programm ka plaadile) ja käivita see. LED2 ja LED3 peaksid hakkama vilkuma.
- Filtri rakendamiseks tuleb arendusplaadil vajutada alla lüliti DIP3.
- Käivita signaaligeneraator ***Signal Generator*** ja vali mõni liitsignaal, mis koosneks väga erinevate sagedustega sinusoididest. Kuula vahet koos ja ilma filtrita. Võrdle genereeritud signaali ja töödeldud signaali ning nende sagedusspektreid.
- Vali signaaligeneraatorist $\frac{1}{f}$ müra. Kuula otse signaali generaatorist tulnud signaali ja filtreeritud signaali. Kuula vahet koos ja ilma filtrita. Võrdle genereeritud signaali ja töödeldud signaali ning nende sagedusspektreid.
- Katseta veel seni, kuni saad aru, mis tüüpi filtriga tegu on ja mis sagedusvahemikus võiksid olla läbilaske-ja tõkkealad.

**Vastata järgmistele küsimustele**

- Mis filter oli koodis kasutusel (madal-, kõrgpääs-, riba- või tõkkefilter)?
- Mis sagedusvahemikus oli filtri läbilaskeala ja tõkkeala?
- Mis sagedusel tekitab häireid vahelduvvool Eestis?

- Millist filtrit kasutatakse, kui on tarvis vooluvõrgust tingitud häireid kõrvalda?

**Kirjandus**


- **Li Tan**, *Digital signal processing: fundamentals and applications,* Amsterdam [etc.] : Elsevier/Academic Press, 2008. Lk 3-7.
- **Dag Stranneby, William Walker**, *Digital Signal Processing and Applications*, Amsterdam [etc.] : Elsevier/Newnes, 2004. Lk 7-9
- **Steven W. Smith**, *The Scientist and Engineer's Guide to Digital Signal Processing*, pt. 3. http://www.dspguide.com/ch3/2.htm

## Lisa 9. Kolmanda praktikumi juhend

# 3. FIR filtri loomine

Filtri projekteerimise saab jagada järgmisteks etappideks:

- Pannakse paika projekteeritava filtri soovitud karakteristikud, näiteks läbilaske- ja tõkkeala, lõikesagedus vms.
- Määratakse kindlaks kriteeriumid, mille abil hinnatakse tegeliku realiseeritava filtri kvaliteeti võrreldes soovitud karakteristikutega.
- Valitakse meetod, mis võidab leida parima tulemuse püstitatud probleemi lahendamiseks.
- Katsetatakse filtri toimet ja analüüsitakse tulemit.

Käesolevas praktikumis saab valida kahe probleemi vahel

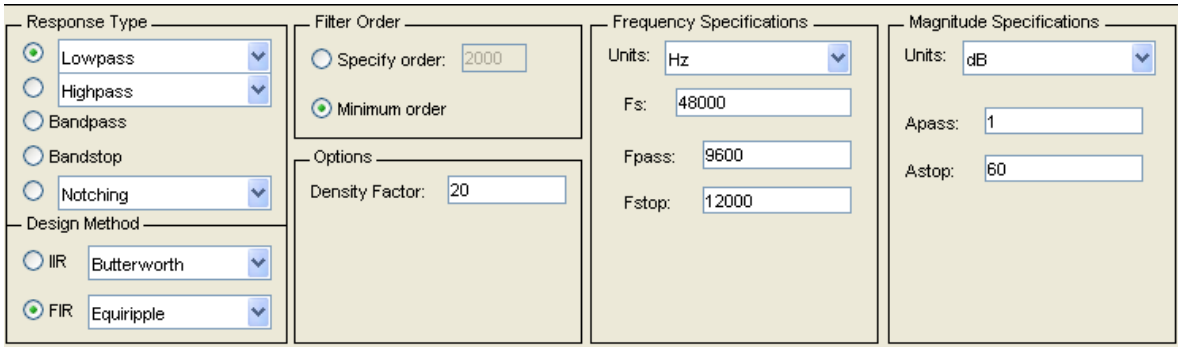
1. Puhtale A noodile on lisandunud müra. Disaini filter, mis taastaks algse puhta sinusoid võimalikult täpselt. Sisendsignaaliks valida signaaligeneraatorist siinus sagedusega 440 Hz ja müra spektraalse tihedusega $10mV/\sqrt{Hz}$.
2. Sinu muusikasüsteemist kostub madalasageduslik „pinin“ sagedusel 150 Hz. Disaini filter võimendi ja kõlarite vahele, mis eemaldaks selle nii, et teisi sagedusi mõjutataks minimaalselt.

Probleem tuleb lahendada **kahe filtri korral**: ühel, kui filter on disainitud kasutades aknameetodit ja teisel, kus filtri disaini meetod on vaba valik.

### Töö käik

**NB!** Kui sa ei ole randmepaela kaudu maandatud, siis maanda end (nt puutu radiaatorit) enne arendusplaadi puutumist. Vastasel juhul võib sinu kehale kogunenud laeng kahjustada plaadil olevaid komponente.

- Sea üles katsestend (meeldetuletuseks joonis 1 esimese praktikumi juhendist). Kui skeem on koos, siis enne süsteemi toitevõrku ühendamist näita koostatud skeem ette praktikumijuhendajale.
- Avada ***Code Composer Studio***, ühendada keskkond arendusplaadiga ja avada projektifail *Praktikum_3.pjt*. Fail asub töölaual kaustas *DSP praktilised tööd**Praktiline töö nr* 3.
- Käivita signaaligeneraator ***Signal Generator*** ja signaali prameetrid vastavalt valitud probleemile.
- Käivita MATLAB ja trüki avanevasse konsooliaknasse **fdatool**. Avaneb graafiline filtri disainimise tööriist ***Filter Design & Analysis Tool***.




**Joonis 3. Filter Design & Analysis Tool sisendparameetrid**

*Response Type* – filtritüüp.

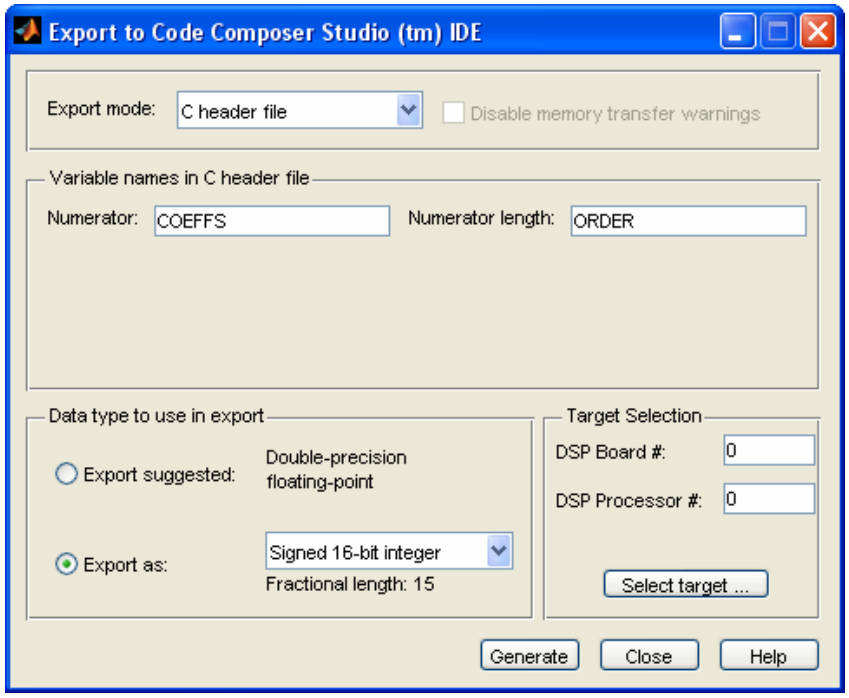
*Design Method* – filtri disaini meetod. Vali kindlasti FIR filter. Üks filtritest peab olema disainitud kasutades aknameetodit (millist täpsemalt, on vaba valik).

*Filter Order* – filtri järk, filter genereeritakse pikkusega $N + 1$, kus $N$ on filtri järk.

*Frequency Specifications* – filtri sageduslikud karakteristikud. Tõkke- ja läbilaskeala määramine. Diskreetimissageduseks on 48 kHz.

*Magnitude Specifications* – amplituudi karakteristikud. Tõkke- ja läbilaskeala amplituudi määramine.

- Sisesta filtri parameetrid ja vajuta **Design Filter**. Graafikul kuvatakse filtri amplituudkoste. Kui oled rahul sellega, siis vali **Targets→Code Composer Studio (tm) IDE**. Vali järgnevad seaded (joonis 4) ja vajuta **Generate**.




**Joonis 4. Filtri eksportimise sätted**

- Kopeeri koefitsiendid loodud failist projektifaili ***filter.h***. Ära unusta ka filtri järgu muutmist põhikoodis ***Praktikum_3.c***.

```
//Filtri järk, koefitsientide arv
#define    ORDER      191
```

- Vali CCS tööriistaribalt **DSP/BIOS→CPU Load Graph**. Kompileeri ja käivita programm ning testi, kas filter töötab sihipäraselt. Jälgi protsessori koormust filtriga ja filtrita.
- Kui oled veendud, et püstitatud probleem on lahendatud, kutsu juhendaja kontrollima.
- Tee samad sammud läbi teise filtri korral.

**Vastata järgmistele küsimustele**

- Võrdle kahte loodud filtrit. Pea silmas filtri pikkust, üleminekuala laiust, tõkke- ja läbilaskeala amplituude ning protsessori ressursikasutust.
- Miks ei saa suure reaalajalise nõudega süsteemis kasutada väga pikki filtreid?
- Kuidas on seotud filtri pikkus ja üleminekuala kuju?
- Mille vastu aitab aknameetodi kasutamine? Kuidas muutub filtri impulsskoste kuju?

**Kirjandus**


- **Li Tan**, *Digital signal processing: fundamentals and applications,* Amsterdam [etc.] : Elsevier/Academic Press, 2008. Lk 215 – 253.
- **Rain Ferents,** *Digitaalne signaalitöötlus. Valitud FIR-filtrite disainimine*, TTÜ Kirjastus, 2011. Lk 9-32.
- **Steven W. Smith**, *The Scientist and Engineer's Guide to Digital Signal Processing*, pt. 14 http://www.dspguide.com/ch14.htm

**Lisa 10. Praktikumijuhendite ja tarkvara DVD**

DVD-l on:

- Signaaligeneraatori *Signal Generator* tarkvaja paigaldaja ja lähtekood.
- TMS320VC5510 arendusplaadi draiver ja Code Composer Studio tarkvara paigaldaja.
- Praktikumides kasutatavad projektifailid.
- Praktikumide juhendid PDF kujul.