УНИВЕРЗИТЕТ У КРАГУЈЕВЦУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА ЧАЧАК

мр Миленко Ћирић, дипл.инж.ел.

РАЗВОЈ САВРЕМЕНИХ *MULTIRATE* МЕТОДА И ТЕХНИКА КОД ПРОЈЕКТОВАЊА УСКОПОЈАСНИХ ДИГИТАЛНИХ ФИЛТАРА

Докторска дисертација

САДРЖАЈ

1. УВОД

У савременим комуникацијама и рачунарским системима, један од битних сегмената представља дигитална обрада сигнала. Развој дигиталне обраде сигнала условљен је радом великог броја истраживача у овој области, што је резултирало увођењем нових техника у дигиталној обради сигнала, побољшаним алгоритмима, као и све бржим и квалитетнијим платформама за реализацију предложених решења.

Један од основних елемената у дигиталној обради сигнала су дигитални филтри. Поред линеарних временски инваријантних система где је учестаност одабирања иста на излазу система као и на улазу, као и у свим чворовима система, са развојем нових апликација све више се захтева да се у различитим чворовима процесирају сигнали са различитим учестаностима одабирања од улазне. Овакви захтеви условљавају коришћење све више хардверских ресурса. Уобичајено решење за овакве захтеве је примена вишебрзинских система (енгл. *Multirate Digital Signal Processing*) код којих се конверзија учестаности одабирања врши у различитим тачкама система, у зависности од потреба за процесирањем сигнала. Промене учестаности одабирања врше се колима за увећање учестаности одабирања (енгл. *up-sampler*) и колима за снижење учестаности одабирања (енгл. *down-sampler*).

Multirate технике дигиталног процесирања представљају у последње време једну од веома често употребљаваних техника дигиталне обраде сигнала, као што је то случај код радио комуникација, обраде слике, дигиталне обраде звука, рачунаских комуникација итд. Суштинска предност *multirate* техника је та што њена примена значајно снижава број аритметичких операција потребних за захтевану обраду сигнала, а са тим у вези и упрошћава комплексност дигиталних структура. Оваква ефикасност *multirate* алгоритама се заснива на способности за симултано употребљавање различитих брзина одабирања у различитим деловима дигиталног система

Промене учестаности одабирања врше се колима за увећање учестаности одабирања (енгл. *up-sampler*) и колима за снижење учестаности одабирања (енгл. *down-sampler*). Најкритичнија фаза у процесу промене учестаности одабирања је спречавање преклапања спектара код децимације, односно уклањање периодичних одраза код интерполације сигнала. Коришћењем одговарајућих филтара може се обезбедити да се *multirate* обрада сигнала врши без значајне деградације корисног сигнала.

Поред ове основне улоге, кола за конверзију учестаности одабирања заједно са другим основним елементима (сабирачима, множачима, колима за кашњење ...) могу бити

корисна за реализацију различитих задатака, као и за побољшање перформанси дигиталних система. У том смислу је развијен поступак проточне обраде сигнала (енгл. *Pipelining –Interleaving*), тј. *PI* поступак са циљем да се побољша ефикасност дигиталног филтрирања. Његовом употребом у реализацији различитих задатака, поред тога што се омогућава једноставније решавање различитих проблема, постижу се значајне уштеде хардверских ресурса. Наиме, редак је случај да су учестаност одабирања и такт дигиталног система усаглашени и то нарочито у случајевима када се дигитални филтри реализују програмабилним хардвером. Тако, уколико је учестаност одабирања знатно нижа од учестаности такта на којој раде елементи система, непотребно се троше ресурси расположивих елемената на чипу из тог разлога што кад се обаве задате операције, елементи чекају долазак наредног одбирка који би требао да се процесира. Иако би хардвер у међувремену могао да се користи и за друге улоге, то се не чини јер то не дозвољава конфигурација система и редослед операција. Применом проточне обраде сигнала омогућено је да се сигнали обрађују паралелно уместо да се обрада обавља секвенцијално.

На основу оваквог приступа омогућено је да се изведу различите технике реализације дигиталних филтара [1], [2]. У овом раду ће бити извршена анализа поступка проточне обраде сигнала за реализацију појединачних филтара, било да се ради о *FIR* или *IIR* филтрима, ради одређивања могућности и ограничења оваквог поступка у смислу начина реализације, броја канала и вредности грешке квантизације. Анализа ће бити вршена упоређивањем обе методе (са и без примене *multirate* поступака) и у временском и у фреквенцијском домену. Биће одређена максимална вредност грешке у целом опсегу учестаности од 0 до π, како би се донео закључак о применљивости предложене методе. Приликом сваког испитивања ће се водити рачуна да предложена решења буду применљива у пракси, у смислу да буду задовољени услови критичне повратне петље, да филтри не буду нереално високог реда. Поред овога, извршиће се анализа утицаја ефекта коначне дужине кодне речи како би се проценио утицај овог ефекта на филтре изведене техником проточне обраде сигнала. Биће предложене неке од нових *multirate* метода које су погодне за овакве реализације.

Циљ овог рада је да се одреде такве примене у којима ефикасност *multirate* поступака у комбинацији са *PI* техником долази до пуног изражаја. Ова побољшања проистичу из комбиновања операција самих реализација *multirate* дигиталних структура и примене технике проточне обраде сигнала на њих, јер ће се показати да су одређене операције комплементарне као и да се поједини блокови нових структура добијених

применом *PI* поступка могу реализовати у мање корака. Тако ће реализације неких сложених дигиталних структура као што су филтри са јако уском прелазном зоном, ускопојасни филтри, филтри са јако малом осетљивошћу на промене коефицијената итд., чија је реализација стандардним методама неизводљива (нереално су високог реда, кашњења су превелика, грешка услед *fixed point* имплементације је неприхватљива..), применом поменутих поступака ће постати могућа.

Дигитални филтри са тако строгим захтевима налазе примену у савременим апликацијама као што су обрада аудио и видео сигнала, „*up*“ и „*down*“ конверзија са великим фактором, процесирање у реалном времену итд. Нарочито је велики напредак извршен код дигиталне обраде сигнала у војној примени где се захтева већа тачност и где су критеријуми строжији него у цивилној пракси. Значајну улогу дигитално процесирање и у склопу њега дигитално филтрирање, нарочито добија са развојем Софтверски Дефинисаног Радија (*Software Defuned Radio - SDR*), који представља технологију у коме се користе софверски модули за извршавање радио функција као што су модулација-демодулација, кодовање, генерисање сигнала и сл. За ту намену су развијене различите хардверске платформе које се састоје из дигиталних процесора сигнала, микропроцесора опште намене или *FPGA* чипова. Ове платформе омогућавају изградњу софтверски реконфигурабилних радио система код којих је могућ динамички избор параметара за сваки од функционалних модула датог система. Брз развој програма *SDR* проистиче из изузетне експанзије и напретка електронских, телекомуникационих и рачунарских технологија и широко се примењује у различитим војним и цивилним апликацијама као што су радар, *Bluetooth, WLAN, GPS, WCDMA, CDMA, GPRS* итд.

Методе реализације филтара које ће бити предложене у овом раду, биће тестиране симулацијом рада нове структуре на *Xilinx*-ovom *FPGA* чипу генерације *Virtex 5* и извршиће се снимање амплитудских и фазних карактеристика коришћењем *Xilinx*-овог програмског пакета *System Generator* (верзија 14.7, са важећом лиценцом). На тај начин ће моћи да се добије оцена о примењивости предложених модификација у смислу одступања амплитудских и фазних карактеристика, грешке услед примене техника фиксног зареза, кашњења итд..

Затим ће се наставити корак даље, и предложене структуре ће бити имплементиране у пракси на *FPGA real-time* платформама произвођача *National Instruments.* За сваку предложену филтерску структуру извршиће се провера *FPGA* имплементације у стварности, снимањем карактеристика филтра, и након тога ће се добити потпуна слика о успешност предложених метода, обзиром да се на тај начин узима

у обзир и грешка аналогно-дигиталног конвертора и да се тестира комплетан дигитални филтер.

Рад се састоји из девет поглавља. Након првог уводног поглавља, у другом поглављу су изложене основе вишебрзиских система дигиталног процесирања, при чему су посебно обрађене оне реализације које ће бити предмет примене поступка проточне обраде сигнала. У трећем поглављу је представљен поступак проточне обраде сигнала и у временском и у фреквенцијском домену. Показано је на који начин се врши процесирање два и више сигнала истим филтром а да при томе не дође до било какве дегенерације неког од процесираних сигнала. Поред основног начина примене принципа проточности приказан је и објашњен модификовани *PI* поступак за реализацију каскадне везе филтара. У четвртом поглављу је извршена анализа *PI* поступка на реализацију *FIR* и *IIR* филтара, када се реализују појединачно као и када се примењује модификовани *PI* поступак за реализацију каскадне везе филтара. Испитивање је вршено по принципу одбирак по одбирак, одређивањем амплитудске карактеристике филтара реализованих у PI техници, као и упоређивањем излаза у временском домену филтара реализованих на уобичајени начин и филтара реализованих у *PI* техници. Пето поглавље се бави применама *PI* поступка на оне структуре у вишебрзинским системима дигиталног процесирања које су погодне за реализацију у *PI* техници. Прво је показано како се *PI* поступак може ефикасно применити на вишестепене филтре са великим бројем степена, затим на ускопојасне филтре изведене поступком фреквенцијског маскирања и на крају је изведен и приказан начин примене *PI* поступка на *QMF* банке са структуром стабла. Предложене реализације су испитане у реалном времену снимањем амплитудске карактеристике, извршено је упоређивање у временском домену и проверен је утицај ефекта коначне дужине кодне речи. За симулације испитиваних модела коришћен је програмски пакет *Matlab* (верзија 8.0.0.783 са важећом лиценцом).

У шестом поглављу су приказане примене предложених метода на реализације сложенијих структура дигиталног филтрирања. Циљ примене ових метода је имплементација дигиталних структура са карактеристикама које није могуће постићи применом неких од уобичајених поступака, а које се дају као готова решења код примене софтверских алата за пројектовање дигиталних структура. У том смислу су представљене следеће имплементације:

- Филтри за велике брзине обраде (*FIR* филтер у комбинацији са *IIR* секцијама другог реда) у *PI* техници са минималним бројем елемената за кашњење. Анализа овако реализованог филтра вршена је применом програмског пакета *Matematuca 6* и

софтверског алата *SchematucSolver Versuon 2*, а провера филтра је извршена применом симболичког процесирања.

- Вишестепени ускопојасни филтри са *IIR* комплементарним филтерским паровима са ниском осетљивошћу амплитудске карактеристике на промене коефицијената филтра. Представљена је једна ефикасна реализација овакве методе на *Xilinx*-овом *FPGA* чипу генерације *Virtex 5*. Извршено је снимање карактеристика филтра и показане су предности примене овакве методе у смислу сужавања пропусног опсега филтра.
- Филтри са јако стрмом амплитудском карактеристиком у прелазној зони реализовани комбиновањем „sharpening“ методе и технике фреквенцијког маскирања. Приказана је реализација која осим сужавања прелазне зоне има за циљ задржавање линеарне фазне карактеристике.
- *CIC* филтри код “*up*” и “*down*” конверзије са високим фактором промене. Приказан је поступак увећања реда конверзије комбиновањем операција везаних за поступак фреквенцијског маскирања, процеса децимације (интерполације) и метода сужавања прелазне зоне филтра. Извршена је имплементација филтра и показана је примена поступка на реалне сигнале (*16QAM* модулисани сигнал).

У седмом поглављу је дат закључак рада, а у осмом поглављу је приказан списак коришћене литературе. У деветом поглављу су представљени коришћени програми у току израде овог докторског рада.

2. ВИШЕБРЗИНСКИ (*MULTIRATE*) СИСТЕМИ ЗА ОБРАДУ СИГНАЛА

Код линеарних временски инваријантних система учестаност одабирања је иста на излазу као и на улазу, а такође и у свим чворовима система, тј. остаје иста у свим деловима система. Са развојем савремених апликација јавља се потреба за конверзијом сигнала у еквивалентни сигнал коме је промењена учестаност одабирања. Системи код којих се мења учестаност одабирања називају се вишебрзински системи (енгл. *multirate systems*). Уколико је дата учестаност одабирања $F = 1/T$ (T - периода одабирања), и ако је потребно да се промени на нпр. $F' = 1/T'$, биће извршена промена учестаности одабирања. Ако је нова учестаност одабирања већа од оригиналне, односно:

$$F' > F,$$

или

$$T' < T \tag{2.1}$$

процес се назива експанзија или интерполација (енгл. *interpolation*).

Процес промене учестаности одабирања на вредност F' која је мања од старе вредности F', односно:

$$F' < F, \tag{2.2}$$

или

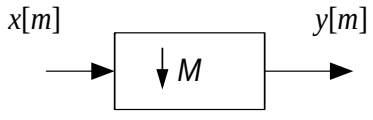
$$T' > T$$

се назива компресија или децимација (енгл. *decimation*).

2.1 ПРОЦЕС ДЕЦИМАЦИЈЕ

Процес снижавања учестаности одабирања се назива децимација. На слици 2.1 је приказан блок дијаграм компресора за степен компресије M, и може се описати следећом једначином:

$$y[m] = x[mM]. \tag{2.3}$$



Слика 2.1 Блок дијаграм компресије сигнала

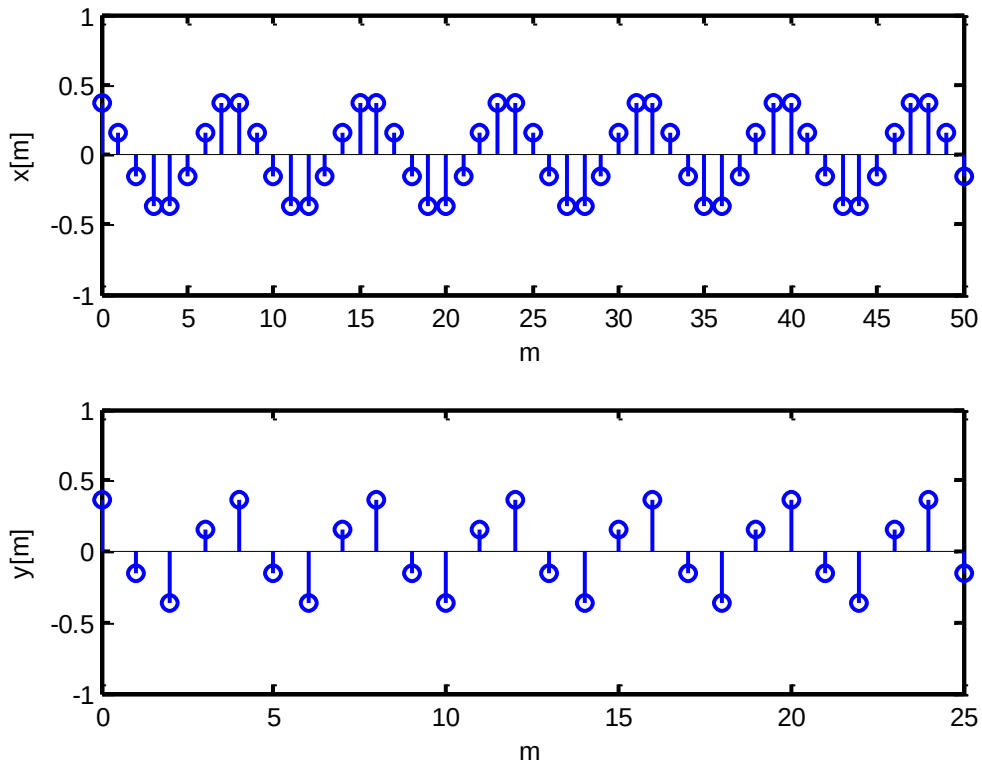
Одбирак m излазног сигнала из компресора је једнак одбирку $M*n$ улазног сигнала, тј. задржавају се само они улазни одбирци који су мултипли од M. Процес компресије у временском домену је приказан на слици 2.2 за $M = 2$.

Међутим, снижавањем учестаности одабирања може да се догоди да није више задовољена теорема одабирања тј. у спектру излазног сигнала може доћи до преклапања спектралних компонената (енгл. *aliasung*). Ефекти преклапања се најбоље сагледавају посматрајући процес децимације у фреквенцијском домену. Применом z-трансформације на обе стране једначине (2.3) добија се:

$$Y(z) = \sum_{m=-\infty}^{\infty} x[Mm]z^{-m} \tag{2.4}$$

Уколико се дефинише секвенца $y_s[n]$ која се добија дискретним одмеравањем улазне секвенце $x[n]$:

$$y_s[n] = x[n]s_M[n] = \begin{cases} x[n], & n = 0, \pm M, \pm 2M, \ldots, \\ 0, & \text{другде} \end{cases} \tag{2.5}$$



Слика 2.2 Процес компресије у временском домену за $M = 2$

где је

$$s_M[n]=\begin{cases}1, & n=0,\pm M,\pm 2M,\ldots,\\ 0, & \text{другде}\end{cases}, \tag{2.6}$$

може писати

$$\begin{aligned}Y(z)&=\sum_{m=-\infty}^{\infty}x[Mm]z^{-m}=\sum_{n=-\infty}^{\infty}y_s[n]z^{-n}=\sum_{m=-\infty}^{\infty}y_s[mM]z^{-m}=\sum_{k=-\infty}^{\infty}y_s[k]z^{-k/M}\\&=Y_s\left(z^{1/M}\right)\end{aligned} \tag{2.7}$$

Z-трансформација сигнала $y_s[n]$ може да се изрази у следећој форми

$$Y_s(z)=\sum_{n=-\infty}^{\infty}s_M[n]x[n]z^{-n} \tag{2.8}$$

Ради прегледности, секвенца $s_M[n]$ може да се дефинише на следећи начин:

$$s_M[n]=\frac{1}{M}\sum_{k=0}^{M-1}W_M^{kn} \tag{2.9}$$

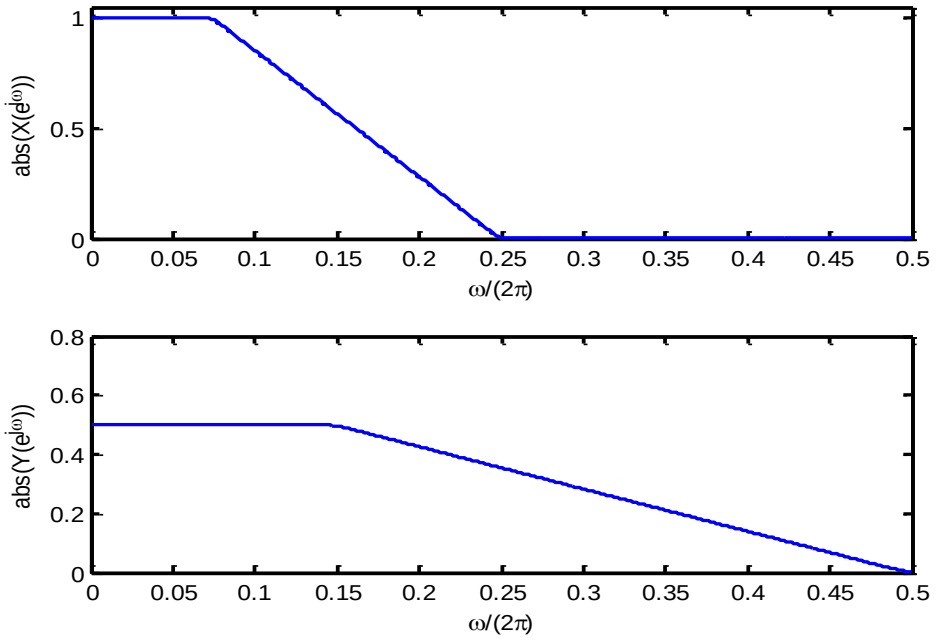
где је $W_M{}^{kn}=e^{-j2\pi kn/M}$. Заменом једначине (2.9) у (2.8) добија се:

$$\begin{aligned}Y_s(z)&=\frac{1}{M}\sum_{n=-\infty}^{\infty}\left(\sum_{k=0}^{M-1}W_M^{kn}\right)x[n]z^{-n}=\frac{1}{M}\sum_{k=0}^{M-1}\left(\sum_{n=-\infty}^{\infty}x[n]W_M^{kn}z^{-n}\right)\\&=\frac{1}{M}X\left(zW_M^{-k}\right)\end{aligned}. \tag{2.10}$$

Применом трансформације $z\to e^{j\omega}$ на једначину (2.10) и имајући на уму да је $W_M{}^{-k}=e^{j2\pi k/M}$ добија се да је

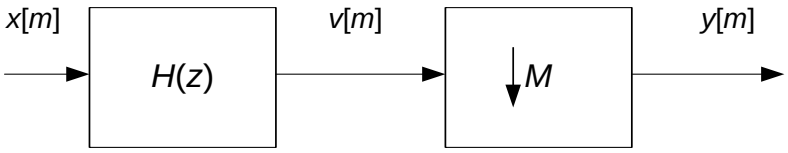
$$Y\left(e^{j\omega}\right)=\frac{1}{M}\sum_{k=0}^{M-1}X\left(e^{j(\omega-2\pi k)/M}\right). \tag{2.11}$$

Претходна једначина показује утицај поступка децимације на спектар излазног сигнала. Уочава се да је спектар $Y\left(e^{jw}\right)$ сума од M униформно померених и продужених верзија спектра $X\left(e^{jw}\right)$ скалираних за фактор $1/M$. На основу релације (2.11) и Слике 2.3 закључује се да ће до преклапања доћи ако је ширина спектра оригиналног сигнала већа од π/M . То значи да ће децимација једино сигнала чији је спектар ограничен на π/M бити извршена без дисторзије. Дакле, пре компресије је потребно ограничити спектар улазног сигнала НФ филтром, а каскадна веза филтра и компресора чини дециматор.



Слика 2.3 Процес компресије у фреквенцијском домену за $M = 2$

Блок шема поступка децимације је приказана на слици 2.4, а блок означен са M се назива компресор.

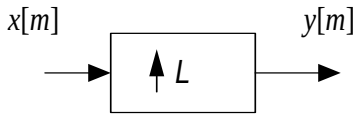


Слика 2.4 Блок шема дециматора

2.2 ПРОЦЕС ИНТЕРПОЛАЦИЈЕ

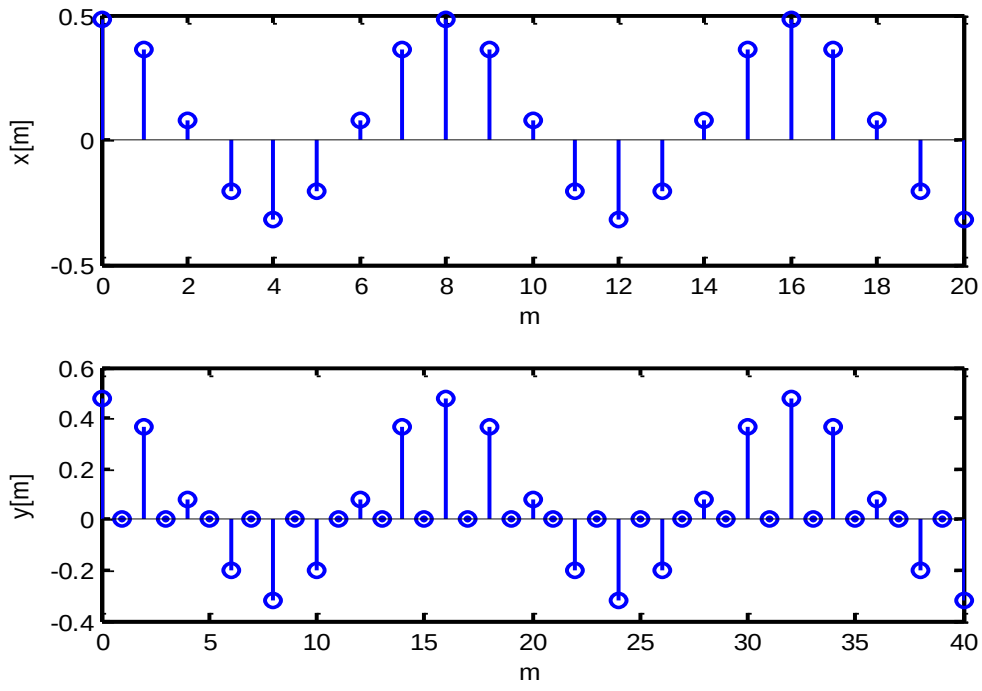
Процес увећања учестаности одабирања се назива интерполација или експанзија. На Слици 2.5 је приказан блок дијаграм експандора за степен експанзије L, и може се описати следећом једначином:

$$y[m]=\begin{cases} x[m/L], & m=0,\pm L,\pm 2L,\cdots \\ 0, & \text{другде} \end{cases} \tag{2.12}$$



Слика 2.5 Блок дијаграм експанзије сигнала

Увећање учестаности одабирања врши се интерполацијом L-1 нових одбирака чија је вредност једнака нули између свака два сукцесивна одбирка улазног сигнала. Процес експанзије у временском домену је приказан на слици 2.6 за $L=2$.

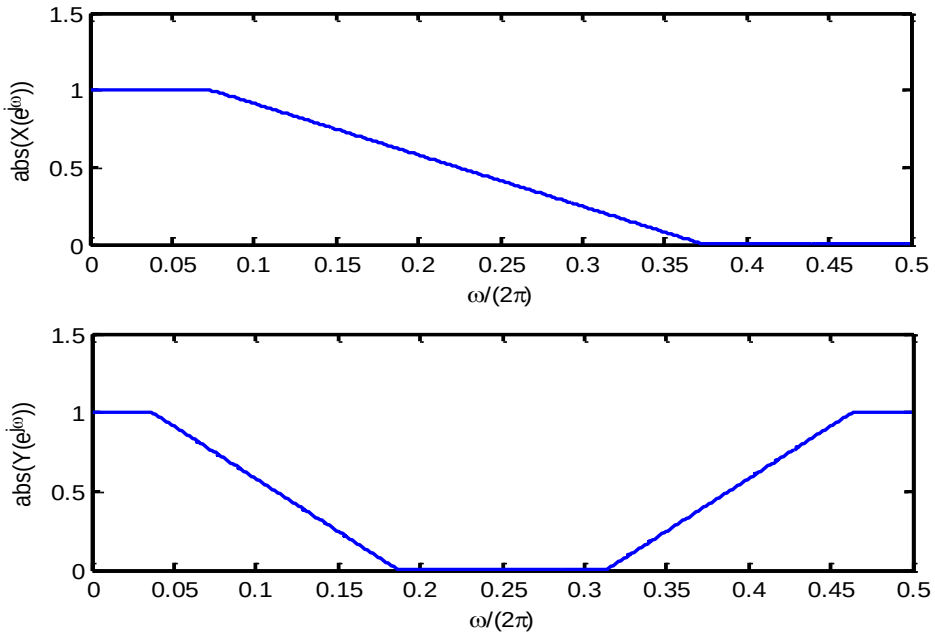


Слика 2.6 Процес експанзије у временском домену за $L=2$

Спектар експандованог сигнала се одређује директном применом *Z*-трансформације на једначину (2.12):

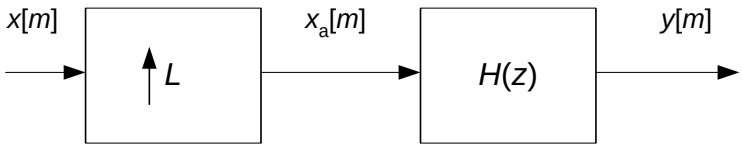
$$Y(z)=\sum_{m=-\infty}^{\infty}y[m]z^{-m}=\sum_{\substack{n=-\infty\\ n=mL}}^{\infty}x[n/L]z^{-n}=\sum_{m=-\infty}^{\infty}x[m]z^{-Lm}=X\left(z^{L}\right), \tag{2.13}$$

што значи да експандовање сигнала у временском домену за фактор L проузрокује појаву L-1 реплика у фреквенцијском домену, као што је то приказано на слици 2.7. Корисни спектар је сабијен у опсег $0\le|w|\le\pi/L$, а изван фреквенције π/L појављују се реплике које треба елиминисати НФ филтром.



Slika 2.7 Процес експанзије у фреквенцијском домену за $L=2$:

Каскадна веза експандора и НФ филтра назива се интерполатор и приказан је на слици 2.8, а блок означен са *L* представља експандор учестаности одабирања.



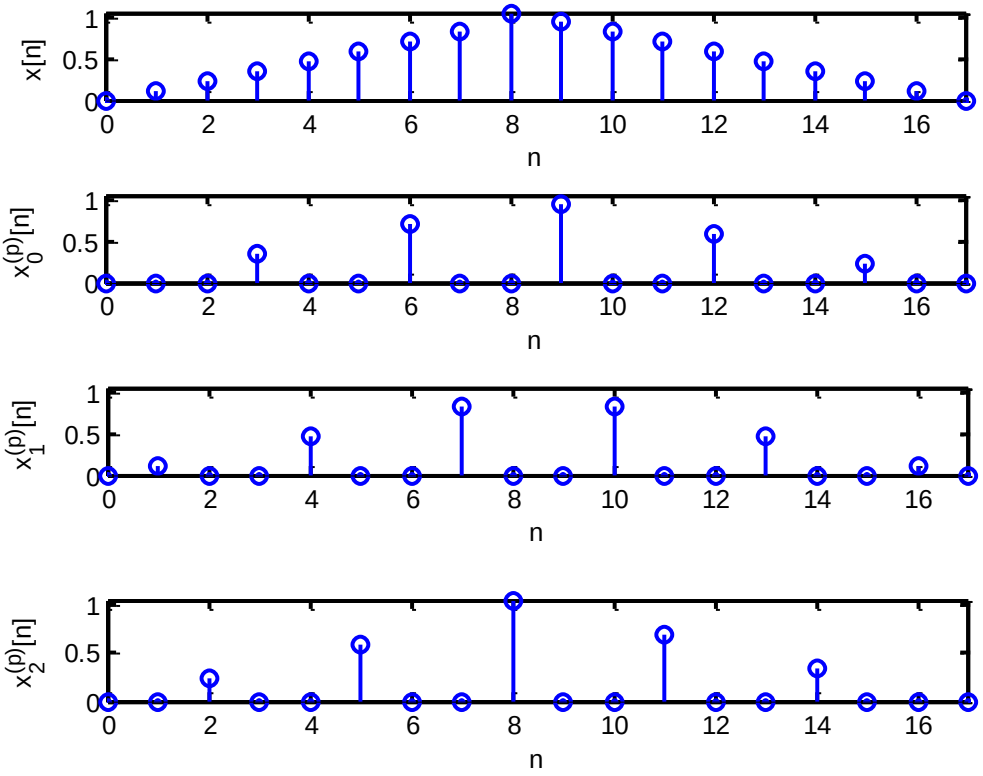
Слика 2.8 Блок шема интерполатора

2.3 ПОЛИФАЗНА ДЕКОМПОЗИЦИЈА

Уколико је дефинисана временски померена функција одабирања $s_{M,k}(n) = s_M(n\text{-}k)$, где је $k = 0, 1, 2, \ldots, M\text{-}1$ на следећи начин:

$$s_{M,k}[n] = \begin{cases} 1, & n = k, \pm M + k, \pm 2M + k, \ldots, \\ 0, & drugde \end{cases}, \quad k = 0, 1, 2, \ldots, M-1. \tag{2.14}$$

Множењем сигнала $x[n]$ одговарајућим функцијама одабирања $s_{M,k}(n)$, $k = 0, 1, 2...M\text{-}1$ добијају се сигнали $\{x_k^{(p)}[n]\}$, $k = 0, 1, 2...M\text{-}1$. Са слике 2.9 уочава се да је сигнал $x[n]$ једнак суми $M\text{-}1$ овако изведених компонената (на слици 2.9 је изабрано $M = 3$), што се може представити на следећи начин:



Слика 2.9 Декомпозиција сигнала $x(n)$ на три полифазне компоненте $x_0^{(p)}(n), x_1^{(p)}(n)$ и $x_2^{(p)}(n)$.

$$\begin{aligned} x[n] &= x_0^{(p)}[n] + x_1^{(p)}[n] + x_2^{(p)}[n] \\ &= x[n]s_{3,0}[n] + x[n]s_{3,1}[n] + x[n]s_{3,2}[n] \\ &= x[n]s_3[n] + x[n]s_3[n-1] + x[n]s_3[n-2]. \end{aligned} \tag{2.15}$$

За произвољно $x[n]$ и M можемо писати:

$$x[n] = \sum_{k=0}^{M-1} x_k^{(p)}[n] = \sum_{k=0}^{M-1} x[n]s_M[n-k] \tag{2.16}$$

Оваква презентација се назива полифазна декомпозиција дискретног сигнала. Компоненте $x_k^{(p)}[n]$, $k = 0, 1, 2, \ldots, M\text{-}1$, се називају полифазне компоненте сигнала $x[n]$. Релација 2.16 одређује полифазну репрезентацију сигнала у временском домену. Примењујући *Z*-трансформацију

$$X(z) = \sum_{n=-\infty}^{\infty} x[n]z^{-n} \tag{2.17}$$

на ову релацију добија се презентација полифазне декомпозиције у фреквенцијском домену. Ради једноставности ће прво бити размотрена Z-трансформација три полифазне компоненте за секвенцу коначне дужине и то за $M = 3$ и дужину секвенце $N = 18$.
Z-трансформација овакве секвенце износи

$$\begin{aligned} X(z) = {} & x[0] + x[1]z^{-1} + x[2]z^{-2} + x[3]z^{-3} + x[4]z^{-4} + x[5]z^{-5} + x[6]z^{-6} + x[7]z^{-7} \\ & + x[8]z^{-8} + x[9]z^{-9} + x[10]z^{-10} + x[11]z^{-11} + x[12]z^{-12} + x[13]z^{-13} + x[14]z^{-14} \\ & + x[15]z^{-15} + x[16]z^{-16} + x[17]z^{-17} \end{aligned} . \tag{2.18}$$

Ако се преуреди израз (2.18) тако да је

$$\begin{aligned} X(z) = {} & \left(x[0] + x[3]z^{-3} + x[6]z^{-6} + x[9]z^{-9} + x[12]z^{-12} + x[15]z^{-15}\right) + \\ & + \left(x[1]z^{-1} + x[4]z^{-4} + x[7]z^{-7} + x[10]z^{-10} + x[13]z^{-13} + x[16]z^{-16}\right) \\ & + \left(x[2]z^{-2} + x[5]z^{-5} + x[8]z^{-8} + x[11]z^{-11} + x[14]z^{-14} + x[17]z^{-17}\right), \end{aligned} \tag{2.19}$$

$X(z)$ може сада да се изрази на следећи начин:

$$\begin{aligned} X(z) &= \left(x[0]+x[3]z^{-3}+x[6]z^{-6}+x[9]z^{-9}+x[12]z^{-12}x[15]z^{-15}\right) \\ &+ z^{-1}\left(x[1]+x[4]z^{-3}+x[7]z^{-6}+x[10]z^{-9}+x[13]z^{-12}+x[16]z^{-15}\right) \\ &+ z^{-2}\left(x[2]+x[5]z^{-3}+x[8]z^{-6}+x[11]z^{-9}+x[14]z^{-12}+x[17]z^{-15}\right), \end{aligned} \tag{2.20}$$

или у сажетијем облику

$$X(z)=\sum_{k=0}^{2} z^{-k} X_k{}^{(p)}(z^3) \tag{2.21}$$

где је

$$X_k^{(p)}(z^3)=\sum_{n=0}^{14} x[3m+k]z^{-3m}, \qquad k=0,1,2. \tag{2.22}$$

Из једначина (2.21) и (2.22) долази се до закључка да је Z-трансформација полифазне компоненте $x_k{}^{(p)}[n]$

$$\left\{x_k{}^{(p)}[n]\right\} \leftarrow z \rightarrow z^{-k} X_k^{(p)}\left(z^3\right), \qquad k=0,1,2. \tag{2.23}$$

Генерално, произвољна секвенца $x[n]$ може да се разложи на M полифазних компонената. Ако је $x[n]$ секвенца бесконачне дужине и ако је $n = Mm + k$, важиће

$$X(z)=\sum_{k=0}^{M-1}\sum_{m=-\infty}^{\infty} x(mM+k)z^{-(mM+k)} = \sum_{k=0}^{M-1} z^{-k} X_k^{(p)}\left(z^M\right) \tag{2.24}$$

где је

$$X_k^{(p)}(z^M)=\sum_{m=0}^{\infty} x[mM+k]z^{-mM}. \tag{2.25}$$

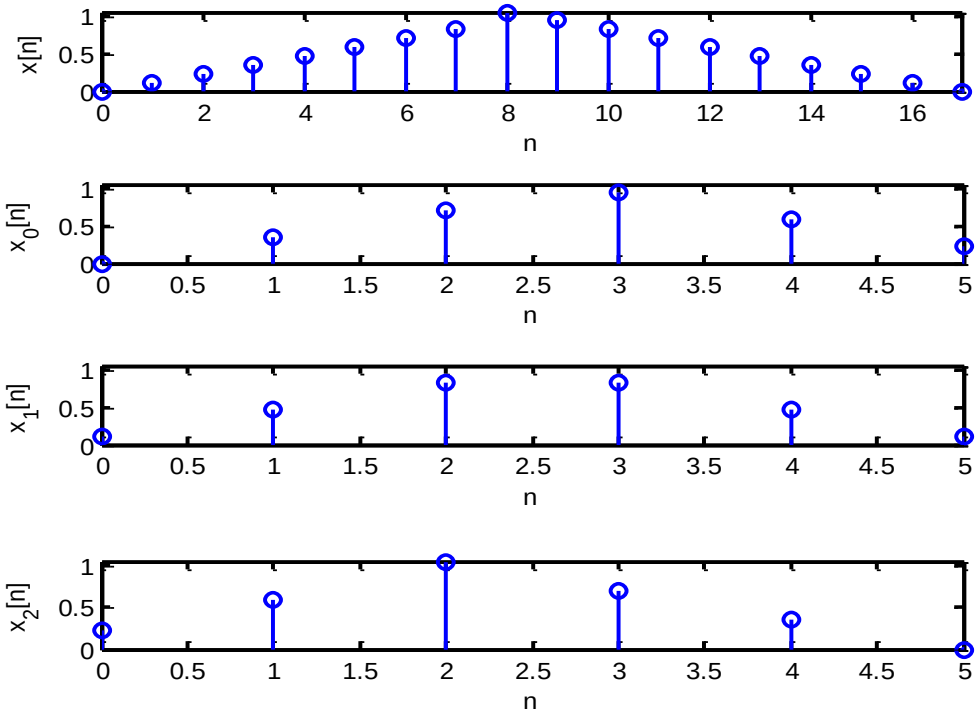
Из једначина (2.24) и (2.25) може се донети закључак да је Z-трансформација полифазне компоненте $x_k{}^{(p)}[n]$

$$\left\{x_k^{(p)}[n]\right\} \leftarrow z \rightarrow z^{-k} X_k^{(p)}\left(z^M\right), \qquad k=0,1,2,\ldots,M-1. \tag{2.26}$$

тако да је Z-трансформација сигнала $x[n]$ у функцији M полифазних компонената

$$X(z)=X_0^{(p)}\left(z^M\right)+z^{-1}X_1^{(p)}\left(z^M\right)+z^{-2}X_2^{(p)}\left(z^M\right)+\ldots+z^{-(M-1)}X_{M-1}^{p}\left(z^M\right). \tag{2.27}$$

Полифазна декомпозиција има широку примену у вишебрзинским системима јер се њеном применом могу добити ефикасне структуре. Полифазне компоненте се добијају дискретним одмеравањем улазне секвенце $x[n]$, и свака компонента садржи $M-1$ нула између свака два суседна одбирка. Како су потребни само ненулти одбирци, нулти одбирци могу да се одбаце из полифазне секвенце $x_k^{(p)}[n]$. Ово одбацивање нултих одбирака извршава се снижавањем учестаности одабирања за фактор M . Јасно је са Слике 2.10 да претходно свака полифазна компонента мора бити померена у временском домену за k одбирака у лево, па тек касније да јој буде снижена учестаност одабирања.



Слика 2.10 Секвенца $x[n]$ и три полифазне компоненте којима је снижена учестаност одабирања

Дакле, ако се полифазној компоненти помереној за k одбирака снизи учестаност одабирања за фактор M добија се

$$x_k[n] = x^{(p)}[nM + k]. \tag{2.28}$$

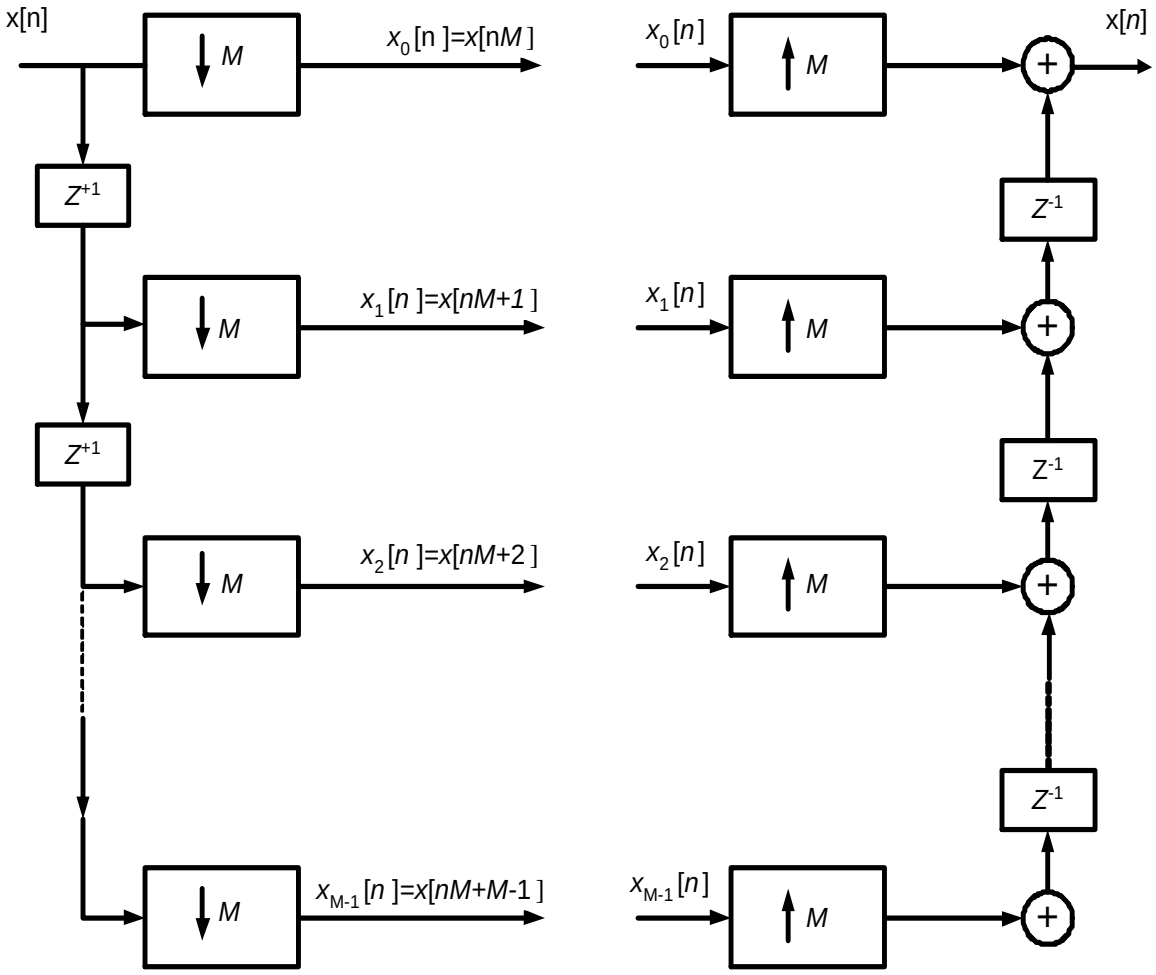
Учестаност одабирања полифазних компонената $x_k[n]$ је M пута мања од оригиналног сигнала $x[n]$. Како је $x_k[n]$ фазно померена верзија секвенце $x_k^{(p)}[n]$ и којој је снижена учестаност одабирања, њена Z-трансформација износи:

$$\{x_k[n]\} \leftarrow z \rightarrow X_k(z) = X_k^{(p)}(z), \qquad k = 0, 1, 2, \ldots, M-1. \tag{2.29}$$

тако да је *Z*-трансформација оригиналног сигнала $X(z)$у функцији *Z*-трансформације компонената $X_k(z), k = 0,1...M\text{-}1$

$$X(z) = \sum_{k=0}^{M-1} z^{-k} X_k\left(z^M\right) \tag{2.30}$$

Полифазна декомпозиција може бити изведена и на алтернативни начин, као што је то приказано на Слици 2.11. Одбирци оригиналног сигнала $x[n]$ се дистрибуирају између субсеквенци $\{x_k[n]\}, k = 0, 1, 2, \ldots, M\text{-}1$. Процедура реконструкције оригиналног сигнала се одвија на следећи начин: (1) *к*-тој компоненти се увећава учестаност одабирања за фактор M и онда се помера удесно за k одбирака, (2) овако добијени сигнали се сабирају и реконструише се оригинални сигнал $x(n)$.



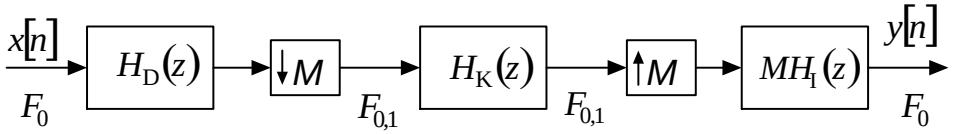
Слика 2.11 Блок дијаграм полифазне декомпозиције и реконструкције (а) декомпозиција (б) реконструкција

2.4 *MULTIRATE* ТЕХНИКЕ ЗА РЕАЛИЗАЦИЈУ ДИГИТАЛНИХ ФИЛТАРА

Реализација дигиталних филтара са стрмом карактеристиком у прелазној области је јако тешка а понекад и немогућа са конвенционалним филтрима. Проблем код једностепених *FIR* филтара је њихова сложеност, тј. ред филтра је обрнуто пропорционалан ширини прелазне зоне, тако да сложеност *FIR* филтара поприма јако велике размере код филтара са уском прелазном зоном. Код *IIR* филтара проблем је велика осетљивост на положај полова функције преноса. Из ових разлога се *multirate* приступ користи при пројектовању дигиталних филтара са стрмом карактеристиком у прелазној зони јер омогућава реализацију дигиталних филтара са истоветном улазном и излазном учестаношћу одабирања, а знатно нижег реда од еквивалентних једностепених филтара.

2.4.1 ВИШЕСТЕПЕНИ (*MULTISTAGE*) УСКОПОЈАСНИ ФИЛТРИ

Овај метод је погодан за примену на нископропусне, високопропусне и филтре пропуснике опсега код којих је ширина опсега мања од четвртине учестаности одабирања. Овакве филтре називамо ускопојасним филтрима. Генерална структура ових филтара је приказана на Слици 2.12



Слика 2.12 Структура вишестепеног филтра

Са Слике 2.12 се види да се вишестепени филтер састоји од кола за снижење учестаности одабирања, *kernel* филтра $H_K(z)$ и кола за увећање учестаности одабирања. Учестаност одабирања F_0 улазног сигнала $x[n]$ се прво смањује на нижу вредност $F_{0,1} = F_0/M$ тако да се филтрирање *kernel* филтром $H_K(z)$ сада врши на нижој учестаности, а затим се повратак на почетну учестаност одабирања F_0 врши колом за увећање учестаности одабирања..

Према томе, примена вишестепених филтара има следеће предности:

1 Захтеви и проблеми филтрирања могу бити подељени на неколико субфилтара нижег реда;
2 Утицај ефеката коначне дужине речи на перформансе укупног филтра је знатно смањен;
3 Аритметичке операције у субфилтрима се изводе на сниженој учестаности одабирања;

Са друге стране примена оваквог приступа проузрокује и низ нежељених ефеката који се морају узети у обзир и извршити њихова минимизација. Као што је речено у претходним поглављима, приликом промене учестаности одабирања јављају се следећи проблеми:

- Приликом сваког снижења учестаности одабирања за фактор M у спектру излазног сигнала ће се јавити сума од M униформно померених и продужених верзија спектра улазних сигнала скалираних за фактор $1/M$ тј доћи ће до појаве *aliasing*-а;
- Приликом сваког увећања учестаности одабирања за фактор L доћи ће до појаве L-1 реплика у фреквенцијском домену, тј. доћи ће до појаве *imaging*-а.

То значи да субфилтри вишестепеног филтра морају ефикасно да спрече поменуте појаве приликом промене учестаности одабирања како не би дошло до дисторзије сигнала. Функција преноса филтра са слике 2.12 износи:

$$H(z)=\frac{Y(z)}{X(z)} \qquad (2.31)$$

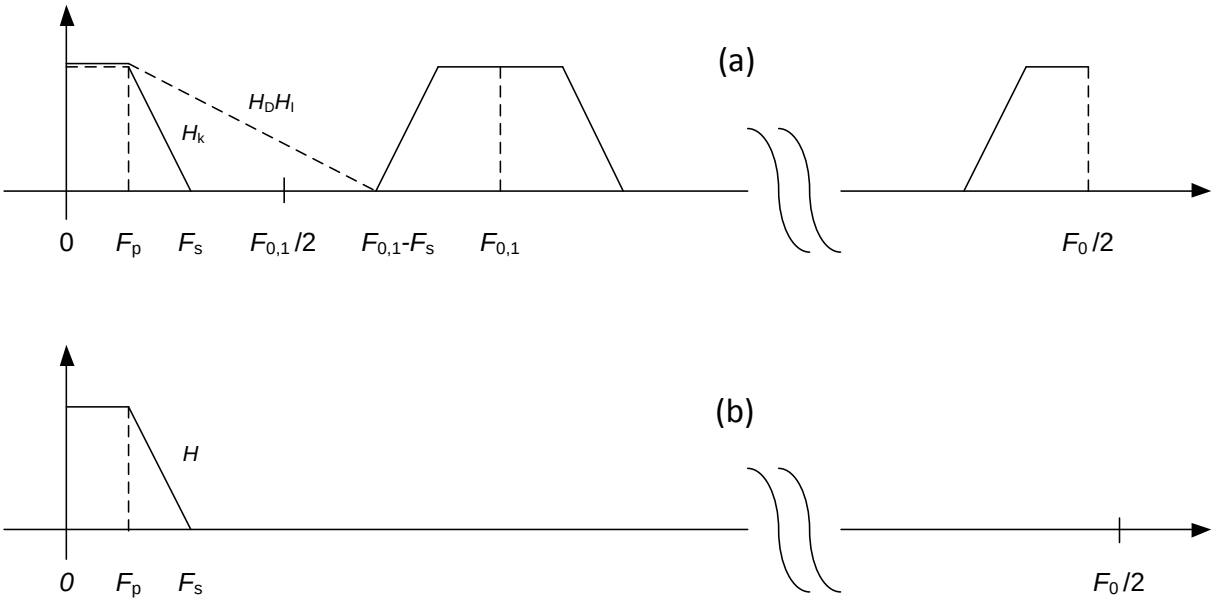
и састоји се из три члана; $H_D(z)$, $H_K(z)$ и $H_I(z)$. Под условом да филтри $H_D(z)$ и $H_I(z)$ елиминишу ефекте *aliasing*-а и *imaging*-а, свеобухватна функција преноса $H(z)$ би износила:

$$H(z)=H_D(z)H_K\left(z^M\right)H_I(z). \qquad (2.32)$$

Заменом $z=e^{j\omega}$ у једначину 2.32 добија се фреквенцијски одзив *multirate* филтра

$$H\left(e^{j\omega}\right)=H_D\left(e^{j\omega}\right)H_K\left(e^{j\omega}\right)H_I\left(e^{j\omega}\right) \tag{2.33}$$

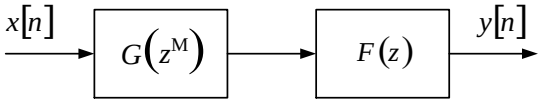
На Слици 2.13 је приказан амплитудски одзив филтра са Слике 2.12. Са слике се види да представљена веза децимационог, интерполационог и *kernel* филтра има ускопојасну карактеристику, док филтри могу бити реализовани са знатно мањим захтевима. Укупни амплитудски одзив представља производ амплитудских одзива $\left|H_K\left(e^{jM\omega}\right)\right|$, $\left|H_D\left(e^{j\omega}\right)\right|$, и $\left|H_I\left(e^{j\omega}\right)\right|$, тако да је укупно минимално слабљење у непропусном опсегу одређено слабљењима филтара $H_K\left(z^M\right), H_D(z)$ и $H_I(z)$. У опсегу $Fs \le F \le F_{0,1}\text{-}F_s$, минимално слабљење у непропусном опсегу је одређено једино слабљењем *кернел* филтра $H_K\left(z^M\right)$, док је за фреквенције веће од $F_{0,1}\text{-}F_s$, минимално слабљење у непропусном опсегу одређено каскадном везом децимационог и интерполационог филтра $H_D(z)$ и $H_I(z)$. Важно је нагласити то да је главна намена децимационог и интерполационог филтра да спрече појаву *aliasing*-а код снижавања учестаности одабирања и *imaging*-а код увећања учестаности одабирања, а не да обезбеђују слабљење у непропусном опсегу.



Слика 2.13 Амплитудски одзив (а) децимационог, интерполационог и кернел филтра (б) укупног филтра

2.4.2 РЕАЛИЗАЦИЈА УСКОПОЈАСНИХ ФИЛТАРА ТЕХНИКОМ ФРЕКВЕНЦИЈСКОГ МАСКИРАЊА

Принцип реализације ускопојасних филтара техником фреквенцијског маскирања (*frequency-response masking technique*) је приказан на Слици 2.14. Као што се са слике види принцип је веома једноставан: ускопојасни филтер је изведен као каскадна веза периодичног модел филтра $G\left(z^M\right)$ и маскирајућег филтра $F(z)$.

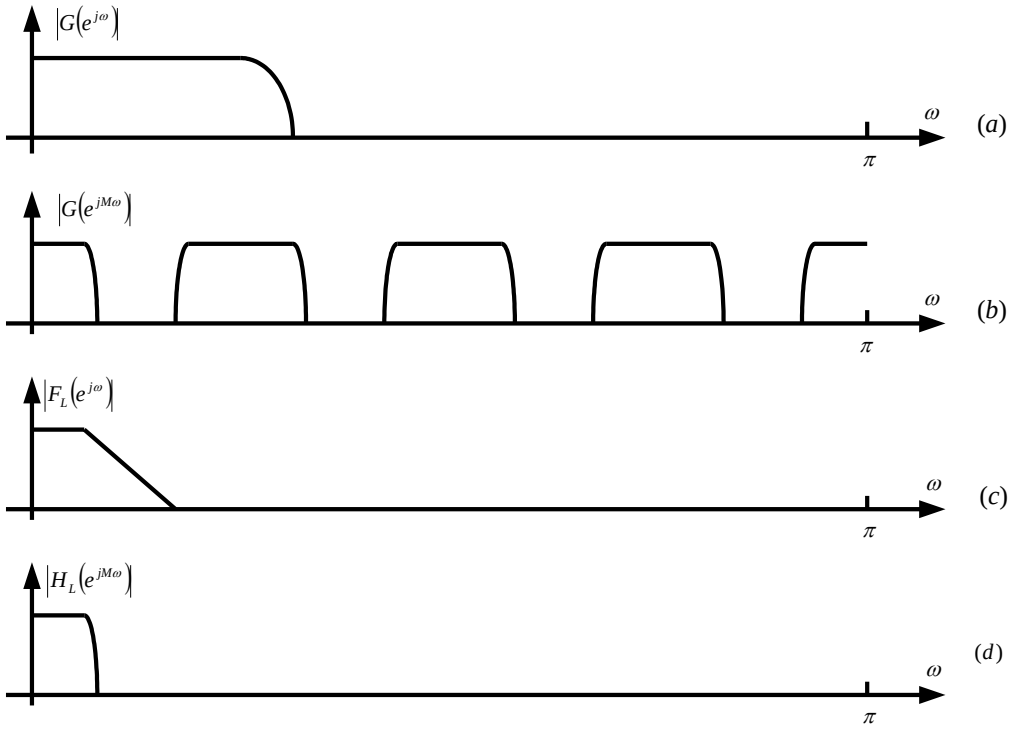


Слика 2.14 Каскадна веза периодичног модел филтра и маскирајућег филтра

Заменом сваког кашњења z^{-1} у модел филтру $G(z)$ чији је фреквенцијски одзив $G\left(e^{j\omega}\right)$ приказан на Слици 2.15.а, добија се периодични модел филтар $G\left(z^M\right)$. Фреквенцијски одзив периодичног модел филтра је приказан на Слици 2.15.б. Периодични одрази (*images*) проузроковани са $G\left(e^{jMw}\right)$ морају бити елиминисани маскирајућим филтром. За реализацију нископропусног филтра, маскирајући филтер би требао бити нископропусни, а представљен је са $F_L\left(e^{jw}\right)$ на Слици 2.15.ц. Каскадна везе периодичног модел филтра $G\left(e^{jMw}\right)$ и маскирајућег филтра $F_L\left(e^{jw}\right)$ чини ускопојасни нископропусни филтер $H_L\left(e^{jw}\right)$ приказан на слици 2.15.д. Код реализације филтара пропусника опсега, маскирајући филтер би требао бити филтер пропусник опсега, а уколико се захтева високопропусни филтер, тада једино филтер $F(z)$ мора бити високопропусни. На основу реченог, за ускопојасне филтре, функцију преноса $H(z)$ можемо да изразимо на следећи начин:

$$H(z)=G\left(z^M\right)F(z), \tag{2.34}$$

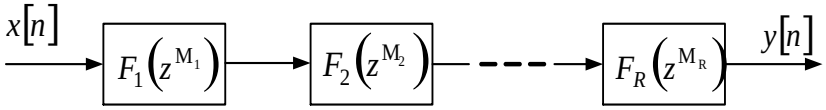
при чему се за $G(z)$ и $F(z)$ произвољно може изабрати *FIR* или *IIR* функција преноса.



Слика 2.15 Принцип технике фреквенцијског маскирања. Реализација ускопојасних филтара

Велика предност оваквог приступа при реализацији дигиталних филтара је то што је прелазна зона укупног филтра M пута мања од прелазне зоне модел филтра. Овакав однос је резултат замене сваког кашњења z^{-1} у $G(z)$ са M кашњења. Из овог разлога је и пропусни опсег такође снижен за исти фактор, тако да је овакав метод применљив само за ускопојасне филтре.

Поступак фреквенцијског маскирања примењује се за извођење ускопојасних филтара са стрмом карактеристиком уз коришћење филтара са знатно широм прелазном зоном. Спецификације за маскирајући филтер могу додатно бити упрошћене коришћењем вишестепене (енгл. *multistage*) имплементације функције $F(z)$ као што је то приказано на Слици 2.16.



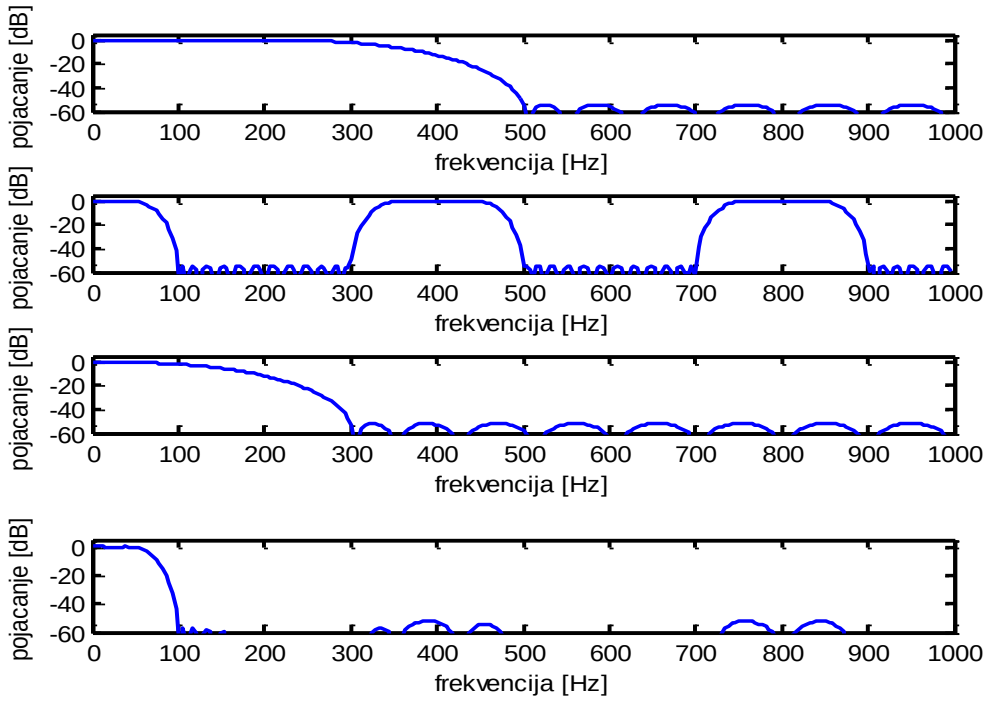
Слика 2.16 Вишестепена имплементација маскирајућег филтра

Укупна функција преноса структуре са слике 2.16 је дата са

$$F(z) = \prod_{k=1}^{R} F_k\left(z^{M_k}\right) \tag{2.35}$$

Периодични филтри $F_1, F_2, ...F_R$ са слике 2.16 су пројектовани тако да уклањају периодичне одразе (*images*) из фреквенцијског одзива посматране структуре.

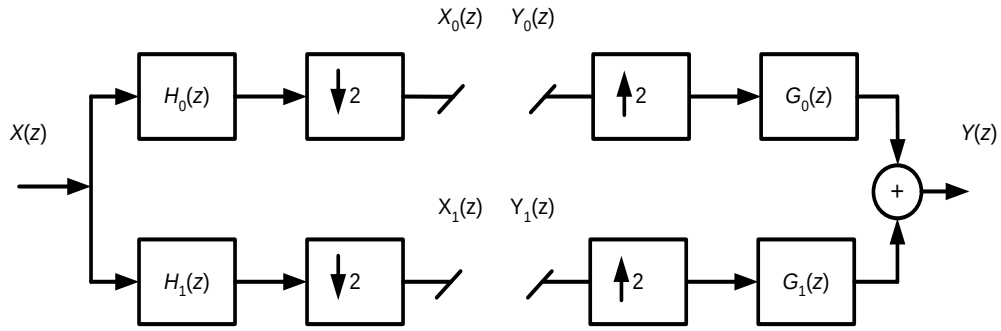
Једна реализација ускопојасног филтра техником фреквенцијског маскирања приказана је на слици 2.17 за $F_p = 50Hz$, $F_s = 100\ Hz$, $F_0 = 2\ 000\ Hz$ и $M = 5$.



Слика 2.17 Ускопојасни *FIR* филтер: појачање модел филтра, периодичног модел филтра, маскирајућег филтра и ускопојасног филтра

2.5 QMF BANKE

Двоканалне *QMF* банке (енгл. *Quadrature mirror filter – QMF*) представљају један од основних елемената у вишебрзинској обради сигнала. Основна улога *QMF* банки је да изврше поделу на подопсеге (типично, опсег високих фреквенција и опсег ниских фреквенција) а затим се врши реконструкција сигнала из подопсега. Врста и начин процесирања сигнала између секција за анализу и секција за синтезу могу бити различити, зависно од области у којој се примењују *QMF* банке [3,4]. Блок шема двоканалне *QMF* банке је приказана на Слици 2.18.



Слика 2.18 Блок дијаграм *QMF* банке

QMF банке се састоје из дела за анализу који садржи НФ филтер $H_0(z)$, ВФ филтер $H_1(z)$ и пар дециматора, док део за синтезу садржи НФ филтер $G_0(z)$, ВФ филтер $G_1(z)$ и пар интерполатора. Због процеса децимације, доћи ће до преклапања у фреквенцијском домену јер филтри нису идеални. Предност *QMF* приступа је у томе што филтри за синтезу могу бити изабрани тако да се у процесу реконструкције неутрализују преклапања настала у делу за анализу.

Ако са $X_0(z)$ и $X_1(z)$ означимо излазе из кола за снижење учестаности, према Слици 2.18 важиће:

$$X_0(z)=\frac{1}{2}\left[X(z^{1/2})\cdot H_0(z^{1/2})+X(-z^{1/2})\cdot H_0(-z^{1/2})\right] \tag{2.36}$$

$$X_1(z)=\frac{1}{2}\left[X(z^{1/2})\cdot H_1(z^{1/2})+X(-z^{1/2})\cdot H_1(-z^{1/2})\right] \tag{2.37}$$

Ако се са $Y_0(z)$ и $Y_1(z)$ означе улази у интерполаторе и ако се претпостави да су $Y_0(z)=X_0(z)$ и $Y_1(z)=X_1(z)$, излаз из *QMF* банке ће бити:

$$Y(z)=G_0(z)\cdot Y_0(z^z)+G_1(z)\cdot Y_1(z^z) \tag{2.38}$$

Заменом израза (2.36) и (2.37) у (2.38) добија се:

$$\begin{aligned}Y(z)=T(z)+B(z)=&\frac{1}{2}\left[H_0(z)\cdot G_0(z)+H_1(z)\cdot G_1(z)\right]\cdot X(z)+\\&\frac{1}{2}\left[H_0(-z)\cdot G_0(z)+H_1(-z)\cdot G_1(z)\right]\cdot X(-z)\end{aligned} \tag{2.39}$$

Први део израза (2.39) $T(z)$ представља жељени излаз из *QMF* банке, док је други део $B(z)$ компонента која се односи на нежељено преклапање, тако да би требало бити:

$$H_0(-z)\cdot G_0(z)+H_1(-z)\cdot G_1(z)=0, \tag{2.40}$$

што ће важити ако је

$$G_0(z)=H_1(-z) \tag{2.41}$$

и

$$G_0(z)=H_1(-z). \tag{2.42}$$

Ако су испуњени претходни услови, излаз из *QMF* банке ће бити:

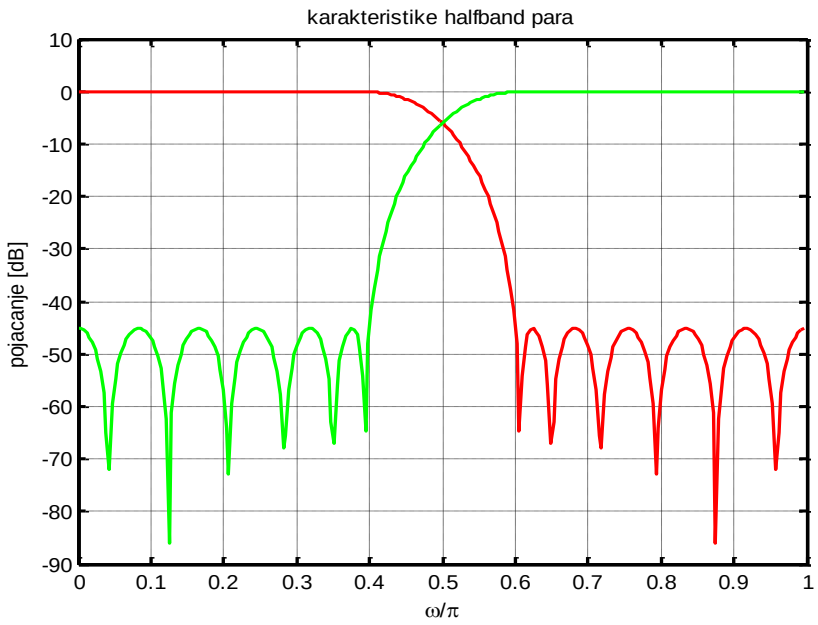
$$Y(z)=\frac{1}{2}\left[H_0(z)\cdot H_1(-z)-H_1(z)\cdot H_0(-z)\right]\cdot X(z) \tag{2.43}$$

а функција преноса *QMF* банке:

$$T(z)=\frac{1}{2}\left[H_0(z)\cdot H_1(-z)-H_1(z)\cdot H_0(-z)\right]. \tag{2.44}$$

Филтри $H_0(z)$ и $H_1(z)$ одређују тип подопсега на које се врши расподела. Најчешће је $H_0(z)$ *NF* филтер а $H_1(z)$ његов "лик у огледалу" у односу на $\pi/2$. На Слици 2.19 је приказана карактеристика пара оваквих филтара (одатле је и израз "огледало" у називу банке) које називамо и *half-band* или полуопсежни филтри. Да би преклапање и скалирање са ½ које настаје у процесу децимације било елиминисано, филтри за синтезу би требали бити на основу (2.43) и (2.44):

$$\begin{aligned} G_0(z)&=2H_0(z)\\ G_1(z)&=-2H_0(-z)\end{aligned} \tag{2.45}$$



Слика 2.19 Амплитудска карактеристика пара *half-band* филтара

Уколико су филтри $G_0(z)$ и $G_1(z)$ изабрани на овакав начин, биће елиминисани ефекати преклапања у процесу децимације, тј. израз за $B(z)$ у једначини (2.39) се избором филтара за синтезу према (2.45) своди на нулу. Заменом (2.45) у (2.38) добија се:

$$Y(z) = \left[H_0^2(z) - H_0^2(-z)\right] \cdot X(z), \tag{2.46}$$

односно функција преноса *QMF* банке је дата са:

$$T(z) = \left[H_0^2(z) - H_0^2(-z)\right]. \tag{2.47}$$

За савршену реконструкцију сигнала потребно је још да функција преноса *QMF* банке има облик:

$$T(z) = c \cdot z^{-n_0} \tag{2.48}$$

тј. да је функција преноса "чисто" кашњење.

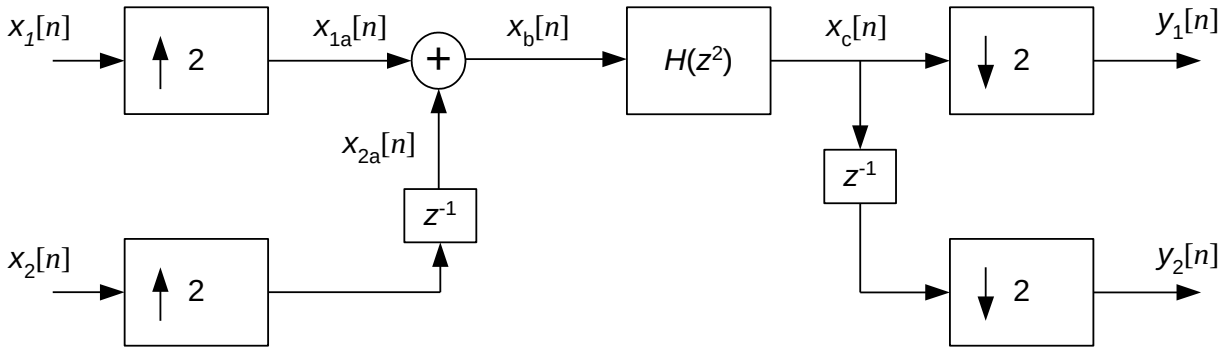
3. ПРОТОЧНА ОБРАДА СИГНАЛА

Дигитално филтрирање добија све значајнију улогу у модерним информационо-комуникационим системима. Брзина процесирања се разликује од случаја до случаја и креће се у врло великом распону, тако да се због сложености савремених апликација захтева све више и више хардверских ресурса. Из ових разлога јавља се потреба за преуређењем структура дигиталног филтрирања, како би се задовољили захтеви у смислу функционалности уз што мању цену.

Поступак проточне обраде сигнала (*енгл. Pipelining-Interleaving*) или скраћено *PI* поступак је развијен са циљем да се побољша ефикасност дигиталног филтрирања. Користи се у случајевима када је потребно процесирати два и више сигнала филтрима истих карактеристика [5,6]. Он омогућава да се увећањем учестаности одабирања K пута и заменом сваког кашњења у филтру $\left(z^{-1}\right)$ са K кашњења $\left(z^{-K}\right)$ користи један филтер за све канале.

3.1 ПРИНЦИП *PIPELINING-INTERLEAVING* ТЕХНИКЕ

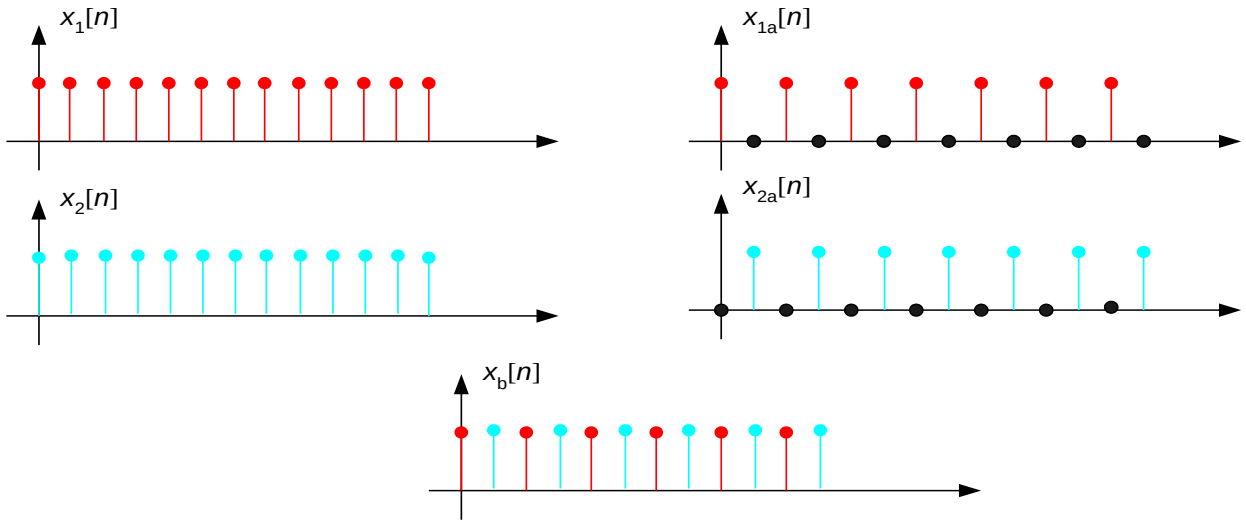
Принцип поступка проточне обраде сигнала приказан је на Слици 3.1 за случај процесирања два канала.



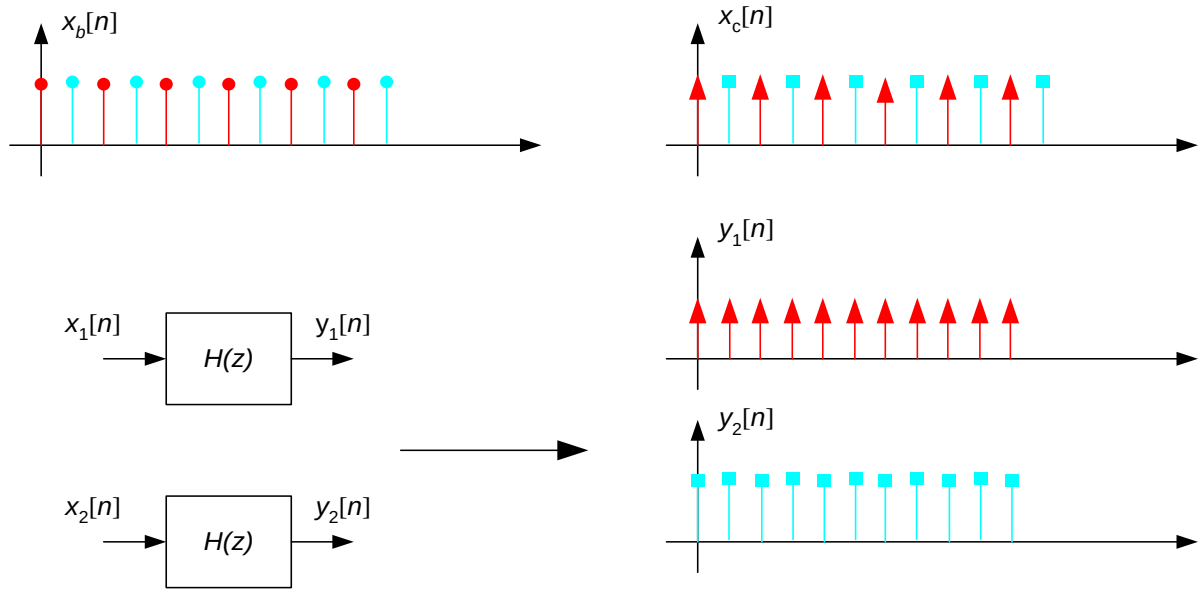
Слика 3.1 *Pipelining-Interleaving* принцип

Улазни сигнали $x_1[n]$ и $x_2[n]$ се воде на кола за увећање учестаности одабирања. Сигнал $x_2[n]$ се води на коло за кашњење, а затим се овако закашњени сигнал $x_{2a}[n]$ сабира са сигналом $x_{1a}[n]$ и тако формира сигнал $x_b[n]$ као на Слици 3.2. После кола за сабирање добија се ефекат као да су дискретни сигнали $x_1[n]$ и $x_2[n]$ временски

мултиплексирани. Овакав сложени сигнал води се на филтер $H(z^2)$, тј. на филтер $H(z)$ коме је свако кашњење (z^{-1}) замењено са два кашњења (z^{-2}) (у случају K канала са K кашњења z^{-K}). Излазни сигнал из оваквог филтра ће бити идентичан оном као да су мултиплексирани излази из оригиналног филтра $H(z)$, као што је приказано на Слици 3.3.



Slika 3.2 Мултиплексирање улазних сигнала



Слика 3.3 Излаз из филтра $H(z^2)$

Сада је потребно само да се раздвоје сигнали по каналима и процес филтрирања два канала једним филтром је завршен. Раздвајање канала врши се колом за снижење учестаности одабирања, а сигнал из другог канала се помера за један одбирак колом за кашњење $\left(z^{-1}\right)$, тако да су излази из *PI* структуре идентични онима који би били добијени засебним филтрирањем.

Претходно разматрање се може доказати и аналитички у временском домену. Наиме, излазни сигнал из филтра $H\left(z^2\right)$ ће бити једнак конволуцији улазног сигнала у проширени филтер $x_b[n]$ и импулсног одзива оваквог филтра $h_z[n]$. Улазни сигнал $x_b[n]$ проширеног филтра може бити представљен на следећи начин:

$$x_b = [x_b[0], x_b[1], x_b[2].....x_b(2k-1)] = [x_1[0], x_2[0], x_1[1], x_2[1].....x_1[k-1], x_2[k-1]], \quad (3.1)$$

Импулсни одзив проширеног филтра $H\left(z^2\right)$ ће бити:

$$h_z = [h[0],0, h[1],0, h[2],0.....h[N-1],0], \quad (3.2)$$

јер је импулсни одзив филтра $H(z)$:

$$h = [h[0], h[1], h[2],.....h[N-1],]. \quad (3.3)$$

На основу претходног ће излазни сигнал из филтра $H\left(z^2\right)$ бити:

$$\begin{aligned} x_c[n] = \sum_{k=-\infty}^{\infty} h_z[k] x_b[n-k] = [x_1[1]\cdot h[1], x_1[1]\cdot 0 \ + \ x_2[1]\cdot h[1], x_1[1]\cdot h[3] \ + \\ + x_2[1]\cdot 0 \ + x_1[2]\cdot h[2], x_1[1]\cdot 0 \ + \ x_2[1]\cdot h[3] \ + \ x_1[2]\cdot 0 \ + \ x_2[2]\cdot h[2].....] \end{aligned}, \quad (3.4)$$

или после сређивања:

$$\begin{aligned} x_c[n] = [x_1[1]\cdot h[1], x_2[1]\cdot h[1], x_1[1]\cdot h[3] \ + \\ x_1[2]\cdot h[2], x_2[1]\cdot h[3] \ + \ x_2[2]\cdot h[2],.....] \end{aligned}. \quad (3.5)$$

Уколико би били раздвојени одбирци сигнала $x_c[n]$ тако да је формиран један сигнал који садржи само непарне одбирке (први, трећи пети...) и други сигнал који садржи само парне одбирке (други, четврту, шести...) тј.

$$\begin{aligned} y_1[n] &= [x_1[1]\cdot h[1],\ x_1[1]\cdot h[3] + x_1[2]\cdot h[2], \ldots\ldots\ldots\ldots] \\ y_2[n] &= [x_2[1]\cdot h[1], x_2[1]\cdot h[3] + x_2[2]\cdot h[2], \ldots\ldots\ldots\ldots] \end{aligned}, \tag{3.6}$$

добио би се исти ефекат као да су сигнали $x_1[n]$ и $x_2[n]$ филтрирани посебно филтром чија је функција преноса $H(z)$. Претходно разматрање доказује поступак проточне обраде сигнала у временском домену.

3.2 *PI* ПОСТУПАК У ФРЕКВЕНЦИЈСКОМ ДОМЕНУ

Поступак проточне обраде сигнала може бити верификован и у фреквенцијском домену. Сигнали на улазу у коло за сабирање (Слике 3.1) ће бити:

$$\begin{aligned} X_{1a}(z) &= X_1(z^2) \\ X_{2a}(z) &= z^{-1}\cdot X_2(z^2) \end{aligned}, \tag{3.7}$$

тако да ће сигнали на улазу и на излазу из филтра $H(z^2)$ бити одређени са:

$$\begin{aligned} X_b(z) &= X_1(z^2) + z^{-1}X_2(z^2) \\ X_c(z) &= H(z^2)\cdot\left[X_1(z^2) + z^{-1}X_2(z^2)\right] \end{aligned}. \tag{3.8}$$

Излази из *PI* структуре ће бити:

$$\begin{aligned} Y_1(z) &= \frac{1}{2}\sum_{k=0}^{1} X_c\left(z^{\frac{1}{2}}\cdot e^{j\frac{2\cdot\pi}{2}k}\right) = \frac{1}{2}\left[X_c\left(z^{\frac{1}{2}}\right) + X_c\left(-z^{\frac{1}{2}}\right)\right] \\ Y_2(z) &= \frac{1}{2}\sum_{k=0}^{1} z^{-\frac{1}{2}}\cdot X`_c\left(z^{\frac{1}{2}}\cdot e^{j\frac{2\cdot\pi}{2}k}\right) = \frac{1}{2}\left[X`_c\left(z^{\frac{1}{2}}\right) + X`_c\left(-z^{\frac{1}{2}}\right)\right] \end{aligned}. \tag{3.9}$$

где је

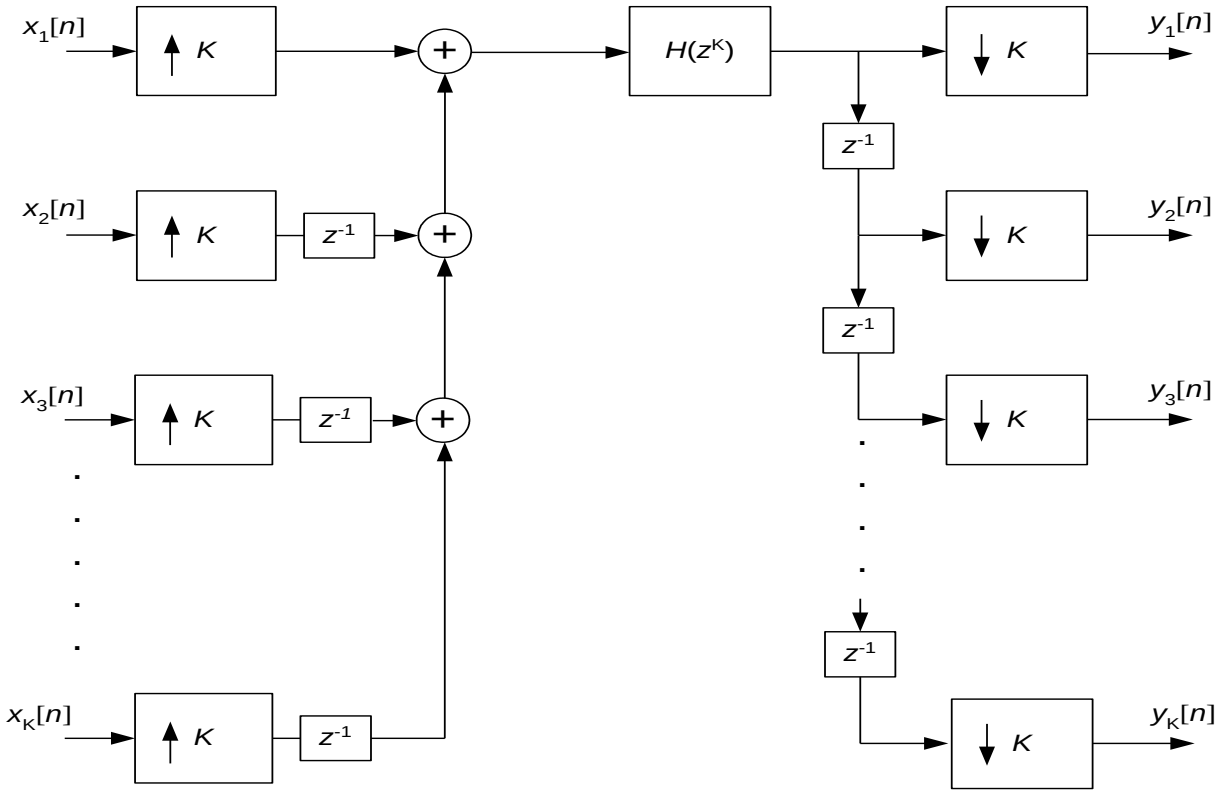
$$X`_c(z) = z^{-1} \cdot X_c(z). \tag{3.10}$$

На основу предходног ће важити:

$$\begin{aligned} Y_1(z) &= H(z) \cdot X_1(z) \\ Y_2(z) &= z^{-1} \cdot H(z) \cdot X_2(z) \end{aligned} \tag{3.11}$$

што значи да се добијају исте вредности као да су канали филтрирани понаособ истим филтром $H(z)$.

Из претходног разматрања следи веома важан закључак да при примени овакве методе нема дегенерације сигнала у смислу *aliasing*-а и *imaging*-а и да нису потребне додатне радње за ограничавање спектра из тог разлога што су излазни сигнали из *PI* структуре потпуно идентични онима које би добили засебним филтрирањем (нема избацивања одбирака или додавања одбирака чија је вредност нула као код увећања или снижавања учестаности одабирања). На исти начин би било показано да *PI* принцип важи и за произвољан број канала. *PI* поступак примењен на K канала је приказан на слици 3.4.



Слика 3.4 *PI* поступак за *K* канала

У овом раду ће бити извршена провера *PI* поступка у реалним условима, тј. и за *FIR* и за *IIR* филтре и то посматрајући процес филтрирања одбирак по одбирак, елеменат по елеменат, како би било утврђено под којим условима је могуће и са каквом тачношћу реализовати предложену структуру.

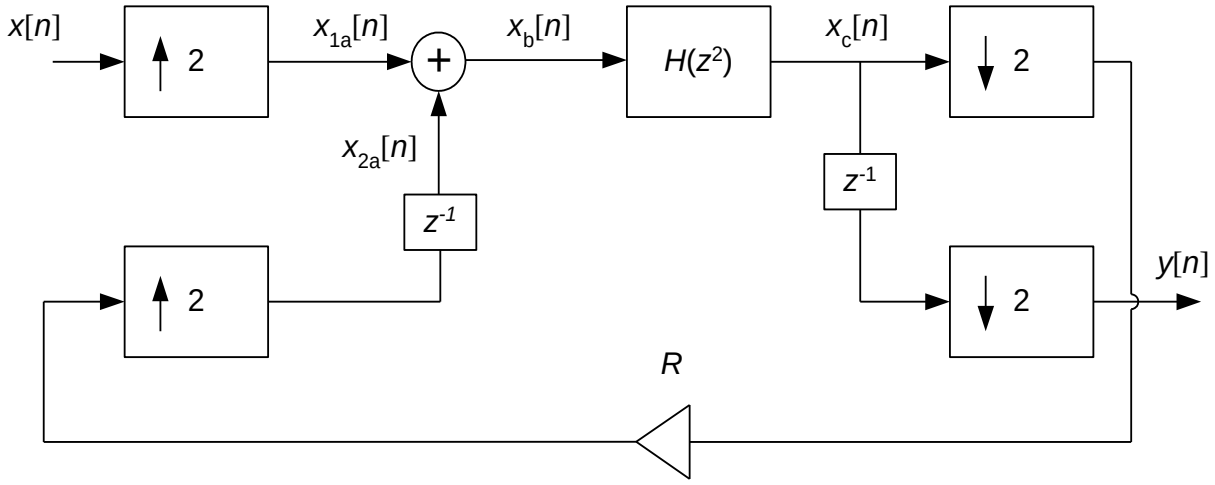
3.3 МОДИФИКАЦИЈА *PI* ПОСТУПКА

Уколико се излазни сигнал једног канала искористи као улаз у други канал, тако да се добије ефекат као да је сваки одбирак улазног сигнала два пута процесиран кроз предложену структуру (Слика 3.5), биће остварена следећа функција преноса:

$$H_N(z) = z^{-1} \cdot H^2(z) \tag{3.12}$$

Ако би се даље у повратној грани извршило множење са неком константом R, и ако би се користио излаз из првог канала, комбиновањем ова два излаза могла би бити реализована следећа функција преноса:

$$H_N(z) = z^{-1} \cdot R \cdot H^2(z) + H(z). \tag{3.13}$$



Слика 3.5 Модификација *PI* поступка

Проширујући примену PI поступка на структуру K-тог реда могла би се релизовати функција преноса типа полинома K-тог реда:

$$H_N(z) = R_K \cdot H^K(z) + R_{K-1} \cdot H^{K-1}(z) + ... + H(z), \tag{3.14}$$

где би избором коефицијената $R_K .....R_2$ била одређена оптимална функција преноса.

Међутим, потребно је верификовати реализацију предложеног поступка у пракси. Наиме мора се водити рачуна о утицају нових повратних спрега и кашњења, и зависно од начина на који је изведен филтер $H(z)$ потребно је извршити проверу предложене структуре у реалном времену, што ће бити урађено у овом раду.

4. РЕАЛИЗАЦИЈА ОСНОВНИХ *MULTIRATE* ДИГИТАЛНИХ ФИЛТАРА

У претходним поглављима су представљени основни принципи технике проточне обраде сигнала. Показано је како би се оваква техника могла применити на реализацију дигиталних филтара као и на реализацију структура које представљају полиноме функције преноса филтра $H(z)$. Потребно је сада размотрити могућности оваквих реализација у пракси за сваки случај понаособ. Наиме, применом технике проточне обраде сигнала врши се промена структуре дигиталних филтара (тј. група филтара), тако да би за сваки конкретан случај било неопходно утврдити услове под којима је таква примена могућа као и одредити оптималне реализације у смислу одступања амплитудске карактеристике, утицају ефекта коначне дужине кодне речи, о утицају увођења нових повратних веза и слично.

4.1 *FIR PI* СТРУКТУРА

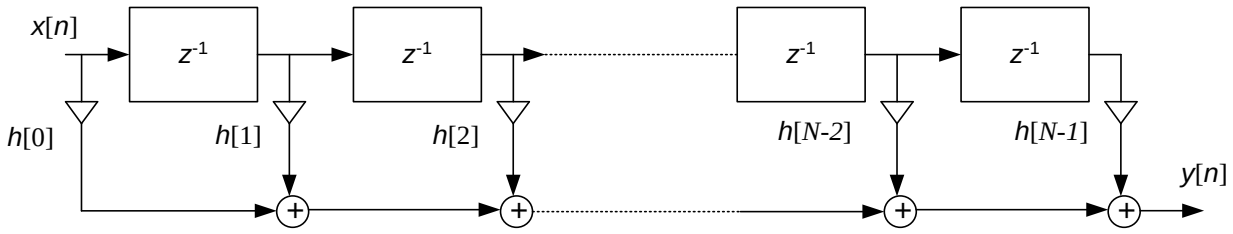
Поступак проточне обраде сигнала може се директно применити на реализацију *FIR* дигиталних филтара. Разлог овоме је структура *FIR* филтара. Као што је познато, улазно-излазна једначина за овај тип филтра дата је нерекурзивном диференцном једначином,

$$y(n)=\sum_{k=0}^{N-1}h(k)x(n-k), \tag{4.1}$$

а функција преноса *FIR* филтра се изражава у облику полинома по z^{-1}

$$H(z)=\frac{Y(z)}{X(z)}=\sum_{k=0}^{N-1}h(k)z^{-k}. \tag{4.2}$$

На основу једначине (4.2) следи директна реализација *FIR* филтара чији је блок дијаграм приказан на Слици 4.1 (трансферзални филтар).



Слика 4.1 Директна структура *FIR* филтра

Уколико се на овакав филтер примени *PI* поступак са Слике 3.5 добија се нова структура која садржи само једну повратну везу и за коју је потребно дефинисати услове како би имплементирала захтевану функцију. Ови услови би били следећи:

- Сва кола морају да раде на истом такту;
- Сва почетна стања треба да буду на нули како би се први улазни одбирак сабрао са нулом из кола за кашњење и све тако редом синхронизовано до последњег одбирка;
- Време процесирања у петљи мора бити краће од дужине такта како би коло за кашњење проследило одговарајући одбирак или нулу;
- Филтер $H\left(z^2\right)$ мора да "зна" дужину улазног сигнала, и по истеку последњег одбирка треба да престане са радом, јер би у супротном наставио да генерише одбирке до бесконачности због затворене повратне петље.

За овако дефинисане услове биће извршено снимање карактеристика нове структуре како би била утврђена величина одступања од оригиналне карактеристике и био проверен утицај грешке заокруживања, на основу чега би могао да се донесе закључак о могућностима овакве примене.

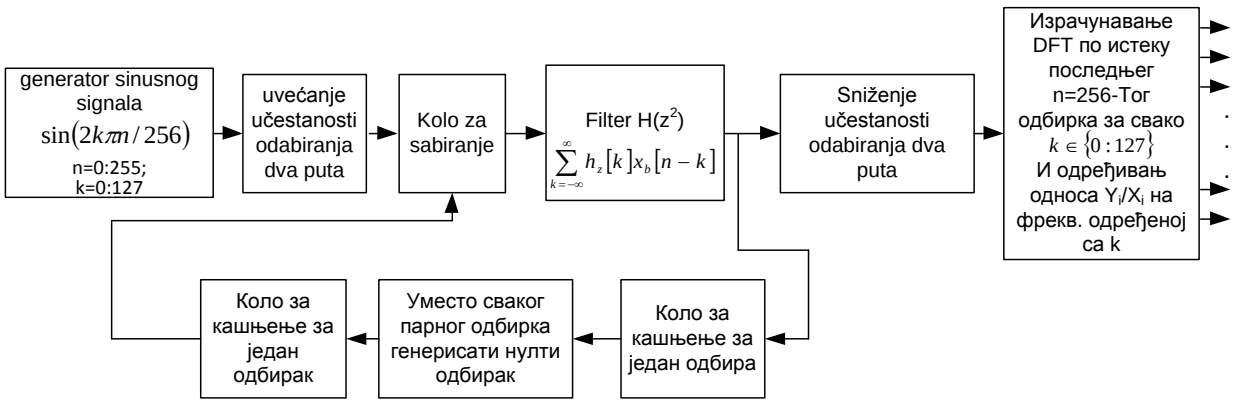
Провера структуре са Слике 3.5 проточне обраде сигнала за два канала код реализације каскадне везе два филтра у реалном времену биће извршена снимањем функције преноса одбирак по одбирак. На улаз структуре биће доведен синусни сигнал дужине 1024 одбирака чија се фреквенција мења у опсегу од нула до π у корацима од по 256 фреквенција. За сваку фреквенцију биће израчуната вредност излазног сигнала $y[n]$ и вредност *FFT*. Како би биле избегнуте разлике које се јављају услед прелазног периода, (битан је једино статички режим) за одређивање *FFT* ће се користити 512 одбирака од укупног броја и то из средине опсега. Однос *FFT* излазног и улазног сигнала на фреквенцији која је изабрана за посматрање ће одређивати тачку по тачку функције преноса. На основу 256 тачака (на колико је фреквенција вршено снимање) биће одређена

функција преноса посматране структуре. Уколико је предложени поступак исправан, овако одређена функција преноса би требало да буде идентична са функцијом преноса каскадне везе два филтра $H^2(z)$. Поред овакве провере у фреквенцијском домену, биће извршено упоређивање обе структуре и у временском домену посматрањем излаза на свим фреквенцијама улазног сигнала.

Приликом симулације предложеног модела елементи ће бити дефинисани на следећи начин:

- Коло за кашњење ће бити реализовано на начин како се и дефинише – са првим тактом генерише нулу, а затим прослеђује улазне одбирке померене за једну периоду одмеравања;
- Коло за увећање учестаности одабирања – са првим тактом прослеђује први одбирак, затим генерише нулу, затим генерише други одбирак и тако редом;
- Коло за снижење учестаности одабирања – са првим тактом прослеђује први улазни одбирак, затим изоставља следећи одбирак (сматра се да је у том тренутку сигнал на његовом излазу једнак нули), затим прослеђује трећи одбирак и тако редом;
- Филтрирање са $H(z^2)$ – биће реализовано тако што ће излаз из њега представљати конволуцију улазног сигнала и импулсног одзива проширеног филтра. Резултати добијени на овај начин ће се разликовати од резултата добијених наредбом *FILTER* јер ће филтер у нашем случају наставити да генерише одбирке и по истеку његовог улазног сигнала (њих *R-1* где је *R*-ред филтра).

Блок шема симулације описаног поступка за случај снимања амплитудске карактеристике каскадне везе два филтра са Слике 3.5 је приказана на Слици 4.2



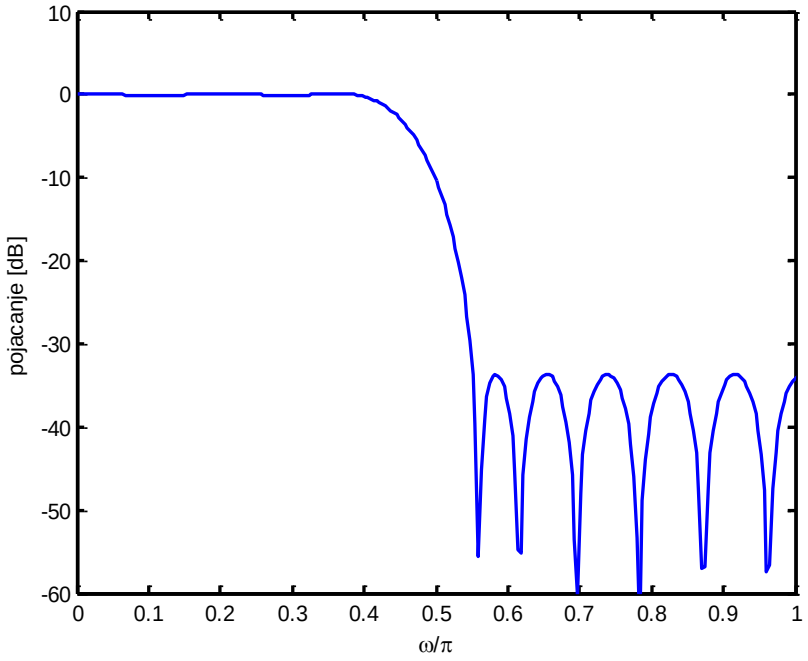
Слика 4.2 Блок шема снимања карактеристика *FIR PI* структуре за реализацију каскадне везе два филтра

За филтер $H(z)$ је изабран *NF FIR* филтер линеарне фазе са спецификацијама:

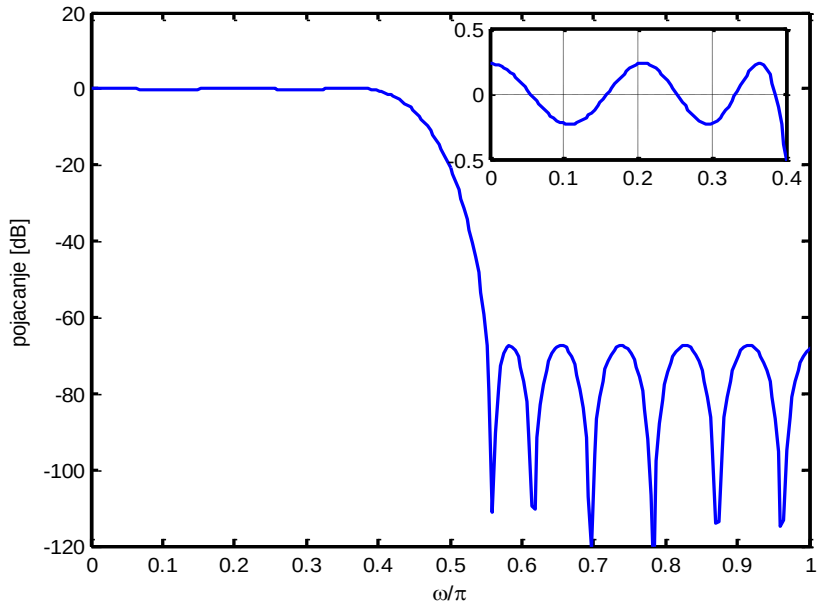
```
aa=35; (минимално слабљење у непропусном опсегу)
ap=0.2; (максимално слабљење у пропусном опсегу)
dev=[(10^(ap/20)-1)/(10^(ap/20)+1),10^(-aa/20)];
[Nf,fp,mag,wt]=remezord([50 70],[1 0],dev,256);
b1=remez(Nf,fp,mag,wt);
```

и при томе је добијено да је ред филтра $N_f = 22$.

Карактеристика овог филтра приказана је на Слици 4.3, а карактеристика каскадне везе два оваква филтра, тј. карактеристика какву би требало добити уколико је предложени поступак исправан, је приказана на Слици 4.4.

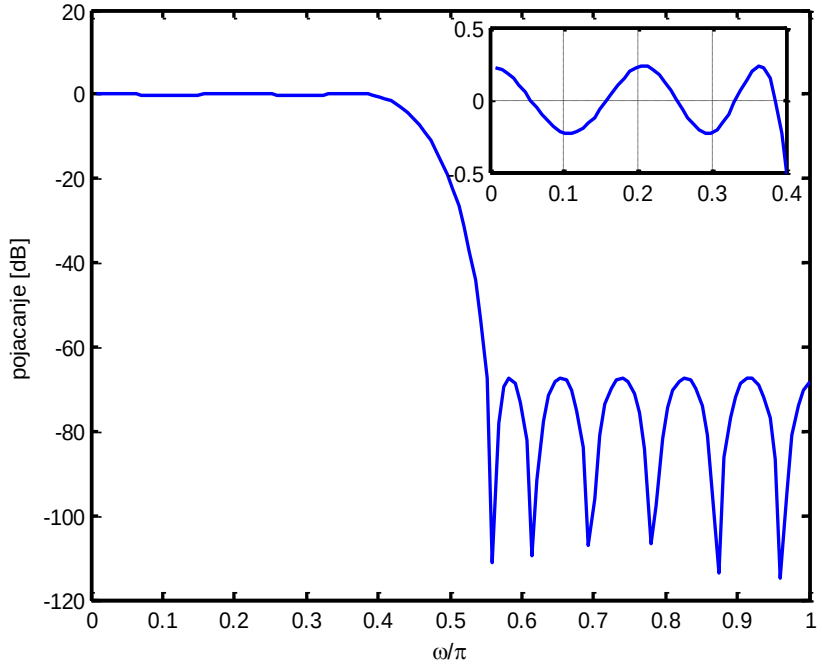


Слика 4.3 Амплитудска карактеристика филтра $H(z)$



Слика 4.4 Амплитудска карактеристика каскадне везе два филтра $H^2(z)$

Погледајмо сада како ће изгледати амплитудска карактеристика каскадне везе два филтра $H(z)$ ако је реализована по алгоритму са Слике 4.2. После извршене симулације добијена је карактеристика која је приказана на Слици 4.5.



Слика 4.5 Амплитудска карактеристика каскадне везе два филтра реализована према *PI* поступку и снимана одбирак по одбирак

На основу израчунатих карактеристика за каскадну везу два филтра $H(z)$ могуће је да се одреди и карактеристика сигнала грешке у временском и фреквенцијском домену, тј. могла би да се одреди функција која би представљала график шума који настаје због примене поступка проточне обраде сигнала. Поступак одређивања карактеристике шума би се састојао у следећем:

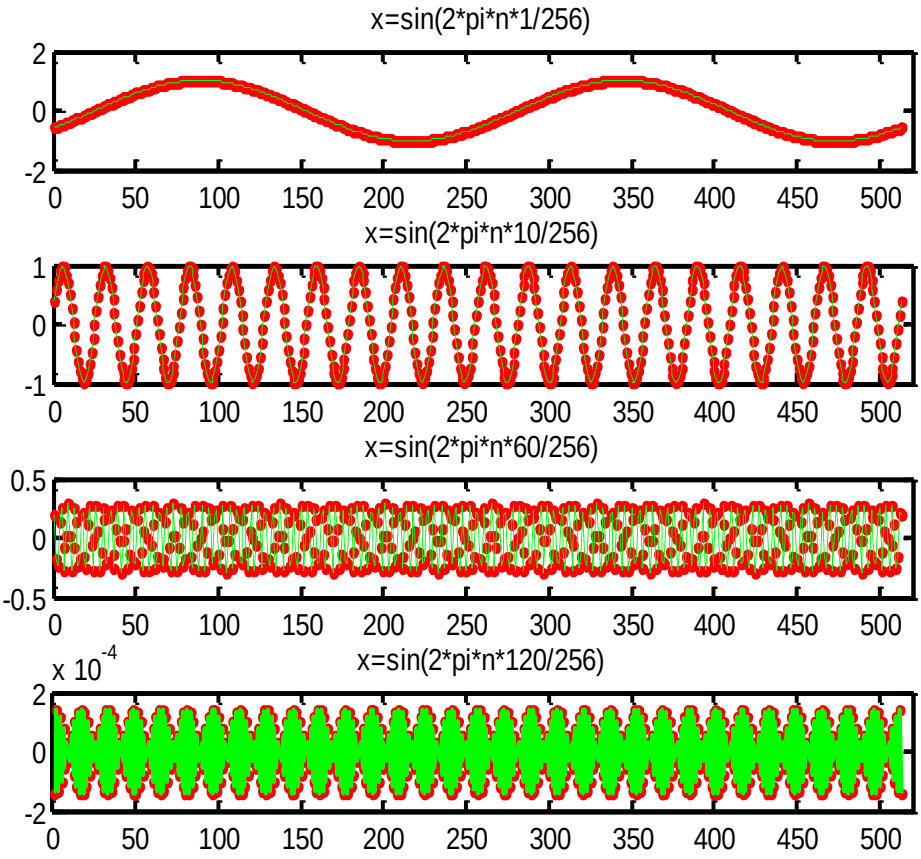
- За сваку фреквенцију из опсега од 0 до π биће израчуната разлика између сваког од 256 излазних одбирака из нове структуре и излаза из филтра $H^2(z)$;
- Одредиће се максимална вредност ове разлике између 256 одбирака за поједине фреквенције;
- На основу претходног биће формиран сигнал максималне разлике у временском домену и на основу њега израчуната амплитудска карактеристика.

Ако је дефинисана максимална грешка у временском домену као максимална разлика између вредности сваког одбирка добијеног на оба начина за све фреквенције након примене предложеног алгоритма добија се:

$E_{\max} = 0$.

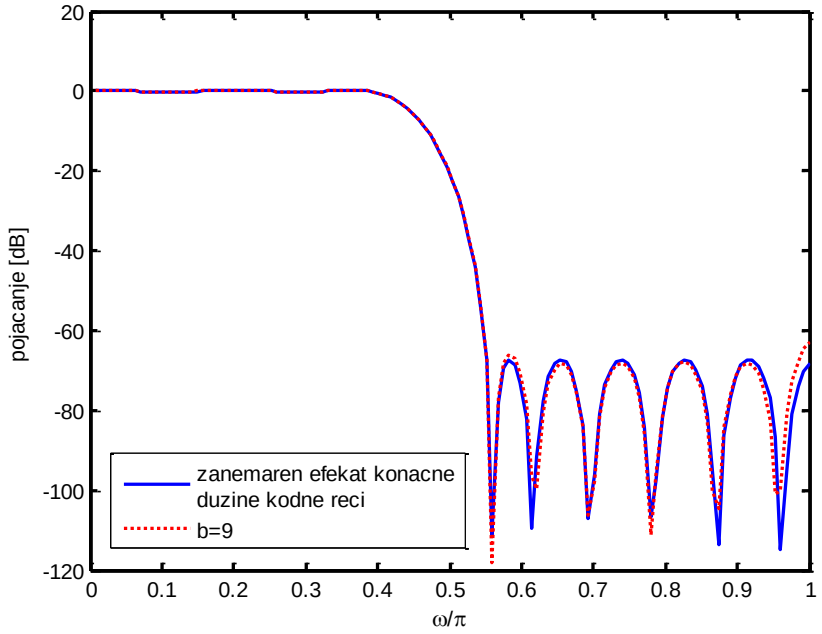
То значи да су излазни сигнали из обе структуре потпуно идентични и да практично нема никакве разлике уколико се каскадна веза два *FIR* филтра реализује применом *PI* технике. Из тог разлога је непотребно приказати график шума услед примене *PI* поступка у овом случају јер је вредност максималне грешке у целом испитиваном опсегу једнака нули. Овакав закључак потврђује и график са Слике 4.6 на ком су упоредно приказани излази из структуре са Слике 3.5 и излази из филтра $H^2(z)$ на неколико фреквенција.

Имајући ово на уму може бити потвђено да су разлике (видљиве су у непропусном опсегу) између амплитудских карактеристика приказаних на Сликама 4.4 и 4.5 једино резултат поступка одређивања карактеристика јер је снимање вршено са коначним бројем улазних одбирака (512) и са коначним бројем фреквенција улазног сигнала (256).

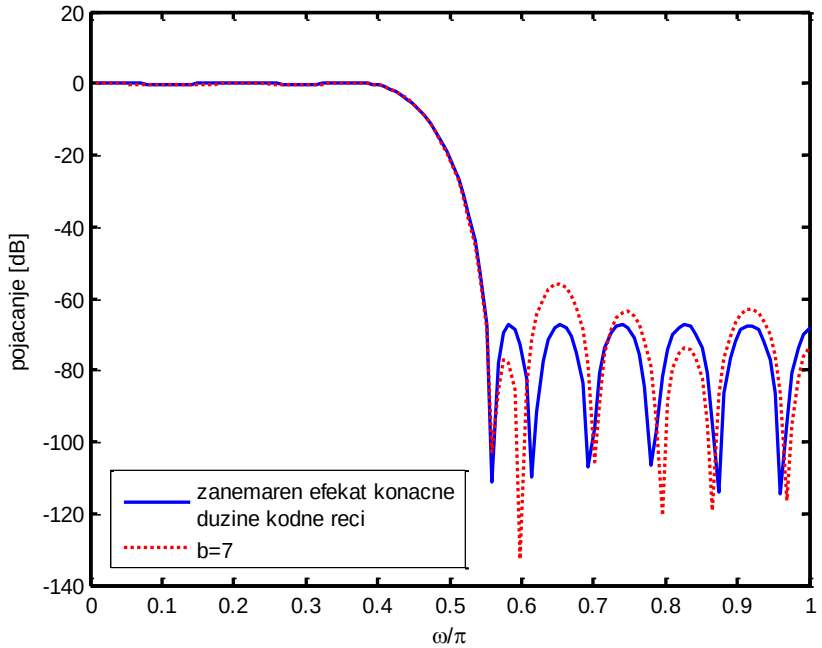


Слика 4.6 Упоредни график излаза из каскадне везе два филтра $H(z)$
и излаза из наше структуре за улазни сигнал $x(n)$

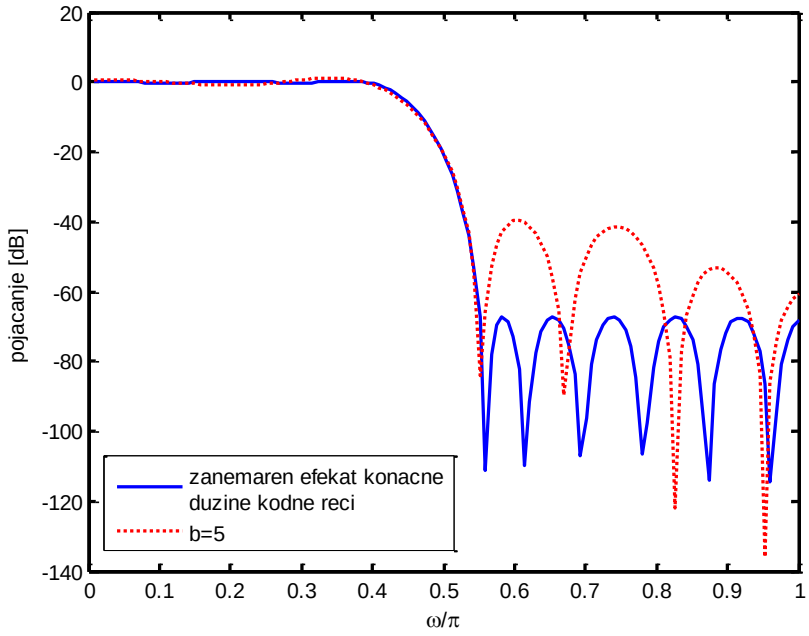
И на крају, да би био донет закључак да предложена структура може бити слободно примењена на реализацију каскадне везе *FIR* филтара потребно је извршити и проверу утицаја ефекта коначне дужине кодне речи. Провера ће бити извршена на тај начин што ће синусни сигнали бити доведени на улаз посматране структуре при чему ће сада дужина кодне речи бити ограничена поступком заокруживања на неколико вредности, и онда ће бити упоређена овако добијена амплитудска карактеристика са карактеристиком која би била добијена без ограничавања. Упоређивања ће бити извршена за дужине кодне речи b=9, 7 и 5. Резултати ове провере су приказани на Сликама 4.7, 4.8 и 4.9 респективно.



Слика 4.7 Утицај ефекта коначне дужине речи на предложену реализацију каскадне везе два филтра $H(z)$ за b=9



Слика 4.8 Утицај ефекта коначне дужине речи на предложену реализацију каскадне везе два филтра $H(z)$ за b=7



Слика 4.9 Утицај ефекта коначне дужине речи на предложену реализацију каскадне везе два филтра $H(z)$ за b=5

Са приказаних графика се види да се може тврдити да је утицај ефекта коначне дужине кодне речи у случају примене *PI* поступка на начин описан код реализације *FIR* филтара релативно мали, иако је испитивање вршено за каскадну везу само два филтра. Са увећањем броја филтара овај ефекат ће се увећавати, али не толико да изазове превелику вредност грешке.

Из претходних анализа може бити донет закључак да се поступак проточне обраде сигнала може слободно применити на реализацију структура са *FIR* филтрима при чему све позитивне особине поступка долазе до пуног изражаја без икакве деградације процесираних сигнала и са минималним утицајем ефекта коначне дужине кодне речи у односу на уобичајену реализацију.

4.2 ОДРЕЂИВАЊЕ ФУНКЦИЈЕ ПРЕНОСА *IIR PI* СТРУКТУРЕ ОДБИРАК ПО ОДБИРАК

Примена поступка проточне обраде сигнала на реализацију *FIR* филтара је ограничена једино технолошким могућностима расположивих ресурса (због увећања учестаности одабирања). За разлику од *FIR* филтара, имплементација овог поступка на *IIR*

филтре је сложенија. Највећи проблем представља увођење нових повратних спрега што условљава промену критичне петље.

Услов да су кашњења по свим затвореним петљама усклађена једино може бити остварен преуређивањем структуре *IIR* филтра. Показано је [7] да је примена поступка проточне обраде сигнала могућа уколико је *IIR* филтер реализован као паралелна структуру са свепропусним секцијама (енгл. *all-pass sections*), као што је то приказано на Слици 4.11, при чему ће се у овом раду користити само излаз нископропусног филтра.

Са Слике 4.10 се види да су функције преноса нископропусног филтра $H(z)$ и филтра пропусника високих учестаности $G(z)$ за предложену реализацију дате релацијама:

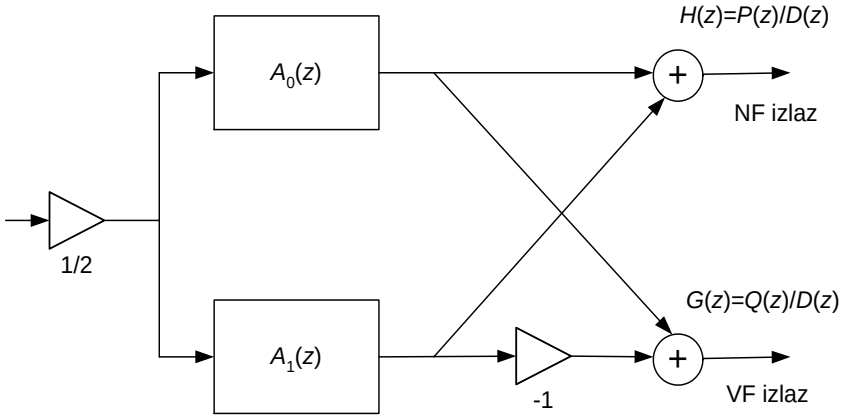
$$\begin{aligned} G(z) &= \frac{1}{2}\left[A_0(z) + A_1(z)\right] \\ H(z) &= \frac{1}{2}\left[A_0(z) - A_1(z)\right] \end{aligned}, \tag{4.3}$$

при чему ове функције чине двоструко комплементарни пар. Нека је

$$G(z) = \frac{P(z)}{D(z)} = \frac{p_0 + p_1 z^{-1} + \ldots + p_N z^{-N}}{1 + d_1 z^{-1} + d_2 z^{-2} + \ldots + d_N z^{-N}}, \tag{4.4}$$

а функција преноса $H(z)$:

$$H(z) = \frac{Q(z)}{D(z)} = \frac{q_0 + q_1 z^{-1} + \ldots + q_N z^{-N}}{1 + d_1 z^{-1} + d_2 z^{-2} + \ldots + d_N z^{-N}}, \tag{4.5}$$



Слика 4.10 Паралелна структура са свепропусницима

при чему из услова комплементарности у смислу снаге следи да мора важити

$$\begin{aligned} \left|G\left(e^{j\omega}\right)\right|^2 + \left|H\left(e^{j\omega}\right)\right|^2 &= 1 \\ G(z)G\left(z^{-1}\right) + H(z)H\left(z^{-1}\right) &= 1 \end{aligned}. \tag{4.6}$$

Посматрајући једначине (4.3) може се закључити да функција $G(z)$ мора имати симетричан бројилац, док функција $H(z)$ мора имати антисиметричан бројилац, тј.

$$\begin{aligned} p_n &= p_{N-n} \\ q_n &= -q_{N-n} \end{aligned}, \tag{4.7}$$

тако да ће за бројиоце важити:

$$\begin{aligned} P\left(z^{-1}\right) &= z^N P(z) \\ Q\left(z^{-1}\right) &= -z^N Q(z) \end{aligned}. \tag{4.8}$$

Замењујући $G(z) = \dfrac{P(z)}{D(z)}$ и $H(z) = \dfrac{Q(z)}{D(z)}$ у једначине (4.6) и поштујући услов (4.8) добија се да је:

$$\left[P(z) + Q(z)\right]\ \left[P(z) - Q(z)\right] = z^{-N} D(z) D\left(z^{-1}\right), \tag{4.9}$$

а комбинујући једначине (4.8):

$$P((z) - Q(z) = z^{-N}\left\{P\left(z^{-1}\right) + Q\left(z^{-1}\right)\right\}. \tag{4.10}$$

Уколико се нуле полинома $\left[P(z) + Q(z)\right]$ означе са $z = \xi_k$, $1 \le k \le N$ онда из једначине (4.9) следи да ће нуле полинома $\left[P(z) - Q(z)\right]$ бити $z = \dfrac{1}{\xi_k}$, $1 \le k \le N$. Такође из једначине (4.10) се види да нуле полинома $\left[P(z) + Q(z)\right]$ унутар јединичног круга представљају и нуле $D(z)$ док нуле ван јединичног круга представљају нуле $D\left(z^{-1}\right)$ јер

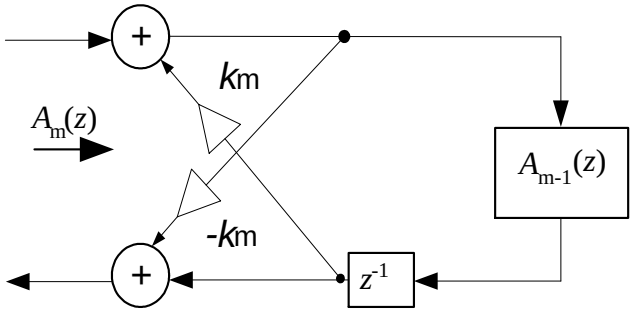
претпостављамо да су $G(z)$ и $H(z)$ стабилне функције преноса. Уколико су $z=\xi_k$, $1\le k\le r$ r нула полинома $[P(z)+Q(z)]$ унутар јединичног круга, а преосталих $z=\xi_k$, $r+1\le k\le N$, *H-r* нула изван јединичног круга, онда ће првих *r* нула функције $D(z)$ бити $z=\xi_k$ за $1\le k\le N$ док ће преосталих *H-r* нула бити $z=\frac{1}{\xi_k}$ за $r+1\le k\le N$, тако да ће функције свепропусника бити одређене следећим једначинама:

$$A_0(z)=\prod_{k=r+1}^{N}\frac{z^{-1}-(\xi_k^{*})^{-1}}{1-\xi_k^{-1}z^{-1}}, \tag{4.11}$$

$$A_1(z)=\prod_{k=1}^{r}\frac{z^{-1}-\xi_k^{*}}{1-\xi_k\ z^{-1}}, \tag{4.12}$$

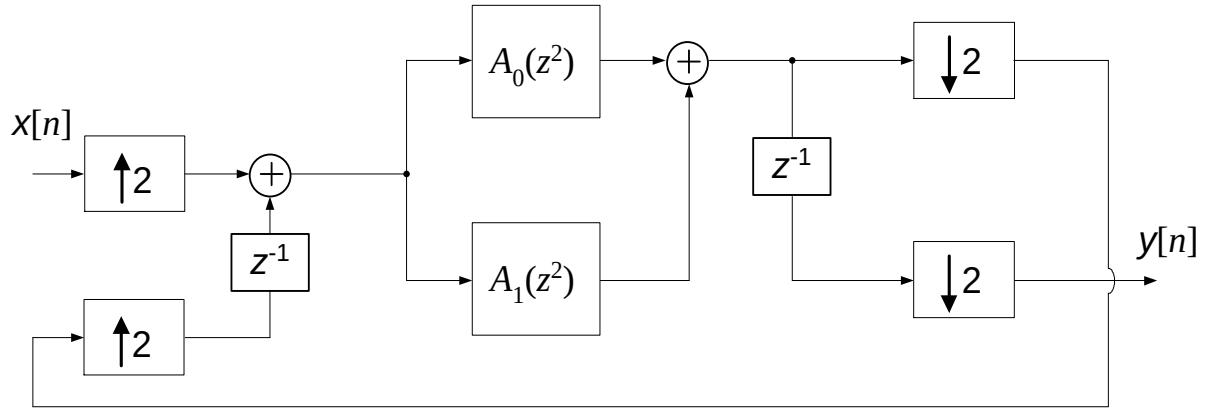
Услове да се могу представити у форми са Сслике 4.10 задовољавају Батервортови, Чебишевљеви и елиптички филтри изведени билинеарном трансформацијом.

Свепропусни филтри ће бити реализовани вишеструком екстракцијом мреже са два пара крајева, као што је то приказано на Слици 4.11, при чему се коефицијенти k_m рачунају за сваки корак. На крају се долази до коначне структуре *IIR* филтра која ће бити коришћена при реализацији функције $H(z^2)$.



Слика 4.11 Реализација свепропусних филтара вишеструком екстракцијом мреже са два пара крајева

На Слици 4.12 је приказан начин реализације по ком ће бити вршено снимање у реалном времену *PI* поступка за извођење каскадне везе два *IIR* филтра тј. реализације функције $H(z^2)$, а на Слици 4.13 начин на који ће бити реализоване функције свепропусника.



Слика 4.12 Модификовани *PI* поступак за *IIR* филтре

Као и у случају снимања карактеристика *PI* поступка код *FIR* филтара, снимање ће бити извршено на тај начин што ће на улаз бити доведен синусни сигнал чија ће се фреквенција мењати у опсегу од нула до π (1024 одбирака и 256 тачака фреквенције за снимање амплитудске карактеристике) и бити одређена тачка по тачка карактеристике. У циљу избегавања утицаја цурења спектра, за одређивање карактеристике као и за израчунавање максималне вредности грешке ће се користити 512 од укупно 1024 одбирка из средине опсега (као и у случају *FIR* филтара).

Сви елементи ће бити реализовани на исти начин као и у случају примене *PI* поступка на реализацију *FIR* филтара, изузев филтра $H(z^2)$ који ће као што је речено бити реализован као паралелна структура са свепропусницима. Почетни услови остају исти као и код примене на *FIR* филтре сем што је потребно да због повратних спрега одређени број излазних одбирака буде одбачен како би излаз из наше структуре био идентичан излазу из каскадне везе два филтра $H(z^2)$ (за број потребан да се први одбирак процесира и стигне до излаза).

За филтер $H(z)$ ће бити изабран елиптички филтер (овакав филтер је могуће извести на начин као на Слици 4.12):

```
[N,wn]=ellipord(0.4,0.5,0.5,36);
```

чиме се добија филтер петог реда са коефицијентима:

```
[b,a]=ellip(5,0.5,36,0.4).
```

Карактеристика овог филтра приказана је на слици 4.14.

Полови функције преноса одређују се наредбом tf2zp на следећи начин:

```
[Z,L,K]=tf2zp(b,a).
```

На тај начин се добијају следећи полови функције преноса:

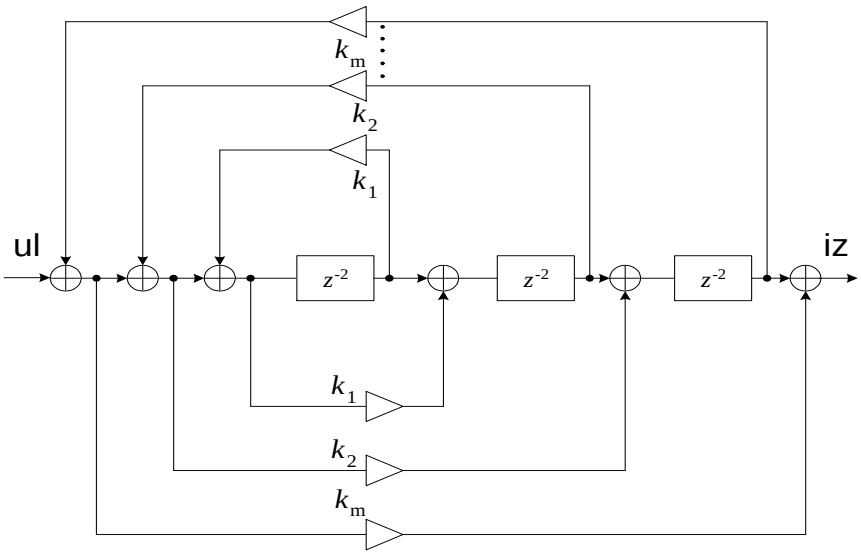
$\zeta_0 = 0.2812 + 0.9033i$

$\zeta_1 = 0.2812 - 0.9033i$

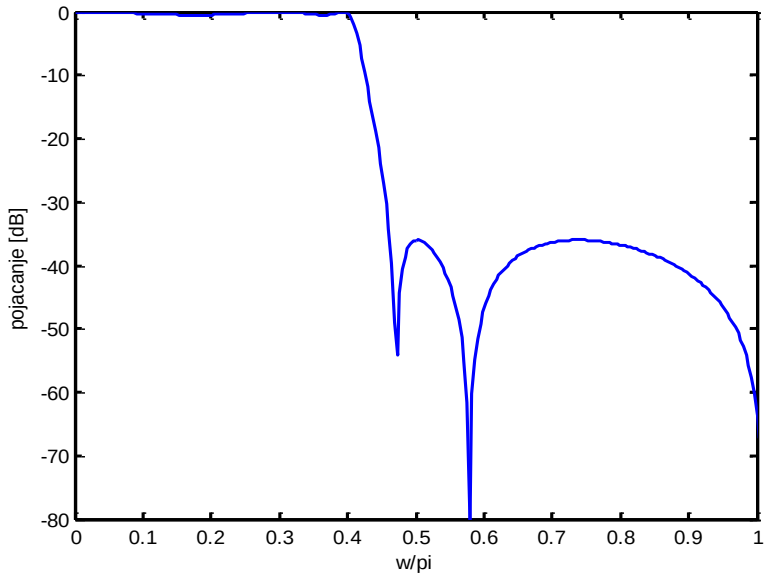
$\zeta_2 = 0.3695 + 0.6442i$

$\zeta_3 = 0.3695 - 0.6442i$

$\zeta_4 = 0.4714$

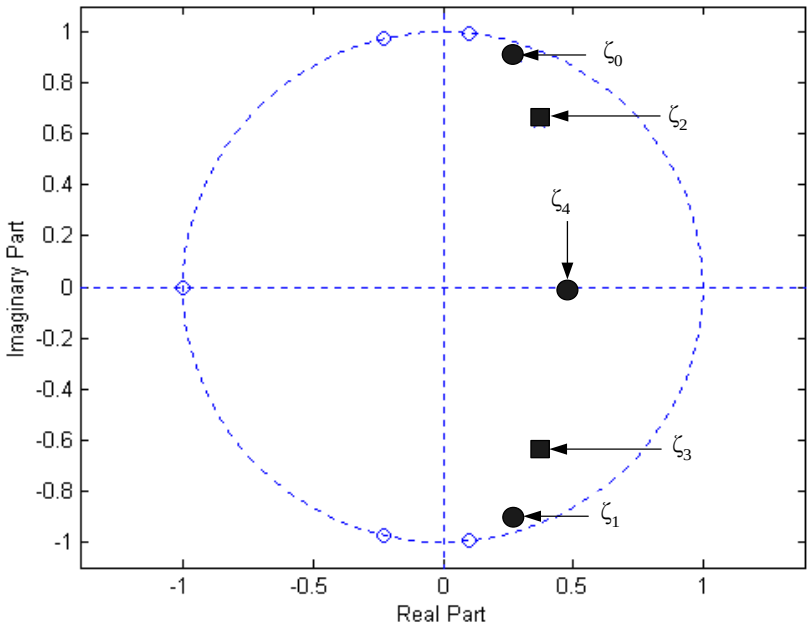


Слика 4.13 Структура свепропусника



Слика 4.14 Амплитудска карактеристика филтра $H(z)$

Њихов распоред је приказан на Слици 4.15 при чему су са "●" означени полови који одређују свепропусну секцију $A_0(z)$, а са " ■" полови који одређују свепропусну секцију $A_1(z)$.

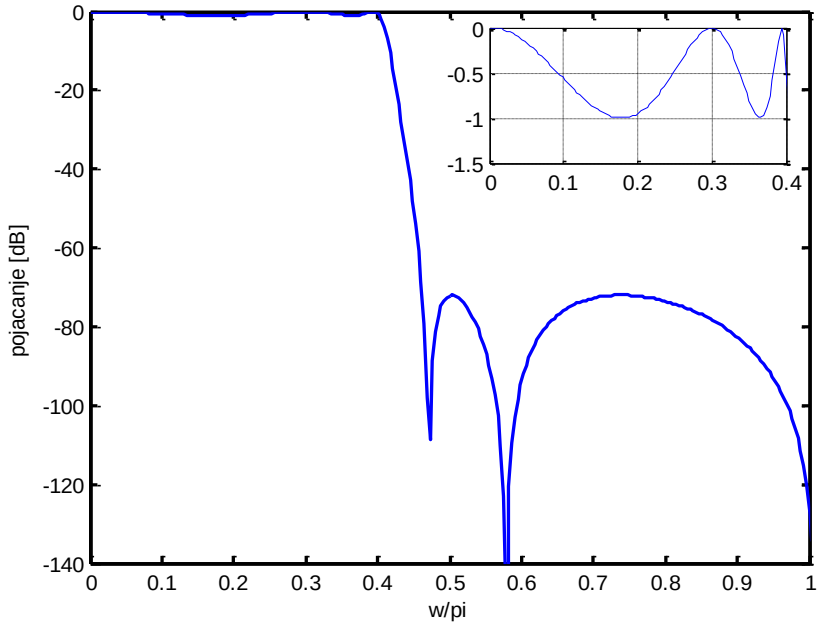


Слика 4.15 Распоред полова функције преноса $H(z)$

Као што је познато, најмања осетљивост карактеристике у пропусном опсегу на промену коефицијената филтра као и најефикаснија реализација у *PI* техници *IIR* филтара у смислу задовољавања услова критичне петље, биће остварена ако су свепропусне секције изведене као каскадне везе свепропусника првог и другог реда [8]. Из тог разлога ће $A_0(z)$ бити реализована као каскадна веза свепропусних секција првог и другог реда, а $A_1(z)$ као свепропусна секција другог реда. Константе k_m свепропусних секција другог реда биће одређене на следећи начин:

```
den1=conv([1,-L(5,1)],[1,-L(2,1)]);
den2=conv([1,-L(3,1)],[1,-L(4,1)]);
k0=poly2rc(den1);
k1=poly2rc(den2);
knew0=fliplr(k0);
knew1=fliplr(k1),
knew0 = [-0.3521    0.8807   -0.4219]
knew1 = [-0.4763 0.5515],
```

док ће константа k_1 секције првог реда бити једнака коефицијенту ζ_4. Карактеристика каскадне везе два филтра $H(z^2)$ (тј. карактеристика каскадне везе филтара без примене *PI* поступка какву би требао остварити) је приказана на Слици 4.16.

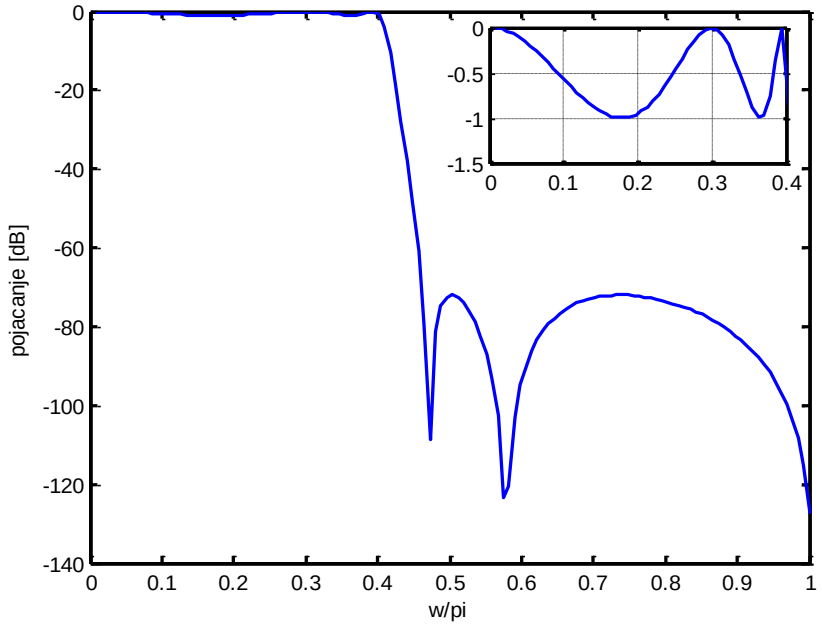


Слика 4.16 Амплитудна каскадне везе два филтра $H(z^2)$

Погледајмо сада како ће изгледати карактеристике наше структуре у случају каскадне везе уколико се примени *PI* поступак на изабрани *IIR* филтер. После извршеног снимања на описани начин добијена је карактеристика која је приказана на Слици 4.17. док су излази у временском домену на неколико фреквенција приказани на Слици 4.18. Максимална грешка за све одбирке на свим фреквенцијама износи:

$$E_{max} = 4.6518 \cdot 10^{-14}$$

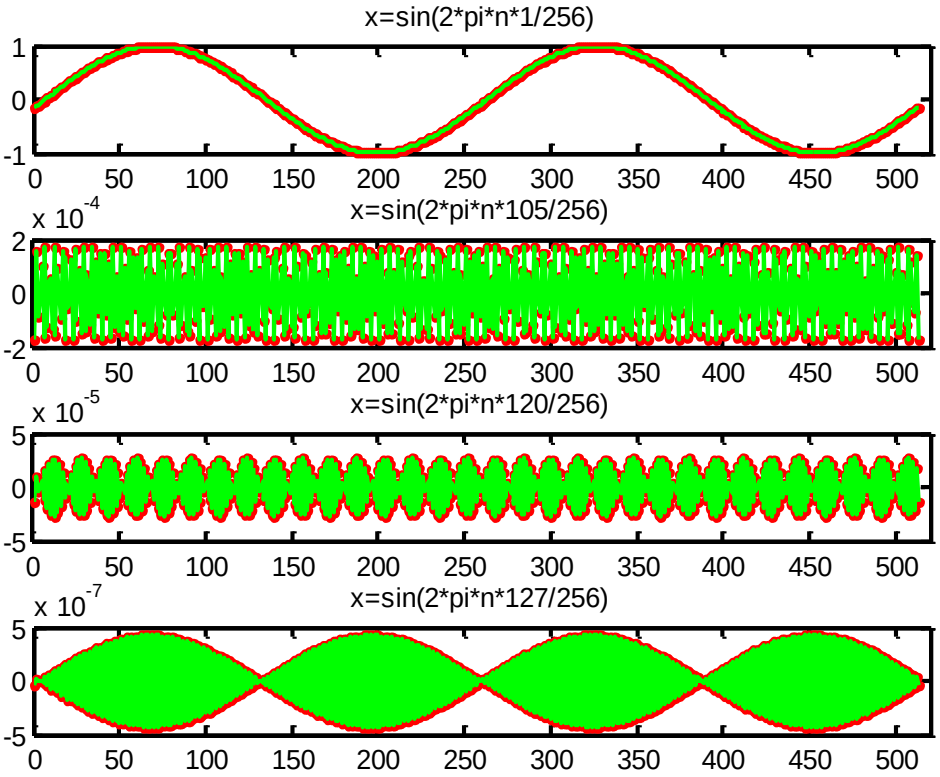
Са приказаних слика је очигледано да се и у случају *IIR* филтара применом *PI* поступка добијају резултати јако блиски захтеваним вредностима. Међутим, максимална грешка сада није једнака нули као у случају примене на *FIR* филтре, а амплитудска карактеристика за нијансу слабије прати карактеристику коју добијамо без примене *PI* поступка. Такође, са слике 4.19 на којој је приказана карактеристика шума коју проузрокује примена *PI* поступка, види се да ниво шума јесте низак али да ипак постоји (а треба и имати на уму да је испитивање вршено за каскадну везу само два филтра.



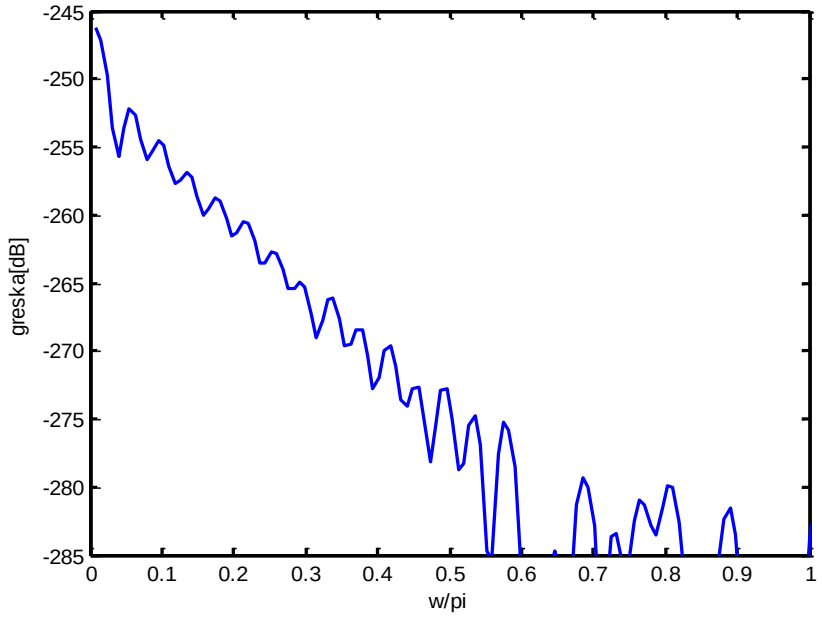
Слика 4.17 Амплитудна нове структуре снимана одбирак по одбирак за случај каскадне везе два филтра

Размотримо сада узроке оваквих одступања. Један од могућих узрока би могла бити грешка заокруживања. Тако на пример, изабрани *IIR* филтер није директно реализован као у случају *FIR* филтара већ су прво израчунати коефицијенти одговарајућег филтра са свепропусним секцијама, а приликом симулације у *Matlab*-у оперише се са коначним вредностима. Други узрок би могао бити у самом поступку симулације. Наиме, приликом одређивања излазног сигнала који се добија узастопним филтрирањем са два *IIR* филтра без примене *PI* поступка користи се наредба *filter*, (у случају *FIR* филтара где оваква грешка није постојала коришћена је наредба *conv*) а чији се алгоритам нешто разликује од поступка одређивања излазног сигнала методом одбирак по одбирак онако како је дефинисан у овом раду. И на крају, ако се има на уму да је величина грешке јако мала (реда 10^{-14}), може се закључити да се она сигурно односи на грешку заокруживања и грешку симулације, а да је предложени поступак реализације са *IIR* филтрима добар.

Приказани графици показују да се поступак проточне обраде сигнала може ефекасно применити и на структуре реализоване са *IIR* филтрима и да практично неће бити разлике између сигнала добијених са или без примене *PI* поступка. Једина разлика у односу на примену на *FIR* филтре је та што није могуће директно применити *PI* поступак на изабрани *IIR* филтер већ се мора потражити она реализација којом ће бити задовољени услови критичне повратне спреге.

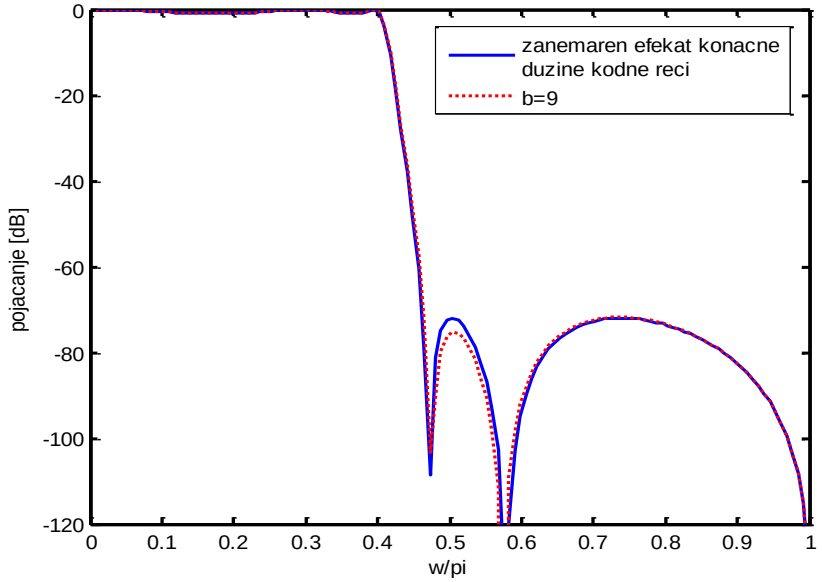


Слика 4.18 Упоредни график излаза из каскадне везе два филтра $H(z)$ и излаза из наше структуре за улазни сигнал $x(n)$

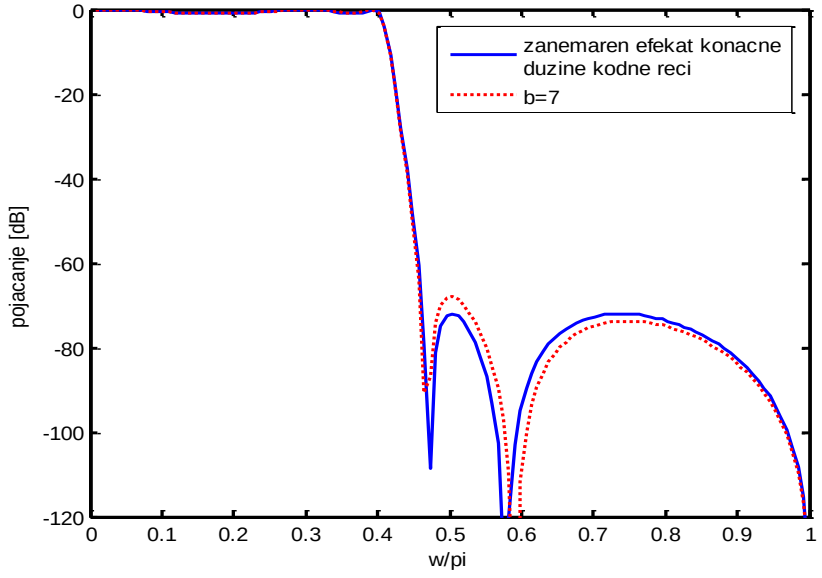


Слика 4.19 Карактеристика шума услед примене *PI* поступка на реализацију каскадне везе два филтра $H(z)$

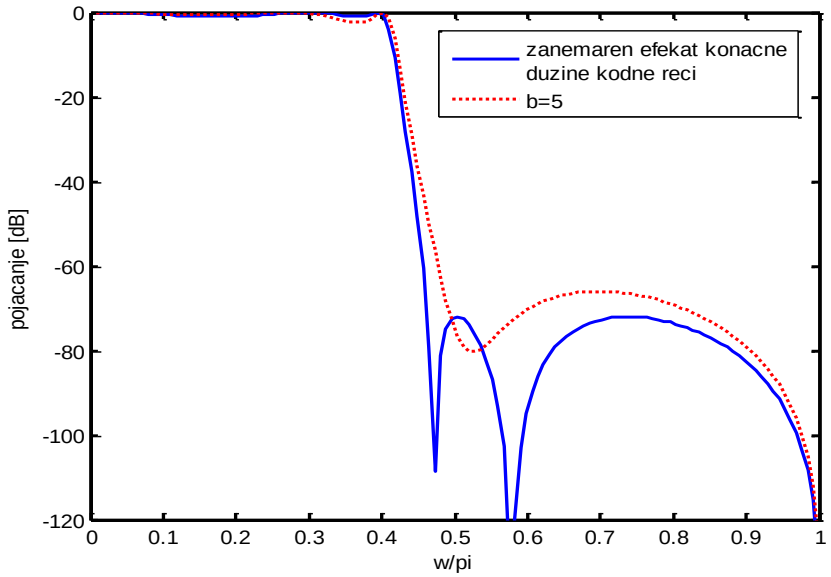
И на крају потребно је да се изврши провера утицаја ефекта коначне дужине кодне речи како би се проверила прихватљивост предложеног решења. Као и у случају *FIR* филтара ова провера ће бити извршена упоређивањем карактеристика наше структуре са и без узимања у обзир ефекта коначне дужине кодне речи. Резултати ових испитивања су приказани на Сликама 4.20, 4.21 и 4.22 на којима је приказан утицај ефекта коначне дужине кодне речи за b=9; b=7 и b=5 респективно.



Слика 4.20 Утицај ефекта коначне дужине речи на предложену реализацију каскадне везе два филтра $H(z)$ за b=9



Слика 4.21 Утицај ефекта коначне дужине речи на предложену реализацију каскадне везе два филтра $H(z)$ за b=7



Слика 4.22 Утицај ефекта коначне дужине речи на предложену реализацију каскадне везе два филтра $H(z)$ за b=5

Са претходних слика на којима је приказан утицај ефекта коначне дужине кодне речи може да се закључи да је утицај овог ефекта и на структуре са *IIR* филтрима реализоване у *PI* техници задовољавајући и прихватљив. Наравно треба имати на уму да је испитивање вршено за *PI* структуру са само два *IIR* филтра тако да би за реализације са већим бројем филтара у каскадној вези морала да се изврши провера утицаја ефекта коначне дужине кодне речи за конкретан случај.

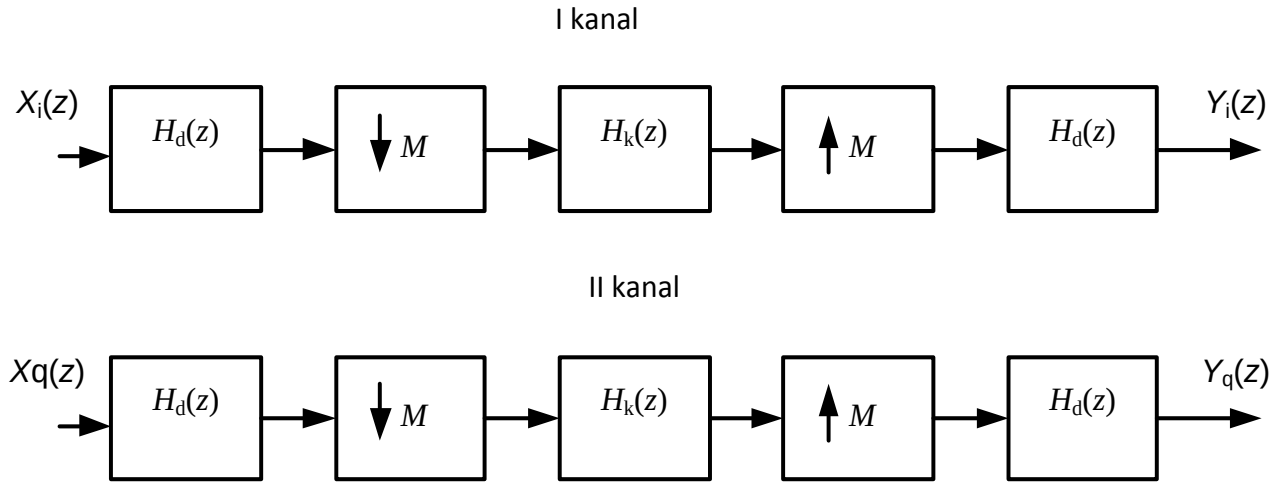
5. ПРИМЕНА *MULTIRATE* МЕТОДА КОД ЕФИКАСНИХ РЕАЛИЗАЦИЈА ДИГИТАЛНИХ ФИЛТАРА

Дигитални филтри са уском прелазном зоном су веома тешки за реализацију, а понекад је и немогуће реализовати их користећи конвенционалне структуре. Проблем код *FIR* филтара је тај што захтевају да ред филтра буде јако велики да би се задовољили захтеви уске прелазне зоне. Код *IIR* филтара, проблем је изражена осетљивост од положаја полова функције преноса. Из ових разлога је у великом броју случајева *мултирате* приступ и једино решење при пројектовању *FIR* или *IIR* филтара са уском прелазном зоном.

Примењујући поступак проточне обраде сигнала могу се извести различите реализације структура *multirate* филтрирања и тиме постићи знатне хардверске уштеде и поједноставити сложене структуре у случајевима код истовременог процесирања два или више канала. У овом поглављу ће бити показано како се *PI* техника може применити на вишестепене филтре, фреквенцијско маскирање и *QMF* банке са структуром стабла и биће приказани резултати такве примене.

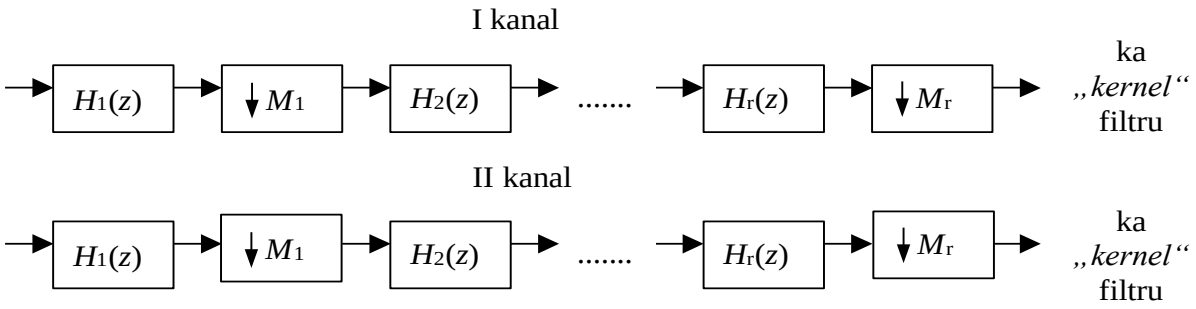
5.1 ВИШЕСТЕПЕНИ (*MULTISTAGE*) ФИЛТРИ

Генерална структура вишестепеног НФ филтра са ниском граничном фреквенцијом за два канала приказана је на Слици 5.1.



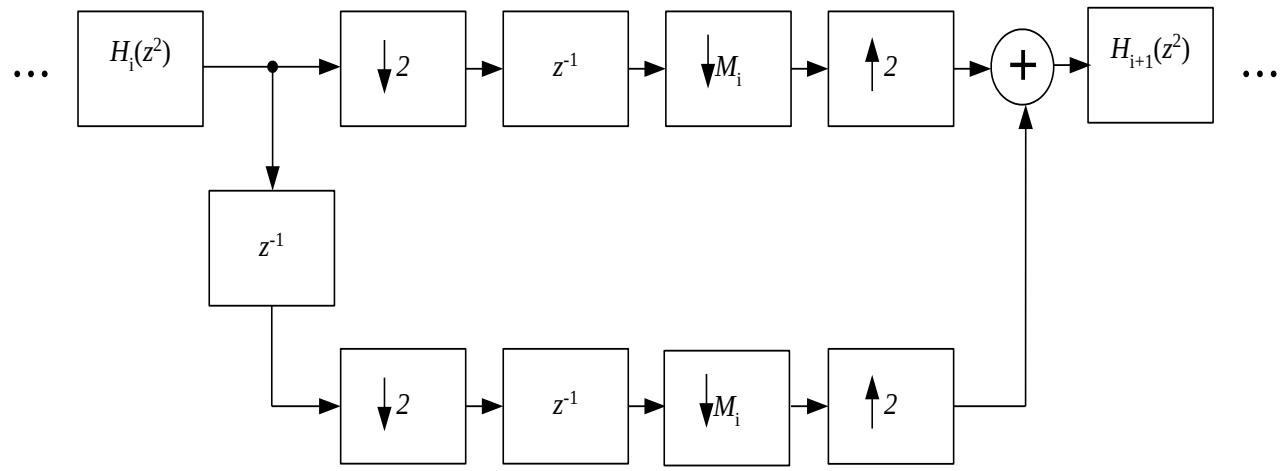
Слика 5.1 Вишестепено филтрирање

У случајевима ускопојасних филтара, где је децимациони/интерполациони фактор велики (на пример $M > 10$), ефикаснија је примена вишеструких децимационих/интерполационих филтара, као што је приказано на Слици 5.2 за децимациони део. На овај начин би уместо једног децимационог/интерполационог филтра било употребљено више оваквих филтара (сагласно броју степена) али знатно нижег реда. Уколико су захтеви за ширином прелазне зоне и дозвољеним слабљењима престроги овакав начин је и једино решење [9].



Слика 5.2 Децимација у више степена

Интерполациони део би се аналогно Слици 5.2 састојао из истог броја кола за увећање учестаности одабирања и исто толико интерполационих филтара. Након примене поступка проточне обраде сигнала на део за децимацију, добија се структура као на Слици 5.3, при чему би број оваквих блокова био онолики на колико је степена подељен M .



Слика 5.3 Један степен децимационог дела вишестепеног филтра

После сваког филтра $H_i\left(z^2\right)$ следи уобичајено раздвајање сигнала по каналима (кола за снижавање учестаности одабирања и кола за кашњење), затим снижавање учестаности одабирања за фактор M_i за сваки канал посебно и на крају поновно мултиплексирање

канала због филтрирања наредним степеном $H_{i+1}(z^2)$, и тако редом све до последњег проширеног филтра у делу за децимацију.

Аналогно овоме, интерполациони део може бити подељен на исти број степена и *PI* поступак се може применити на сваки степен за два канала. На тај начин ће бити добијена слична структура блокова између суседних проширених филтара, осим што ће се уместо снижавања учестаности одабирања за фактор M_i, сада вршити увећавање учестаности за исту вредност.

Закључује се да се директном применом *PI* технике вишестепено филтрирање чак и усложњава. Из тог разлога неопходно је да се крене корак даље тј. да се потраже такве реализације које би поједноставиле структуру са Слике 5.3. Сложена структура добијена применом *PI* технике на описани начин би могла да се упрости ако се обрати пажња на редослед операција над одбирцима сигнала сваког канала. Тако на пример ако се посматра структура између проширених филтара која је приказана на Слици 5.3, види се да се она састоји једино из кола за промену учестаности одабирања и кола за кашњење. У зависности од броја канала (у датом случају ради се о два канала) и фактора M са којим су реализовани вишестепени филти, може се дефинисати алгоритам помоћу ког би се директно израчунавали одбирци излазног сигнала из посматране структуре (тј. сигнал на улазу у филтер $H_{i+1}(z^2)$) у функцији улазних одбирака у структуру (тј. сигнал са излаза филтра $H_i(z^2)$).

Дакле, део између $H_i(z^2)$ и $H_{i+1}(z^2)$ би могао да се дефинише као ново коло, назовимо га еквивалентни дециматор, чија би улога била:

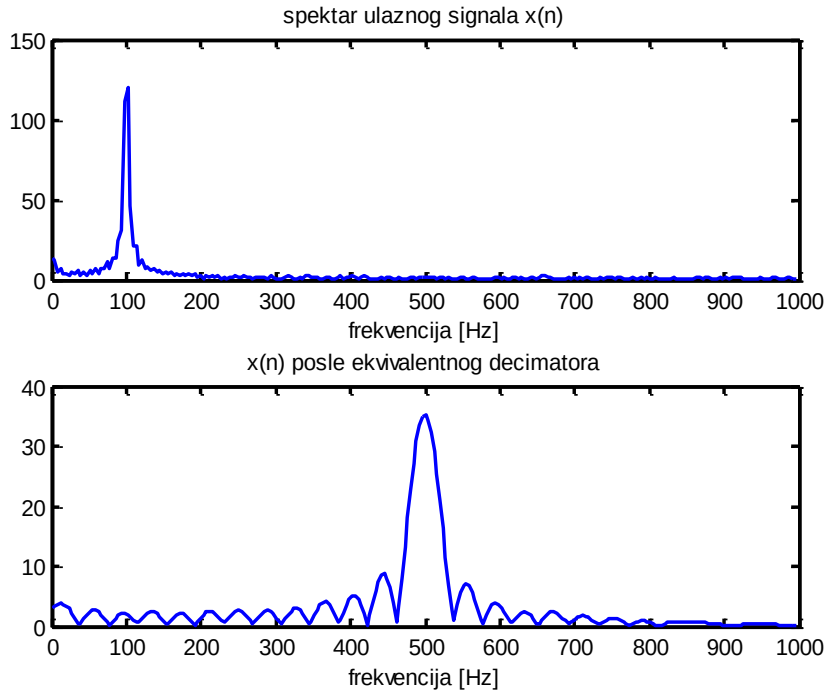
- сачувај два узастопна одбирка,
- изостави следећих $2\cdot(M_k\text{-}1)$одбирака,
- понављај поступак до последњег одбирка.

На исти начин, идентични интерполациони филтри се реализују у *PI* техници, а функција модификованог интерполатора би била:

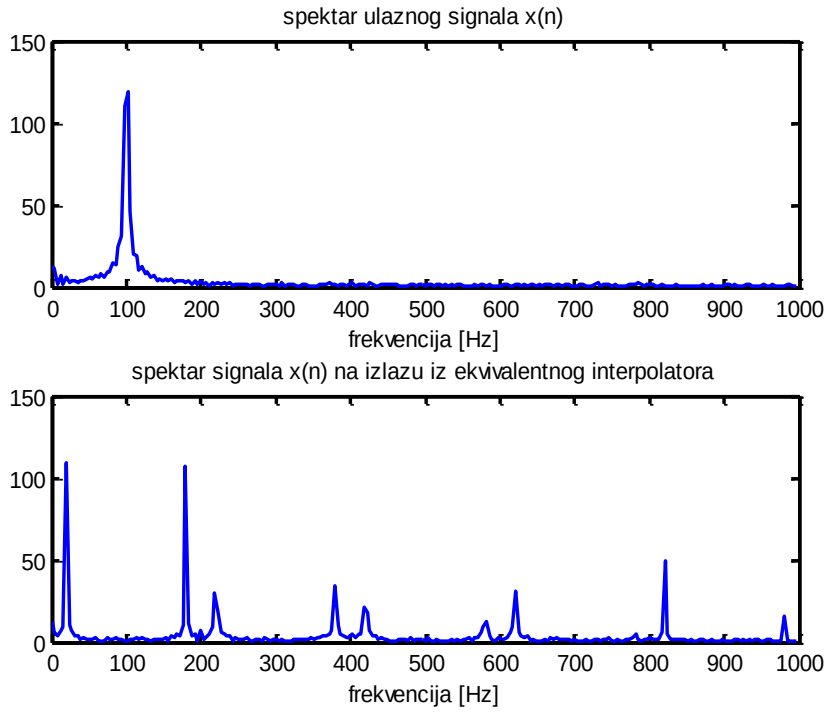
- сачувај два узастопна одбирка,
- убацити $2\cdot(M_k\text{-}1)$нула,
- понављај поступак до последњег одбирка.

Као што се види, алгоритми еквивалентног дециматора и интерполатора зависе једино од броја канала реализованих у *PI* техници и броја степени вишестепеног филтра, тако да се генерално могу користити од случаја до случаја, мењајући само филтер $H(z)$ у зависности

од захтеваних карактеристика На Сликама 5.4 и 5.5 је приказан процес поступка еквиваленте децимације и еквивалентне интерполације у фреквенцијском домену при $M = L = 5,$



Слика 5.4 Процес поступка еквивалентне децимације у фреквенцијском домену за $M = 5$



Слика 5.5 Процес поступка еквивалентне децимације у фреквенцијском домену за $L = 5$

Погледајмо резултате које би добили у случају примене претходног разматрања у пракси. Прво ће бити приказан поступак директног филтрирања, затим применом вишестепених филтара и на крају ће бити посматрано филтрирање за случај два канала применом поступка проточне обраде сигнала, при чему ће бити уведене предложене новине. Нека је потребно реализовати филтер следећих карактеристика:

$F_p = 50\ Hz, F_s = 100\ Hz,$

$\delta_p = 0{,}01,\ \delta_s = 0{,}001 (60\ dB),$

$F_o = 2\ 000 Hz.$

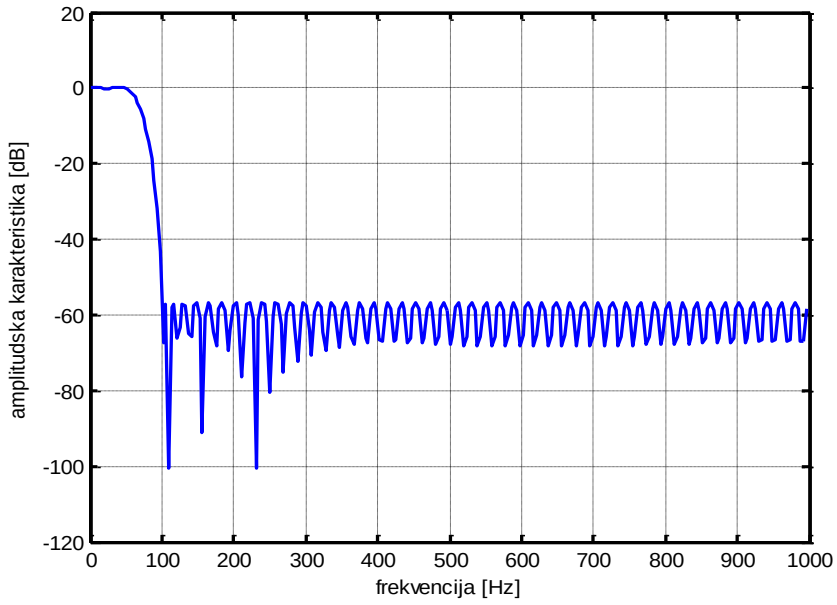
Ако би био коришћен само један *FIR* филтер важило би:

```
[N,fp,mag,wt]=remezord([50 100],[1 0],[0.01 0.001],2000);
b=remez(N,fp,mag,wt);
N=102.
```

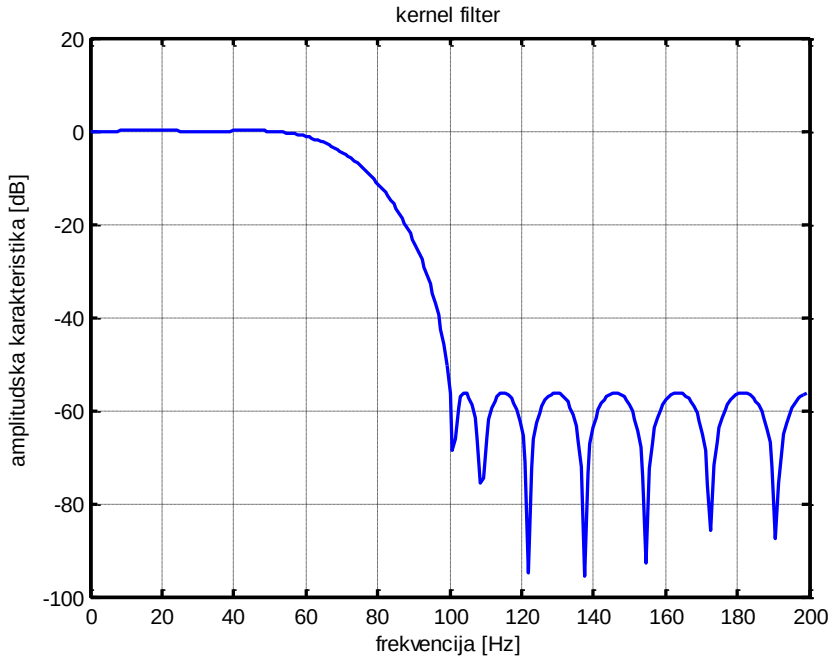
Значи да би *FIR* филтер био јако великог реда, тј. *N=102*. Карактеристика оваквог филтра је приказана на Слици 5.6.

Нека је сада потребно реализовати овакав филтар као вишестепени филтер са једним степеном (једним интерполатором и једним дециматором). Нека је $M = 5$. *Кернел* филтар је сада одређен за фреквенцију одабирања $2000/5 = 400\text{Hz}$. За децимациони и интерполациони филтар остаје $F_0 = 2000\ Hz$ за граничне фреквенције $50\ \text{Hz}$ и $300\ \text{Hz}$, а δ_s *kernel* филтра остаје исто. За децимациони и интерполациони филтар се бира $\delta_s = 0{,}01$ а за сва три филтра $\delta_p = 0.01/3$.

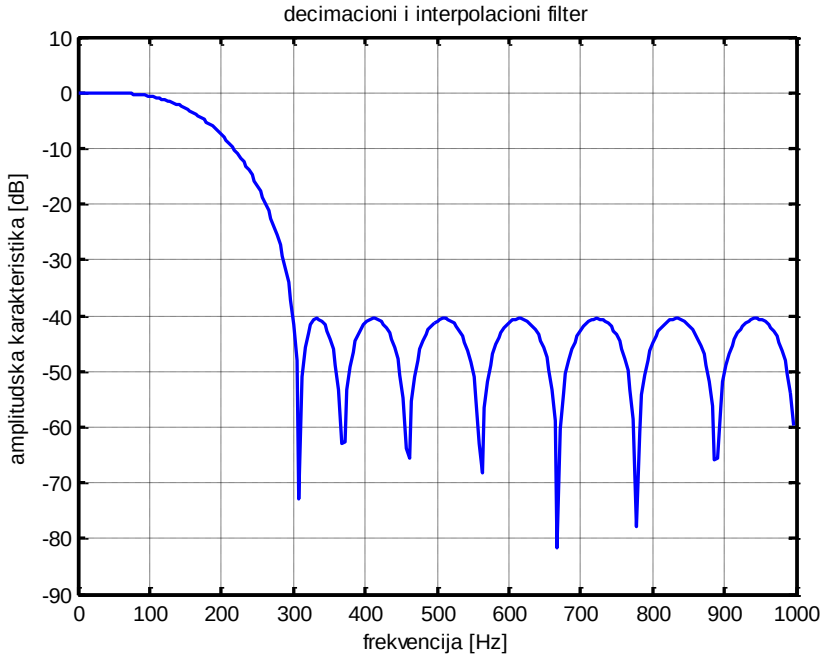
За овако дефинисане услове добија се да је *kernel* филтер 17-тог реда (у случају директне реализације ред филтра је *N=102*). Његова карактеристика као и карактеристика децимационог и интерполационог филтра и укупна карактеристика приказане су на Сликама 5.7, 5.8 и 5.9.



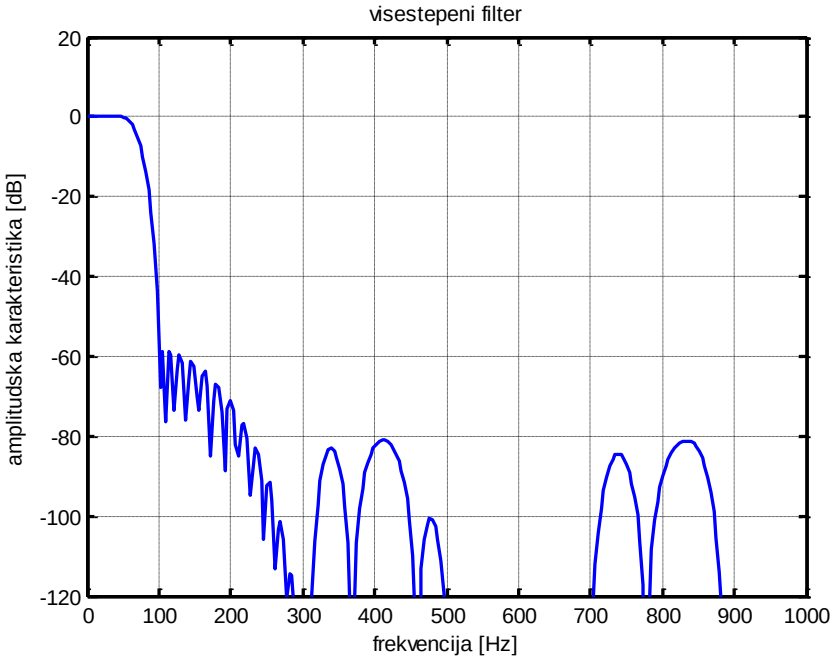
Слика 5.6 Амплитудна карактеристика филтра који би се директно применио за задате услове



Слика 5.7 Амплитудна карактеристика *kernel* филтра



Слика 5.8 Амплитудна карактеристика интерполационог и децимационог филтра



Слика 5.9 Амплитудна карактеристика вишестепеног филтра

Претходне слике показују предности вишестепеног филтрирања у односу на директну примену. Наравно, за два канала би била требна два оваква филтра. Проблем међутим представља стање уколико је M сувише велико (на пример $M > 10$). Нека је сада потребно реализовати филтар истих карактеристика

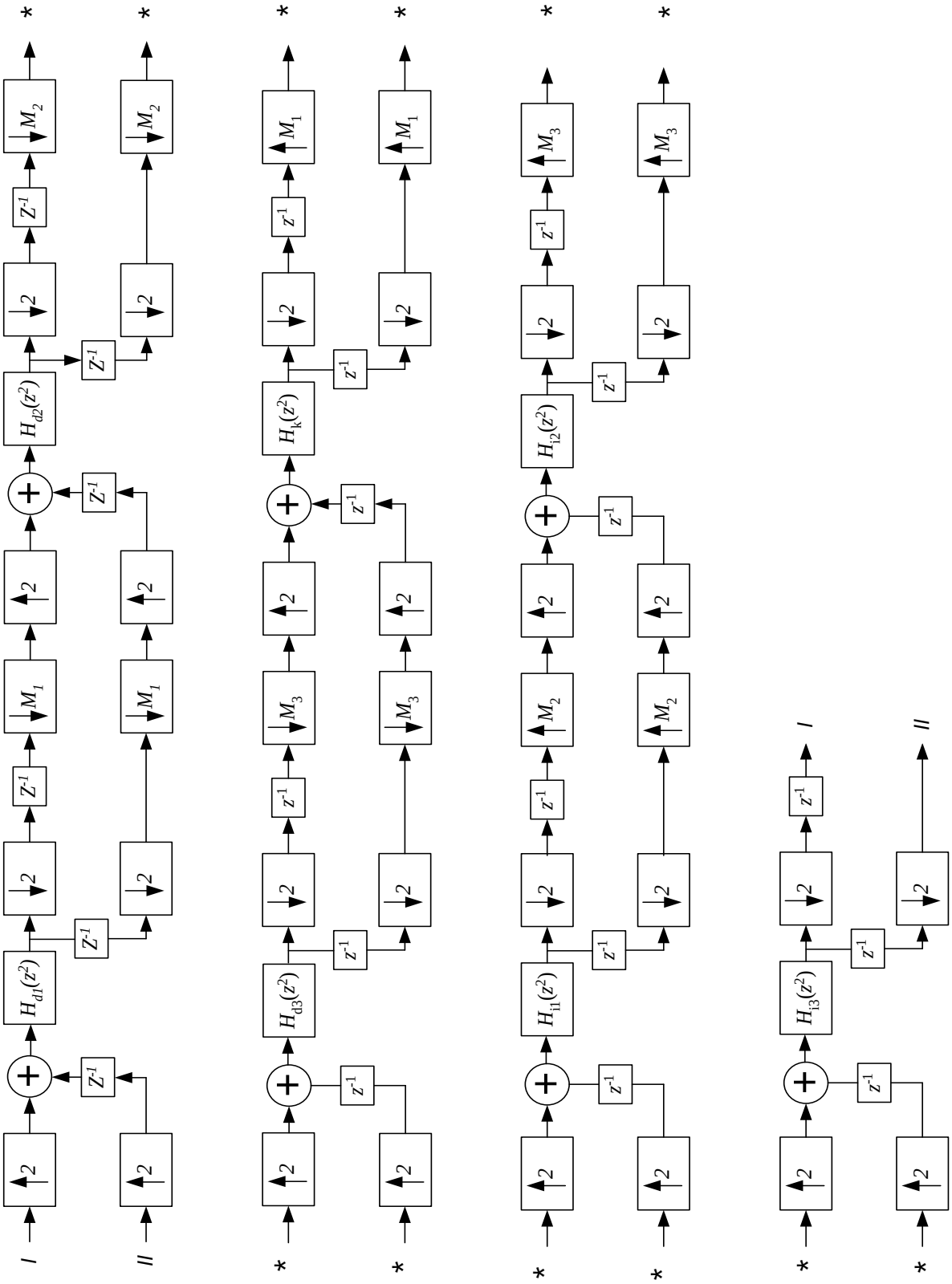
($F_p = 50\ Hz, F_s = 100\ Hz, \delta_p = 0{,}01, \delta_s = 0{,}001$) али да је $F_o = 3200\ Hz$. Ако би *kernel* филтар остао исти ($F_{ok} = 400\ Hz$) требало би да је $M = 3200/400 = 8$. Због тога ће овакав филтер бити реализован у три степена са истим $M = 2$, при чему ће бити примењен предложени поступак, како је претходно објашњено (за случај два канала). На Слици 5.10 приказан је начин реализације у *PI* техници за два канала. Посматрајући слику стекао би се утисак да примена поступка проточне обраде сигнала само усложњава поступак филтрирања, међутим ако би била уведена нова кола еквивалентног дециматора и еквивалентног интерполатора поступак би се знатно упростио. После улазног мултиплексирања канала следили би децимациони филтри проширени за по једно кашњење заједно са еквивалентним дециматором и то онолико њих на колико степена је развијено филтрирање, затим следи проширени *kernel* филтер а иза њега следе блокови проширеног интерполационог филтра заједно са еквивалентним интерполаторима. На крају се сигнали раздвајају по каналима на начин уобичајен код поступка проточне обраде сигнала. Слика 5.11 приказује описани поступак, одакле се закључује да процес филтрирања није више толико сложен и да је знатно упрошћен у односу на стандардну примену *PI* технике (Слика 5.10).

Провера предложеног поступка биће извршена на исти начин на који је проверен поступак проточне обраде сигнала код примене на реализацију каскадне везе *FIR* и *IIR* филтара. На улаз структуре биће доведен синусоидални сигнал чија се фреквенција мења у опсегу од нула до π и за сваку фреквенцију ће бити одређена тачка по тачка функције преноса.

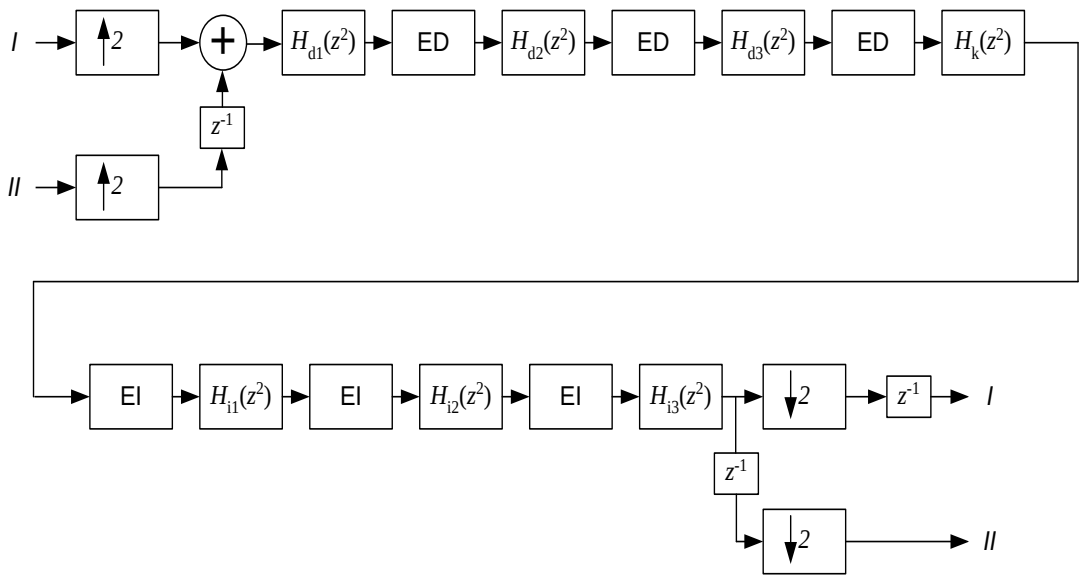
Филтри ће бити дефинисани на следећи начин:

Филтри $H_{d1}(z)\ i\ H_{i3}(z)$*:* $F_0 = 3\ 200\ Hz, F_p = 50\ Hz, F_s = 1\ 200\ Hz, \delta_p = 0.01/3, \delta_s = 0{,}001$
Филтри $H_{d2}(z)\ i\ H_{i2}(z)$*:* $F_0 = 1600\ Hz, F_p = 50\ Hz, F_s = 600\ Hz, \delta_p = 0.01/3, \delta_s = 0{,}001$*;*
Филтри $H_{d3}(z)\ i\ H_{i1}(z)$*:* $F_0 = 800\ Hz, F_p = 50\ Hz, F_s = 300\ Hz, \delta_p = 0.01/3, \delta_s = 0{,}001$*;*
"Kernel" филтер $H_k(z)$*:* $F_0 = 400\ Hz, F_p = 50\ Hz, F_s = 100\ Hz, \delta_p = 0.01, \delta_s = 0{,}001$*;*

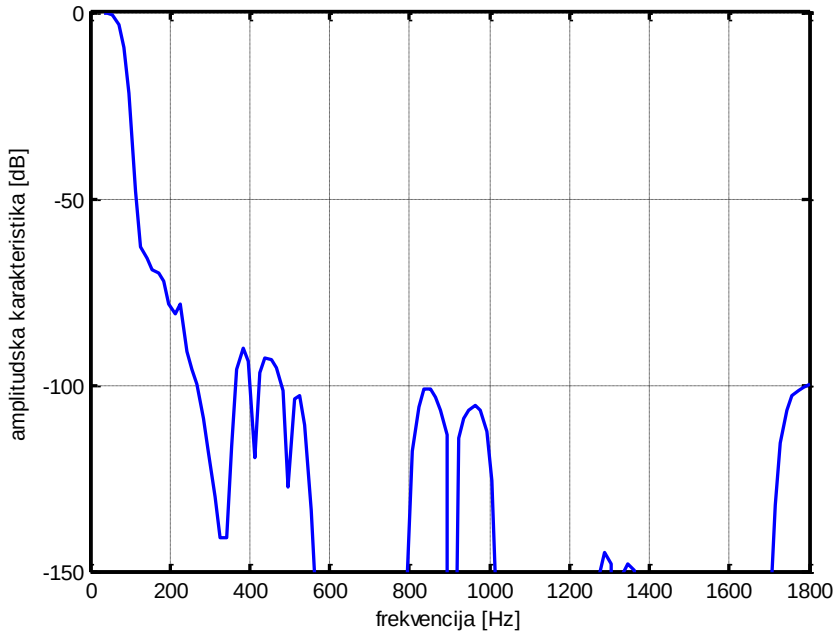
Карактеристика структуре са Слике 5.11, при чему је употребљен филтер дефинисан на претходни начин, приказана је на Слици 5.12. Са слике се закључује да амплитудска карактеристика нове структуре задовољава дефинисане захтеве. Такође, посматрајући амплитудску карактеристику примећује се да су одступања карактеристике значајнија него у случају реализације само једног филтра.



Слика 5.10 Примена *PI* поступка на вишестепени филтер са три степена и два канала



Слика 5.11 Коначна структура вишестепеног филтра са три степена и два канала

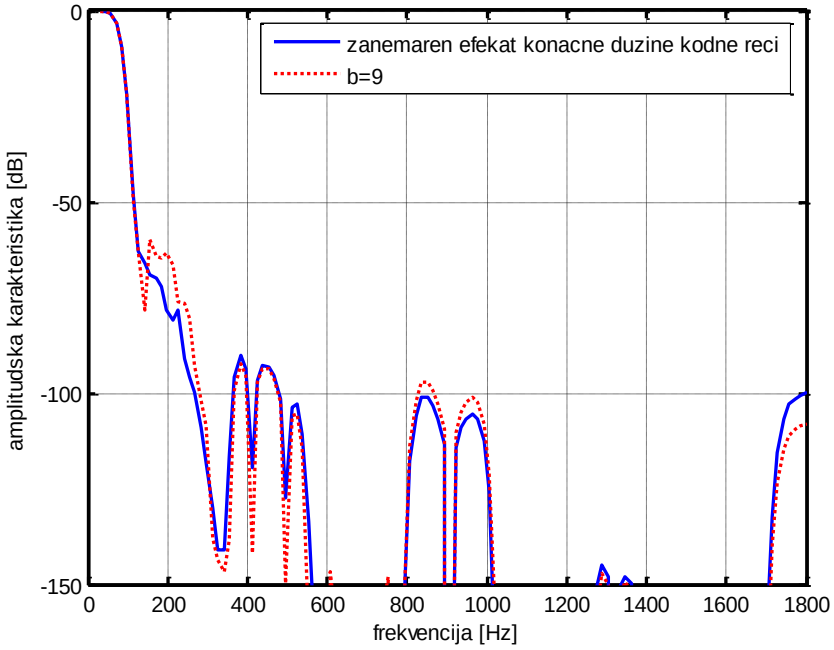


Слика 5.12 Амплитудна карактеристика нове структуре снимана одбирак по одбирак

У нашем примеру су за све филтре (децимационе, интерполационе и *kernel* филтер) били употребљени *FIR* филтри, за које је показано да је максимална грешка коју уноси примена поступка проточне обраде сигнала једнака нули. Из тог разлога може да се донесе закључак да су одступања амплитуске карактеристике последица грешке заокруживања због увећаног броја аритметичких операција у односу на пример на реализацију каскадне везе само два *FIR* филтра, због строгих захтева карактеристике

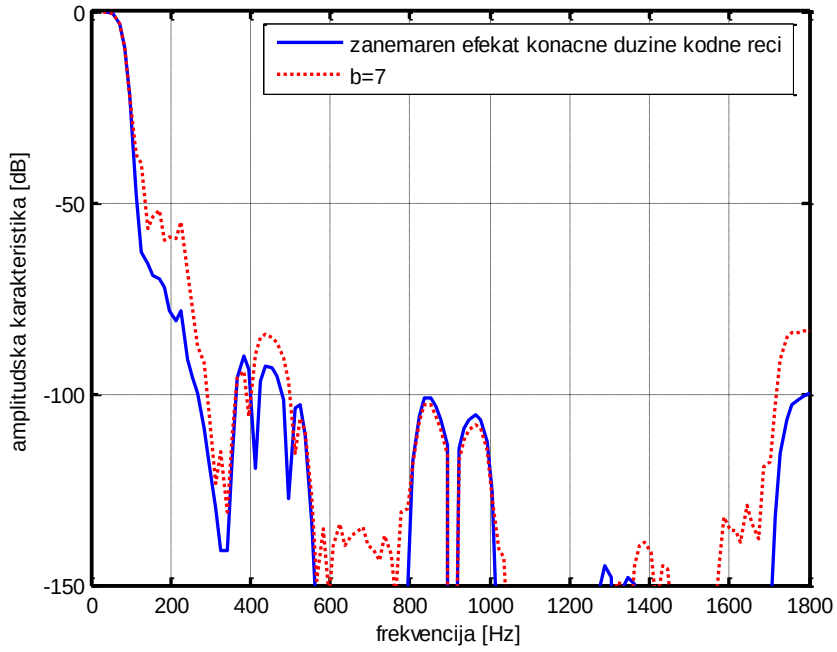
вишестепеног филтра (да су тражене карактеристике реализоване директном применом, ред *FIR* филтра би био јако велики и практично неостварив). Треба имати на уму, да се предложеним поступком врши филтрирање два и више канала, тако да се већи број аритметичких операција односи на број операција по каналу. Из тог разлога би било потребно извршити анализу утицаја коначне дужине кодне речи, како би била извршена проверу нове структуре и како би се упоредили резултати старе и нове структуре.

На сликама 5.13, 5.14 и 5.15 је приказан утицај ефекта коначне дужине кодне речи за 5-битно, 7-битно и 9-битно квантованим коефицијентима добијеним процесом заокруживања. Са слика се види да се утицај грешке квантовања увећава са смањивањем дужине кодне речи.

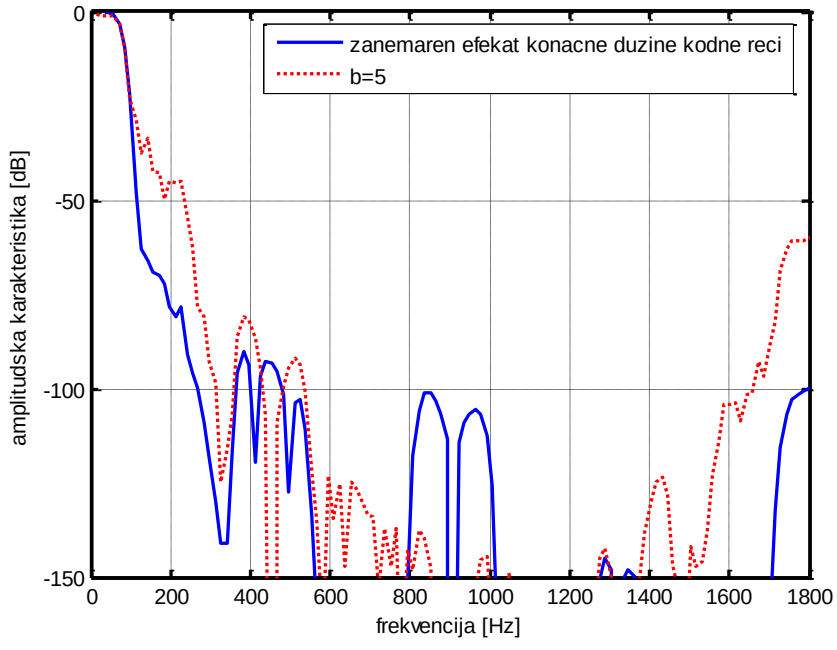


Слика 5.13 Приказ утицаја ефеката коначне дужине (амплитудска карактеристика) речи за нову структуру при b=9

Такође се види да су ефекти коначне дужине кодне речи нешто израженији код реализације вишестепених филтара техником проточне обраде сигнала. Овакав закључак би био и логичан, обзиром да се број аритметичких операција по каналу увећао у односу на реализацију каскадне везе два *FIR* филтра. То значи да број канала не би могао да се увећава у недоглед, јер би грешка квантовања прешла прихватљиве вредности.



Слика 5.14 Приказ утицаја ефеката коначне дужине (амплитудска карактеристика) речи за нову структуру при b=7



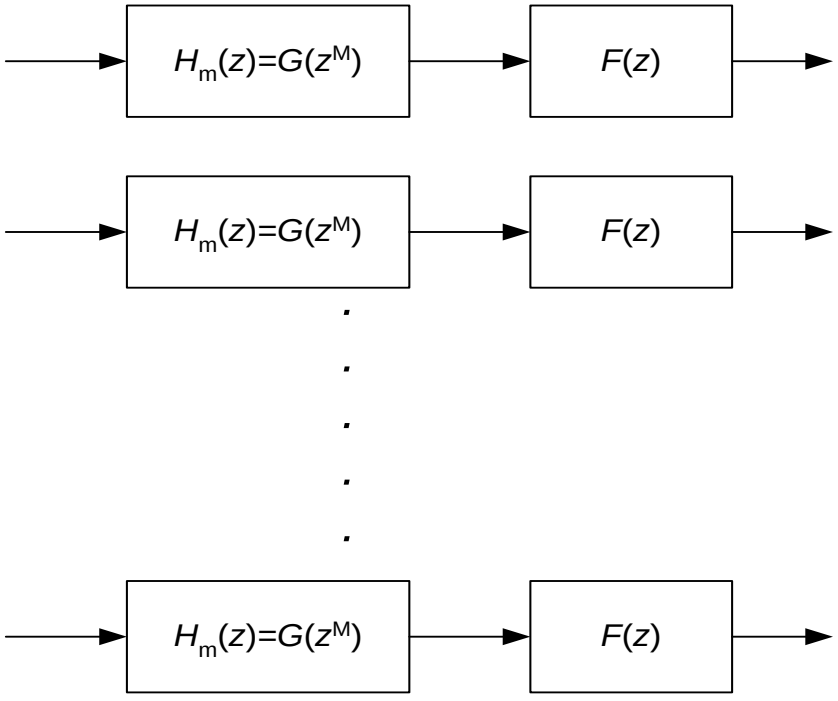
Слика 5.15 Приказ утицаја ефеката коначне дужине (амплитудска карактеристика) речи за нову структуру при b=5

Наравно, ова грешка зависи такође и од захтеване ширине пропусног опсега (због увећања реда филтра), тако да треба правити компромисе између захтева за смањењем ширине пропусног опсега и прелазне зоне са једне стране и броја канала које би требало реализовати предложеном техником.

Из ове анализе може да се донесе закључак, да структура са Слике 5.11 може успешно бити примењена за реализацију више канала са идентичним ускопојасним филтрима. Оваквом применом добијају се знатне уштеде у смислу смањења хардверских ресурса, при чему се код веома уских филтара мора водити рачуна о томе да при већем броју канала који се реализују са једним филтром мора извршити провера утицаја ефекта коначне дужине кодне речи за конкретан случај.

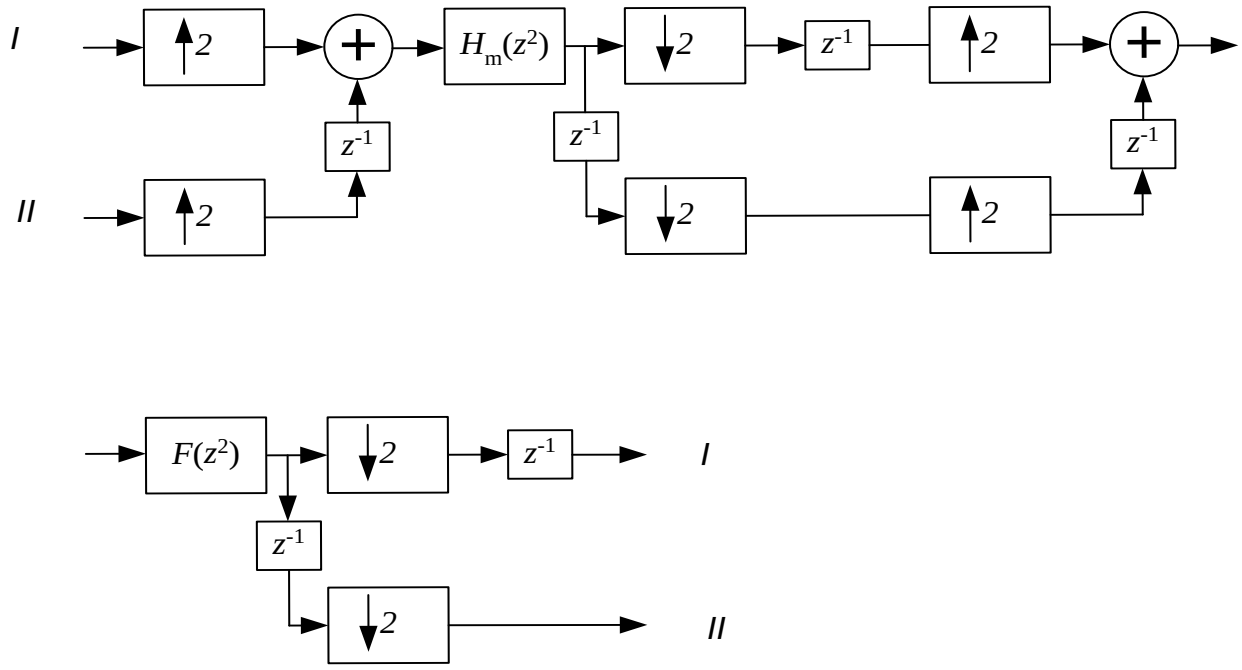
5.2 ФРЕКВЕНЦИЈСКО МАСКИРАЊЕ

Као што је познато, принцип фреквенцијског маскирања се састоји у примени периодичног модел филтра, који настаје заменом сваког кашњења у модел филтру $G(z)$ са M кашњења тј. $H_m(z) = G(z^M)$ и маскирајућег филтра $F(z)$ као што је објашњено у тачки 2.4.2. Уколико се исти филтер користи за филтрирање више канала, добиће се структура као на Слици 5.16.



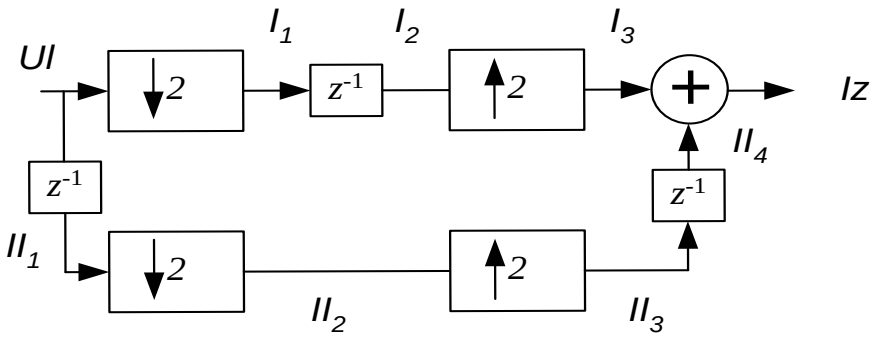
Слика 5.16 Принцип фреквенцијског маскирања за *n* канала

Посматрајући Слику 5.16, може се донети закључак да би *PI* поступак могао да се примени тако што би периодични модел филтри $H_m(z)=G(z^M)$ за све канале били реализовали једним проширеним филтром $H_m(z^K)$, а маскирајући филтри $F(z)$ за све канале такође једним проширеним филтром $F(z)$. После примене *PI* поступка посебно на периодичне модел филтре, а посебно на маскирајуће филтре добија се резултат као што је то приказано на Слици 5.17 (за два канала). Као што се види, део између филтара $H_m(z^2)$ и $F(z^2)$ представља структуру која зависи једино од броја канала. Размотримо улогу поменуте структуре у временском домену.



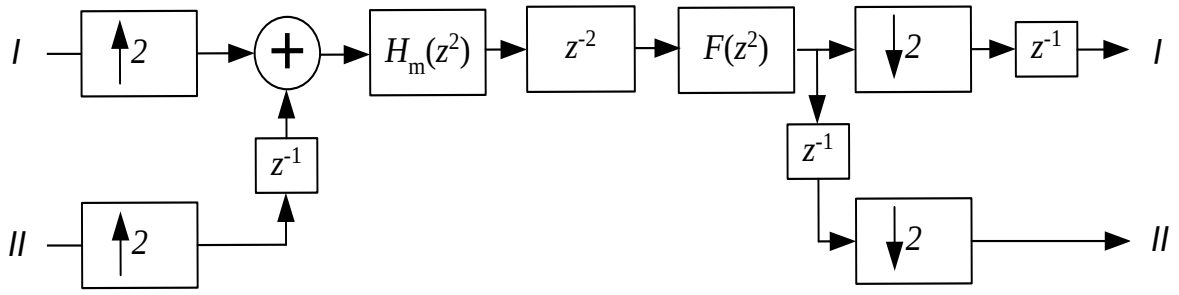
Слика 5.17 Примена *PI* технике на фреквенцијско маскирање за два канала

Уколико се израчунају излази из сваког елемента структуре која је издвојена на Слици 5.18 одбирак по одбирак, за оба канала и на крају изврши сабирање одговарајућих одбирака са улаза *II3* и *II4*, види се да ће излазни сигнал из посматране структуре представљати само улазни сигнал закашњен за два одбирка (у случају канала биће закашњен за одбирака), тако да би поступак фреквенцијског маскирања за два канала на који је примењена *PI* техника на предложени начин, изгледао као што је то приказано на Слици 5.19. Закључује се да би исти овакав резултат био и директно добијен ако би каскадна веза филтара била посматрана као један филтер.



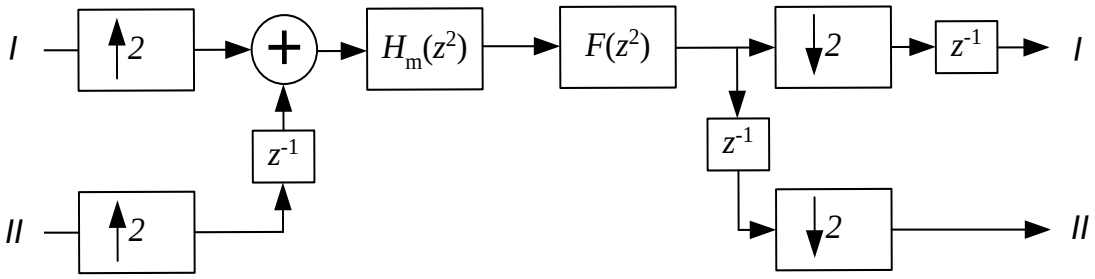
Слика 5.18 Структура између филтара $H_m\left(z^2\right)$ и $F\left(z^2\right)$

Овакав закључак је веома важан, не само на примену *PI* технике на фреквенцијско маскирање, већ генерално на све примене где се реализује *PI* поступак за K канала, а у којима се налазе каскадне везе више филтара. У оваквим случајевима не било неопходно да се укључује у *PI* шему сваки филтер појединачно, већ би сви филтри у једном каналу били посматрани као појединачна структура. На тај начин би знатно била упрошћена почетна примена са Слике 5.17, а изостављањем кола за кашњење између периодичног модел филтра и маскирајућег филтра са Слике 5.19, долази се да коначне реализације поступка фреквенцијског маскирања за два канала као што је то приказано на Слици 5.20.



Слика 5.19 Модификована структура *PI* технике примењене на фреквенцијско маскирање

Потребно је нагласити, да овакво упрошћавање није могуће вршити у случајевима када се заједно са филтрима у каскадној вези налазе и кола за промену учестаности одабирања, зато што излаз из структуре са Слике 5.18 која је замењена колом за кашњење, неће бити само улазни сигнал у структуру померен за два одбирка (у случају K канала K одбирака).



Слика 5.20 Коначна структура *PI* технике примењене на фреквенцијско маскирање

Да би претходно разматрање било проверено на примеру у пракси, биће примењен поступак као у ранијим случајевима и биће извршено снимање карактеристике нове структуре а затим њено упоређивање са карактеристиком која би била добијена директном применом поступка фреквенцијског маскирања (провере ће бити извршена за случај два канала).

Нека је потребно реализовати филтер следећих карактеристика директном применом технике фреквенцијског маскирања:

$Fp = 50\ Hz, Fs = 100\ \text{Hz},$

$\delta_p = 0{,}01,\ \delta_s = 0{,}001\,(60\ d\text{B}),$

$Fo = 2\ 000 Hz$

Нека је $M = 5.$ Значи, модел филтар би био:

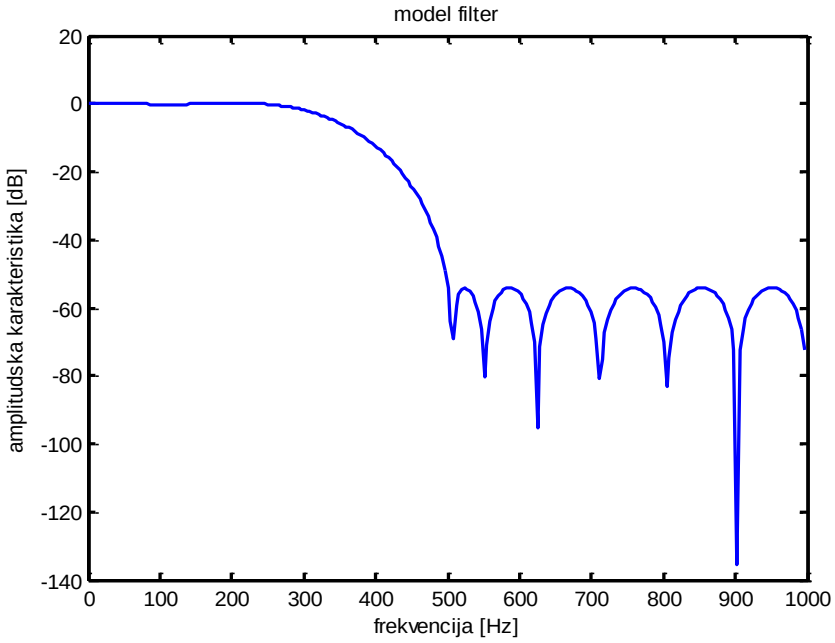
$Fp = 250\ Hz, Fs = 500\ Hz, Fo = 2\ 000 Hz.$

$\delta_p = 0{,}01,\ \delta_s = 0{,}001\,(60\ d\text{B}),$

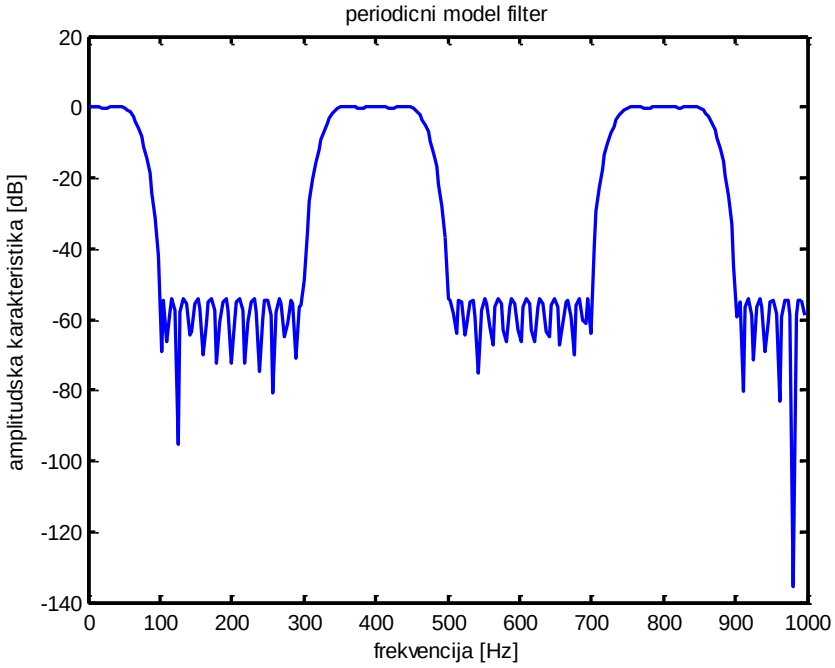
За такве услове се бира:

```
[Nk,fpk,magk,wtk]=remezord([250 500],[1 0],[0.01 0.001],2000);
```

Ред модел филтра би био $Nk = 19$. Његова карактеристика је приказана на слици 5.21. Периодични модел филтер се добија заменом сваког кашњења са M кашњења, а у посматраном случају је $M = 5$. Карактеристика периодичног модел филтра приказана је на Слици 5.22.



Слика 5.21 Амплитудна карактеристика модел филтра

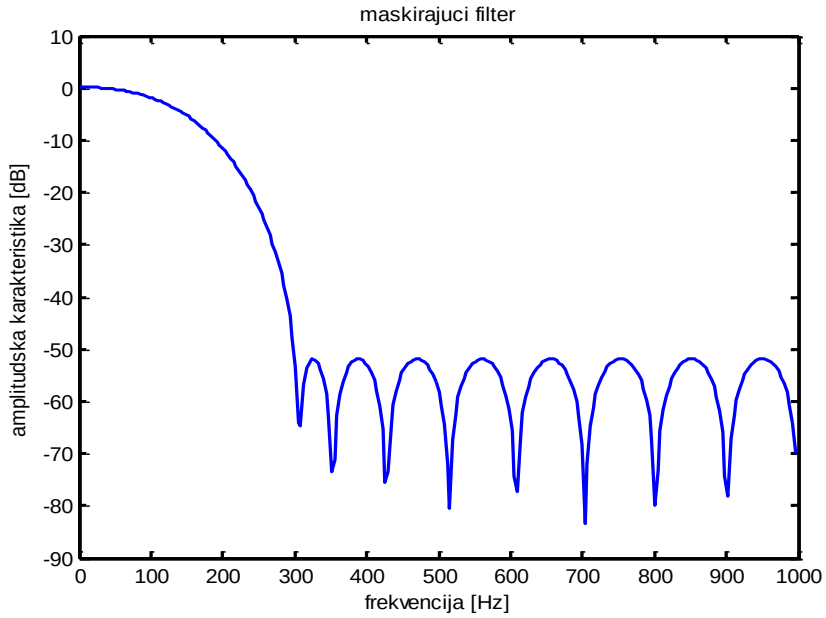


Слика 5.22 Амплитудна карактеристика периодичног модел филтра

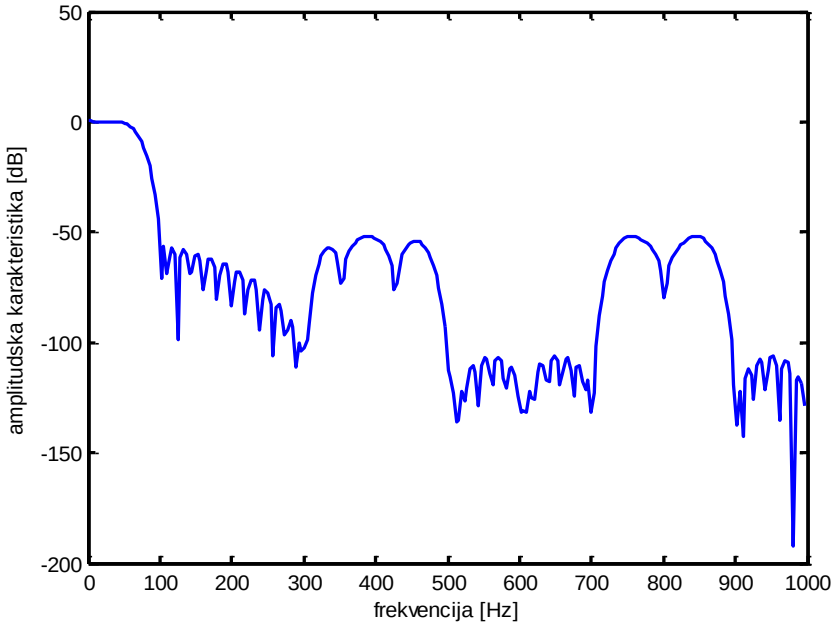
Нека је потрбно да реализујемо НФ филтер. Због тога ће бити задржана само прва компонента, тако да ће бити потребан маскирајући филтер следећих спецификација:

```
[Nm,fpm,magm,wtm]=remezord([50 300],[1 0],[0.01 0.001],2000.
```

Амплитудна карактеристика оваквог филтра је приказана на Слици 5.23, док је амплитудна карактеристика свеукупног филтра приказана на Слици 5.24.



Слика 5.23 Амплитудна карактеристика маскирајућег филтра



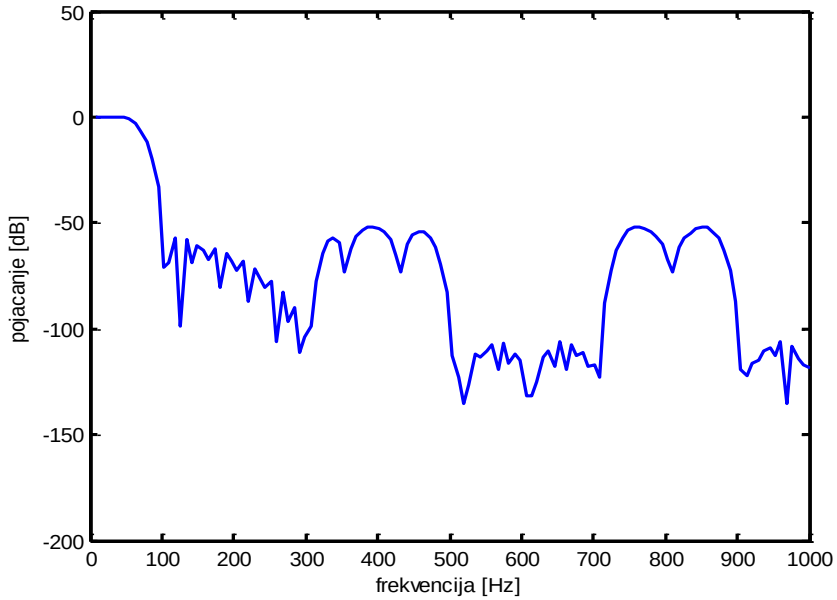
Слика 5.24 Амплитудна карактеристика укупног филтра

Размотримо реализацију поменутог филтра применом технике проточне обраде сигнала на поменути начин. Нека су услови такви да је потребно извршити филтрирање у неколико канала идентичним филтрима реализованих техником фреквенцијског маскирања. Уместо тога, биће примењен поступак проточне обраде сигнала и биће

показано да се филтрирање може извршити једним филтром као што је то приказано на Слици 5.20 за случај два канала.

Карактеристика структуре са Слике 5.20 биће снимљена на исти начин као и у претходним случајевима, тј. на улаз посматране структуре биће доведен синусни сигнал чија се фреквенција мења у опсегу од нула до π и за сваку фреквенцију ће бити одређена тачка по тачка карактеристике.

После извршеног снимања добијена је карактеристика као што је то приказано на Слици 5.25.



Слика 5.25 Амплитудна укупног филтра рачуната одбирак по одбирак

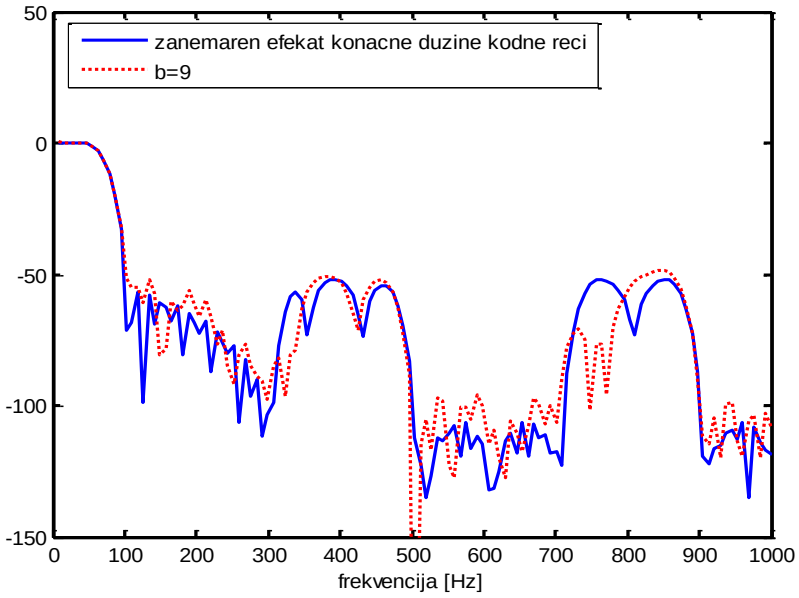
Са слике се види, (овде се ради о директној примени *PI* технике на реализацију два филтра) да је карактеристика готово идентична оној која се добија применом поступка фреквенцијског маскирања и да се грешке односе искључиво на грешке заокруживања, и да предложени поступак даје добре резултате.

Међутим, величина ове грешке, иако би се упоређивањем карактеристика у фреквенцијском домену могло закључити да су оне готово идентичне, би требало испитати и у временском домену. Јаснија слика о величини ове грешке може бити сагледана уколико се израчуна максимална вредност грешке у временском домену за све испитиване фреквенције (опсегу од 0 до π) и за сваки од 1024 улазна одбирка. После симулације у *Matlab*-у добија се да је ова вредност:

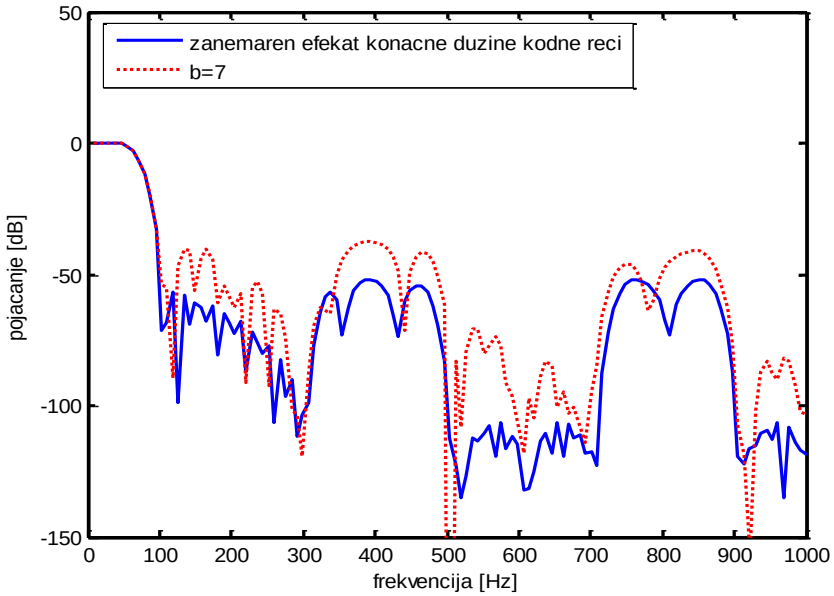
$$E_{\max} = 0$$

То значи да уопште нема разлике између излазних сигнала добијених са и без примене поступка проточне обраде сигнала примењеног на филтрирање техником фреквенцијског маскирања. Имајући на уму поглавље 5.1 где је била исту вредност $E_{\max} = 0$, закључује се је овакав резултат био очекиван због сличности ове две примене. Наиме, једина разлика између реализације каскадне везе два *FIR* филтра и предложене примене проточне обраде сигнала на фреквенцијско маскирање је та што се у периодичном модел филтру свако кашњење мења са M кашњења, док је маскирајући филтер реализован такође као *FIR* филтер, тј. техника проточне обраде сигнала је примењена на каскадну везу два *FIR* филтра.

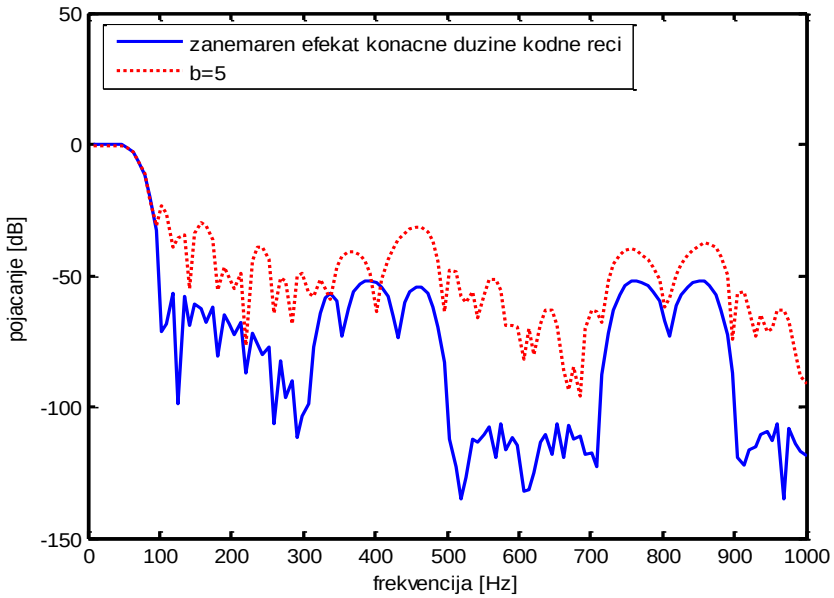
Из истих разлога, утицај ефекта коначне дужине кодне речи би требало да буде занемарљив, као и у случају каскадне везе два *FIR* филтра, што је и приказано на сликама 5.26, 5.27 и 5.28.



Слика 5.26 Приказ утицаја ефеката коначне дужине (амплитудска карактеристика) речи за нову структуру при b=9



Слика 5.27 Приказ утицаја ефеката коначне дужине (амплитудска карактеристика) речи за нову структуру при b=7



Слика 5.28 Приказ утицаја ефеката коначне дужине (амплитудска карактеристика) речи за нову структуру при b=5

Сагласно претходном разматрању, уколико је филтрирање предложеним принципом фреквенцијског маскирања изведено *IIR* филтрима, резултати ће бити аналогни реализацији каскадне везе *IIR* филтара. Наравно, у оба случаја се претпоставља да захтеване карактеристике маскирајућег филтра нису престроге, како не би дошло до

увећања реда филтра и увећања грешке заокруживања због повећаног броја аритметичких операција.

На основу претходног разматрања, може да се донесе закључак да предложени начин реализације технике фреквенцијског маскирања даје резултате идентичне онима до којих би се дошло применом уобичајених метода, уз знатне уштеде хардвеских ресурса. Величина грешке и утицај ефекта коначне дужине кодне речи зависе од броја канала и од избора модел и маскирајућег филтра, али су њихове вредности минималне за уобичајене примене.

5.3 *QMF* БАНКЕ СА СТРУКТУРОМ СТАБЛА

Као што је раније поменуто, поступак проточне обраде сигнала најефектнију примену има у таквим структурама где се јавља вишеструка примена истог филтра. Један од таквих случајева су тзв. вишеканалне *QMF* банке са структуром стабла (енгл. *Tree Structured QMF fulters bank*) [10-12]. На слици 5.29 приказана је осмоканална *QMF* банка са структуром стабла. Са слике се закључује да се филтри банке анализе $H_0(z)$ и $H_1(z)$, као и филтри банке синтезе јављају више пута и да су могући кандидати за примену *PI* поступка. Такође се види, да се учестаност одабирања смањује два пута са сваким увећањем нивоа, при чему се у нивоу 2 и у банци анализе и у банци синтезе процесирају две независне секвенце филтрима $H_0(z)$ и $H_1(z)$ тј. $F_0(z)$ и $F_1(z)$. На нивоу 3 број независних секвенци које се процесирају истим филтром се повећава на четири и тако редом у зависности од броја канала банке. У другом нивоу, на два филтра $H_0(z)$ може бити примењен поступак проточне обраде сигнала и они могу бити реализовани једним филтром $H_0(z^2)$, а исто тако и филтер $H_1(z)$, при чему ће оба филтра имати исти улаз. Ако се настави даље, четири филтра $H_0(z)$ и $H_1(z)$ могу да се реализују са по једним филтром $H_0(z^4)$ тј. $H_1(z^4)$ и тако даље колико банка има канала. На слици 5.30 је приказан начин реализације осмоканалне банке анализе применом поступка проточне обраде сигнала.

На сличан начин се може реализовати и банка синтезе. Четири филтра $F_0(z)$ и $F_1(z)$ могу се заменити филтрима $F_0(z^4)$ и $F_1(z^4)$ респективно, на трећем нивоу банке синтезе. Затим се уместо два филтра $F_0(z^4)$ и $F_1(z^4)$ са другог нивоа употребљавају

филтри $F_0(z^2)$ и $F_1(z^2)$ и на крају долазимо до излазних сигнала *QMF* банке са структуром стабла.

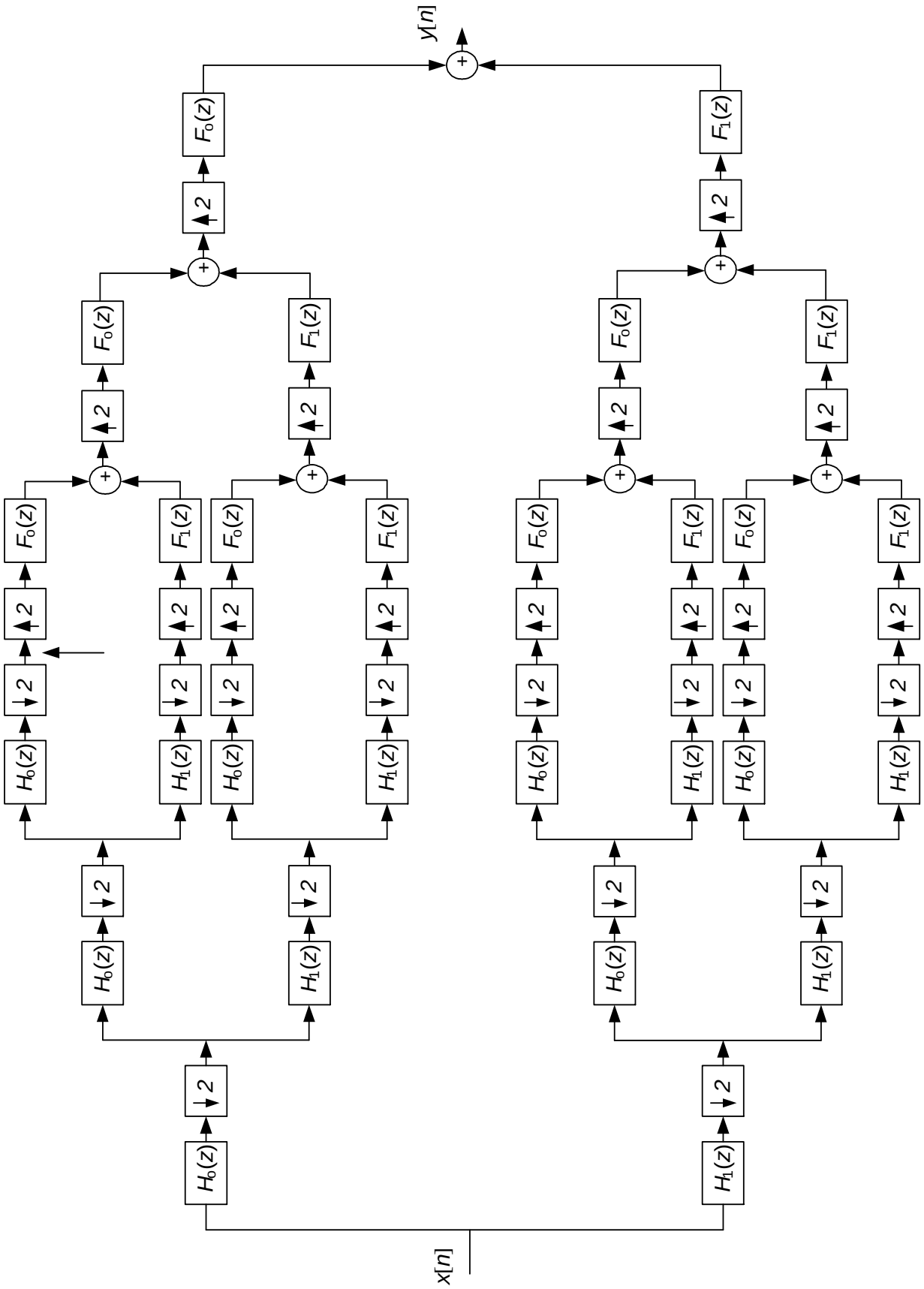
Закључује се да се поступак проточне обраде сигнала једноставно може применити на реализацију свих филтара на појединим нивоима са само два филтра, осим што се морају увести додатни комутатори како би се остварило сједињавање односно раздвајање одбирака за ове филтре. Број потребних регистара остаће непромењен у односу на оригиналну имплементацију, при чему ће они сада да раде на учестаностима једнаким са учестаношћу улазног сигнала.

Провера реализације банке са структуром стабла у *PI* техници може бити извршена на сличан начин као и у претходним случајевима, при чему ће се због сложености структуре (за осмоканалну банку са Слике 5.29 би била потребна реализација укупно 28 филтара) посматрати само једна грана банке анализе, на пример она где се налазе три филтра $H_0(z)$ заредом и уместо снимања амплитудске карактеристике банке упоређивање ће бити вршено у временском домену.

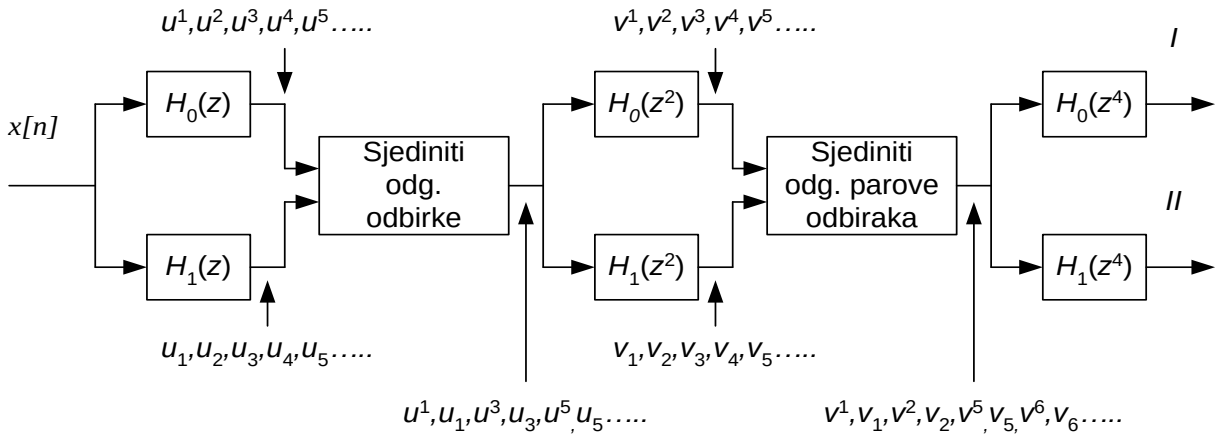
На улаз банке ће бити доведен синусни сигнал чија се учестаност мења у опсегу од 0 до π. За сваку учестаност ће бити одређен излазни сигнал посматране гране. На излазу *I* (Слика 5.29) из модификоване осмоканалне банке анализе морали би се добити сигнали идентични онима који би били добили по оригиналној структури. Треба имати на уму, да ће се на излазу I појавити сложени сигнал који се састоји из четири мултиплексирана излаза из оригиналне структуре и то она која се завршавају филтром $H_0(z)$, док ће се на излазу *II* јавити сигнал мултиплексиран од сигнала као са излаза оригиналне структуре који се завршавају филтром $H_1(z)$. Максимална разлика између вредности одбирака за обе реализације мора бити довољно мала да би се тумачила као грешка заокруживања.

За филтре $H_0(z)$ и $H_1(z)$ ће бити изабрани *halfband* филтри на следећи начин:

```
as = 40;
delta = 10^(-as/20);
h0= firhalfband('minorder',fp,delta);
h1= firhalfband('minorder',fp,delta,'high');
```

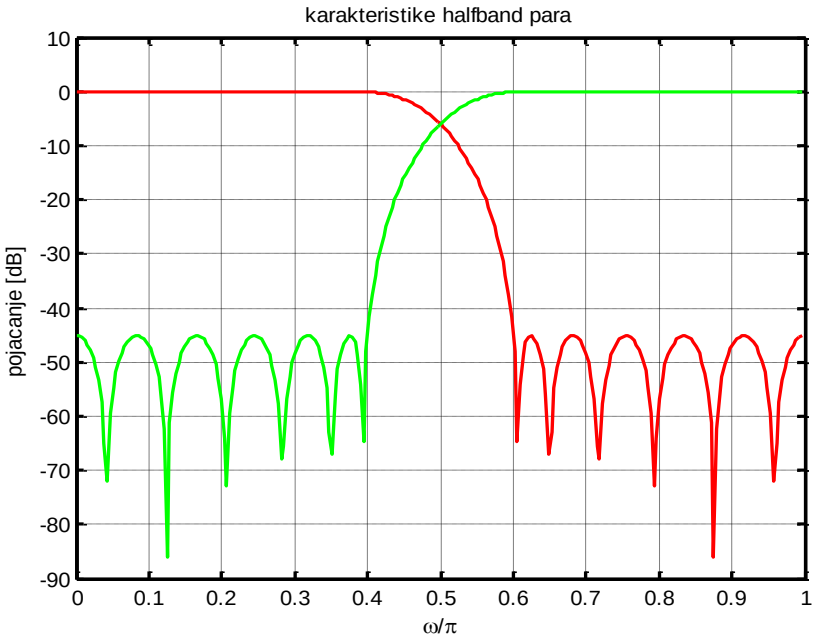


Слика 5.29 Осмоканална *QMF* банка са структуром стабла



Слика 5.30 Реализација у *PI* техници осмоканалне банке анализе

Амплитудске карактеристике овако изабраних филтара $H_0(z)$ и $H_1(z)$ су приказане на Слици 5.31. Приликом одређивања излазних сигнала су коришћена 1024 одбирка, како би био генерисан довољан број излазних одбирака за одређивање карактеристике, због вишеструког снижавања учестаности одабирања.



Слика 5.31 Амплитудске карактеристике филтара $H_0(z)$ и $H_1(z)$

Излазни сигнали посматране гране *QMF* банке у овом случају не могу бити добијени директно израчунавањем конволуције коефицијената филтара због вишеструког снижавања учестаности одабирања, већ ће бити одређени снимањем по принципу ”фреквенцију по фреквенцију”.

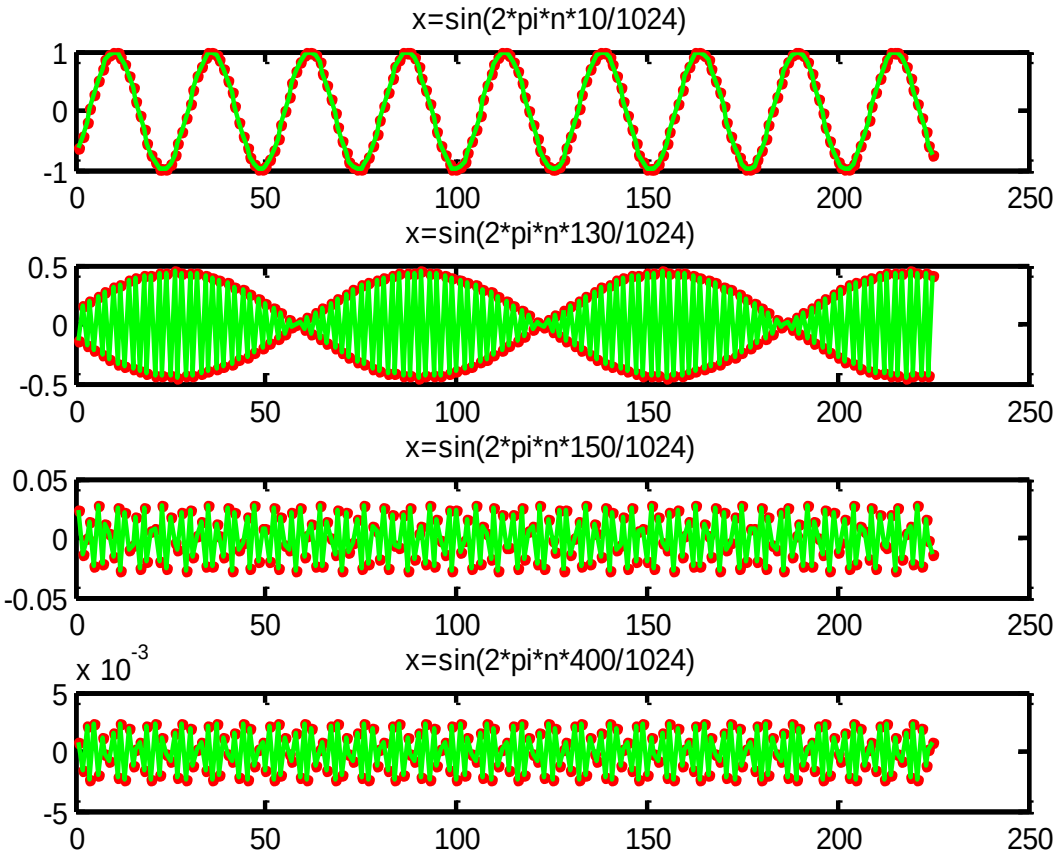
После имплементације оваквог поступка у *Matlab*-у добијени су графици који су приказани на Слици 5.32, а који представљају упоредни приказ излаза из обе структуре на неколико фреквенција из опсега 0 до π (снимање је вршено до улаза у трећи ниво банке анализе). Максимална разлика између одговарајућих одбирака обе структуре за све фреквенције износиће $E_{\max} = 0$

Имајући на уму да је максимална грешка код реализације каскадне везе два *FIR* филтра такође била једнака нули (филтри $H_0(z)$ и $H_1(z)$ су реализовани као *FIR halfband* филтри), долази се до закључка да величина максималне грешке код предложене примене *PI* поступка на *QMF* банке са структуром стабла одговара основној примени на појединачне филтре.

Приказани резултати показују да се поступак проточне обраде сигнала једноставно може применити на реализацију *QMF* банки дигиталних филтара са структуром стабла и да практично не постоје разлике између резултата добијених применом поменуте методе и резултата добијених директном применом. Разлог овоме је тај што је *PI* поступак директно примењен за реализацију групе истоветних филтара за сваки ниво банке анализе (исти резултат би био добијен и за банку синтезе) тј. нема увођења повратних спрега као код реализације каскадних везе филтара и новим поступком се врши само прекомбиновање редоследа операција над одбирцима процесираног сигнала. Због тога и није потребно вршити испитивање утицаја ефеката коначне дужине кодне речи јер ће овај утицај бити истоветан за стару и нову структуру.

Уколико би било потребно да се претходни поступак реализује са *IIR* филтрима, била би наметнута иста она ограничења која су везана за примену *PI* поступка на *IIR* филтре, тј. морали би бити изведени као паралелне структуре са свепропусницима. На крају, због постојања паралелних грана и сложености структура вишеканалних *QMF* банки са структуром стабла, морала би бити проверена *PI* структура у смислу прилагођавања условима критичне петље.

На крају може да се закључи да су *QMF* банке са структуром стабла идеалан кандидат за примену поступка проточне обраде сигнала и да све предности овог поступка долазе до изражаја у њиховом случају. Као што је показано, није потребно вршити никакве модификације нове структуре већ се *PI* поступак може директно применити на реализацију *QMF* банки са структуром стабла, а да при томе добијемо значајне хардверске уштеде.



Слика 5.32 Приказ излаза из обе структуре у временском домену

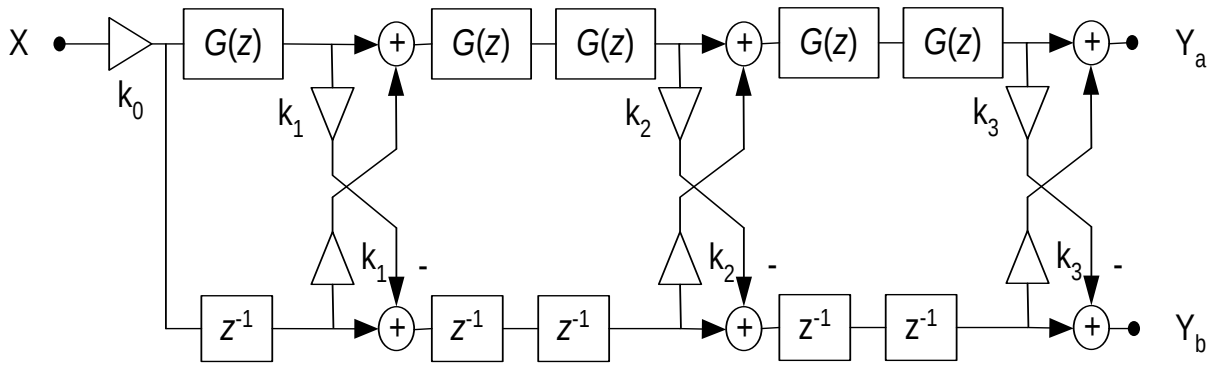
6. НАПРЕДНЕ МЕТОДЕ *MULTIRATE* РЕАЛИЗАЦИЈА УСКОПОЈАСНИХ ДИГИТАЛНИХ ФИЛТАРА

6.1 ФИЛТРИ ЗА ВЕЛИКЕ БРЗИНЕ ОБРАДЕ СА МАЛИМ БРОЈЕМ ЕЛЕМЕНАТА ЗА КАШЊЕЊЕ

У претходним поглављима је показано да се комбинацијом техника обраде сигнала на различитим учестаностима одабирања и поступка проточне обраде сигнала, могу реализовати ефикасни дигитални филтри са бесконачним или коначним импулсним одзивом за потребе реализација телекомуникационих уређаја.

Ефикасност примене поступка проточне обраде сигнала зависи од структуре конкретне реализације. Као што је познато, основни разлог примене проточности обраде јесте да се паралелно обрађују сигнали и тиме избегну реализације у којима се обрада сигнала обавља секвенцијално [13,14]. У прилог овоме је то што се ретко догађа да су учестаност одабирања и брзина обраде сигнала такви да су усаглашени, и то посебно ако се дигитални филтри изводе програмабилним хардвером. Неки произвођачи програмабилног хардвера предлажу решења у којима се користи једна секција другог реда и мултиплексирају се коефицијенти множача и одбирци сигнала. Тиме се један хардверски блок користи као универзални који реализује све секције другог реда. Ово отвара могућност примене технике проточне обраде сигнала, нарочито ако би се користиле идентичне вредности за константе множача и као идентичне структуре.

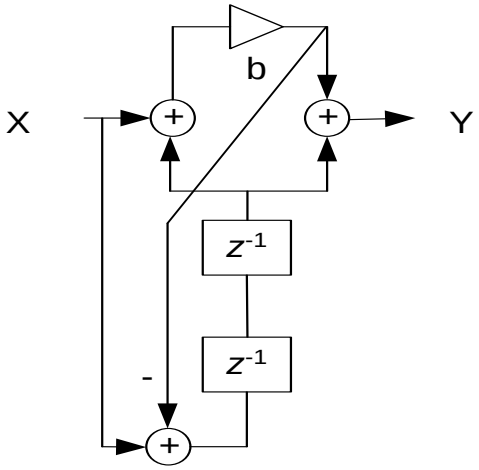
Погледајмо како би изгледала једна оваква примена у случају дигиталног филтра за велике брзине обраде. Структура овог филтра је погодна за процесирање на вишим учестаностима одабирања (*FIR* филтер у комбинацији са *IIR* секцијама другог реда) а приказана је на Слици 6.1.



Слика 6.1 *FIR* филтер у комбинацији са *IIR* секцијама другог реда

Функција преноса филтерске секције са Слике 6.2 је функција типа *allpass*:

$$G(z)=\frac{b+z^{-2}}{1+bz^{-2}} \tag{6.1}$$



Слика 6.2 Основна секција другог реда

Ако би била израчуната функција преноса структуре са Слике 6.1, користећи излаз Y_a био би добијен следећи израз:

$$H(z)=\frac{Y_a(z)}{X(z)}=k_0\begin{pmatrix}(G^5+G^4k_1z^{-1}-G^3k_2(k_1+k_3)z^{-2}+\\+G^2k_2(1-k_1k_3)z^{-3}+Gk_1k_3z^{-4}+k_3z^{-5})\end{pmatrix} \tag{6.2}$$

На основу једначине 6.2, после развијања функције преноса у полином по z^{-1}, види се да би филтер могао бити реализован и на начин приказан на Слици 6.3. Функција преноса филтра за велике брзине обраде би могла бити представљена у форми:

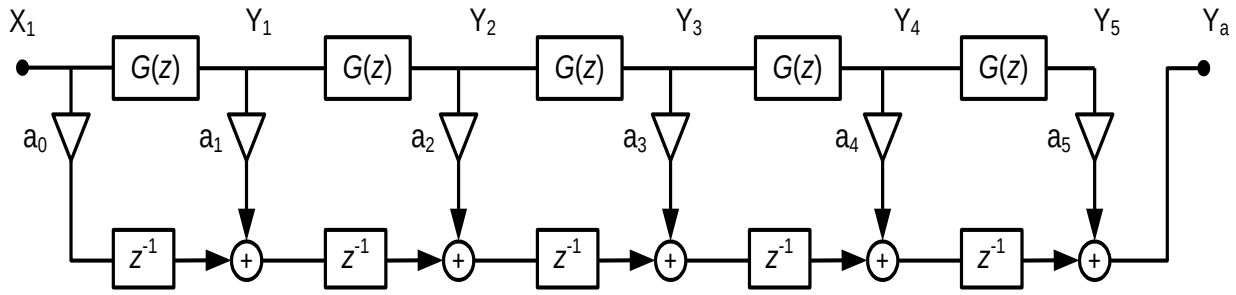
$$H(z)=\frac{Y_a(z)}{X(z)}=a_5G^5+a_4G^4z^{-1}+a_3G^3z^{-2}+a_2G^2z^{-3}+a_1Gz^{-4}+a_0z^{-5}, \tag{6.3}$$

при чему би веза између коефицијената a_0.....a_5 и k_0.....k_5 упоређујући изразе 6.3 и 6.2 била

$$a_5=k_0,\quad a_4=k_0k_1,\quad a_3=-k_0k_2(k_1+k_3),\quad a_2=-k_0k_2(1-k_1k_3), \tag{6.4}$$
$$a_1=-k_0k_1k_3,\quad a_0=k_0k_3.$$

Размотримо сада на који начин би могао бити примењен поступак проточне обраде сигнала на посматрани филтер. Као што је раније речено, *PI* поступак се примењује у

случајевима уколико постоји потреба за филтрирањем идентичним филтром по више канала или уколико се врши филтрирање идентичним филтром више пута заредом [15]. Код обе реализације (Слике 6.1 и 6.3) основна филтерска секција $G(z)$ се јавља пет пута, међутим једино се код реализације према Слици 6.3 сигнал процесира филтерском секцијом $G(z)$ пет пута за редом, док се код реализације са Слике 6.1 филтрирање са $G(z)$ врши највише два пута за редом док би једну филтерску секцију реализовали појединачно. Ово нам говори да би већу ефикасност код примене проточне обраде сигнала у смислу смањења елемената за реализацију постигли уколико би se *PI* поступак применио на реализацију приказану на Слици 6.3.



Слика 6.3 Реализација филтра погодна за проточну реализацију

Нека је потребно реализовати филтер са следећим карактеристикама:

F_p=0,2046; F_s=0,2954; A_p=0,0626 dB; A_s=18,4441 dB,

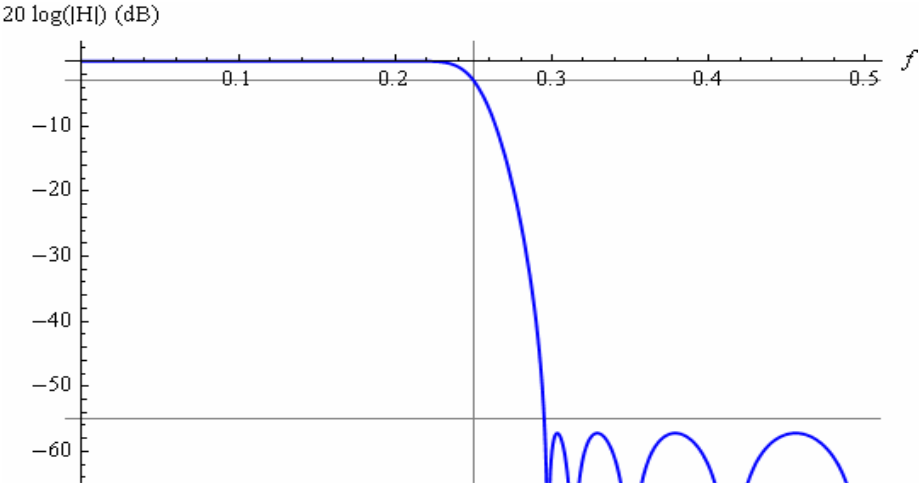
на основу којих су константе множача [14]:

b=1/2 + 1/16; k_0 = 0.24000685, k_1 = 2.37428361, k_2 = –0.54068333, и k_3 = 0.10932683,

тј.

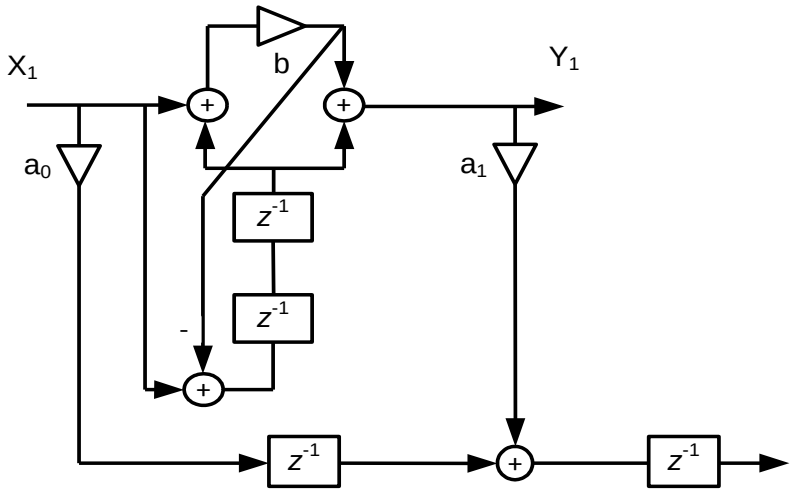
$$\begin{aligned} a_0 &= 0.0262391880887855,\ a_1 = -0.062299274218910634, \\ a_2 &= -0.09608352383954676,\ a_3 = 0.3222924216495991, \\ a_4 &= 0.5698443302427285,\ a_5 = 0.24000685 \end{aligned} \tag{6.5}$$

Амплитудска карактеристика оваквог филтра приказана је на Слици 6.4.



Слика 6.4 Амплитудска карактеристика филтра са Слике 6.2

Заменом основних секција другог реда у структури приказаној на Слици 6.3, долази се до коначне структуре *IIR/FIR* филтра на који ће бити примењена проточна обрада сигнала (на Слици 6.5 је приказан први сегмент оваквог филтра).



Слика 6.5 Први сегмент *IIR/FIR* филтра са основном секцијом другог реда

Функције преноса у тачкама Y_1.....Y_5 износе:

$$\frac{b+z^{-2}}{1+bz^{-2}},\left(\frac{b+z^{-2}}{1+bz^{-2}}\right)^2,\left(\frac{b+z^{-2}}{1+bz^{-2}}\right)^3,\left(\frac{b+z^{-2}}{1+bz^{-2}}\right)^4,\left(\frac{b+z^{-2}}{1+bz^{-2}}\right)^5 \tag{6.6}$$

и представљају функције преноса филтра другог реда на *i*-ти степен, *i*=1,2...5. Ова особина отвара могућност да се филтер на излазу једног филтра другог реда поново процесира овом секцијом *и* пута, тако да је довољно да је реализујемо само једанпут.

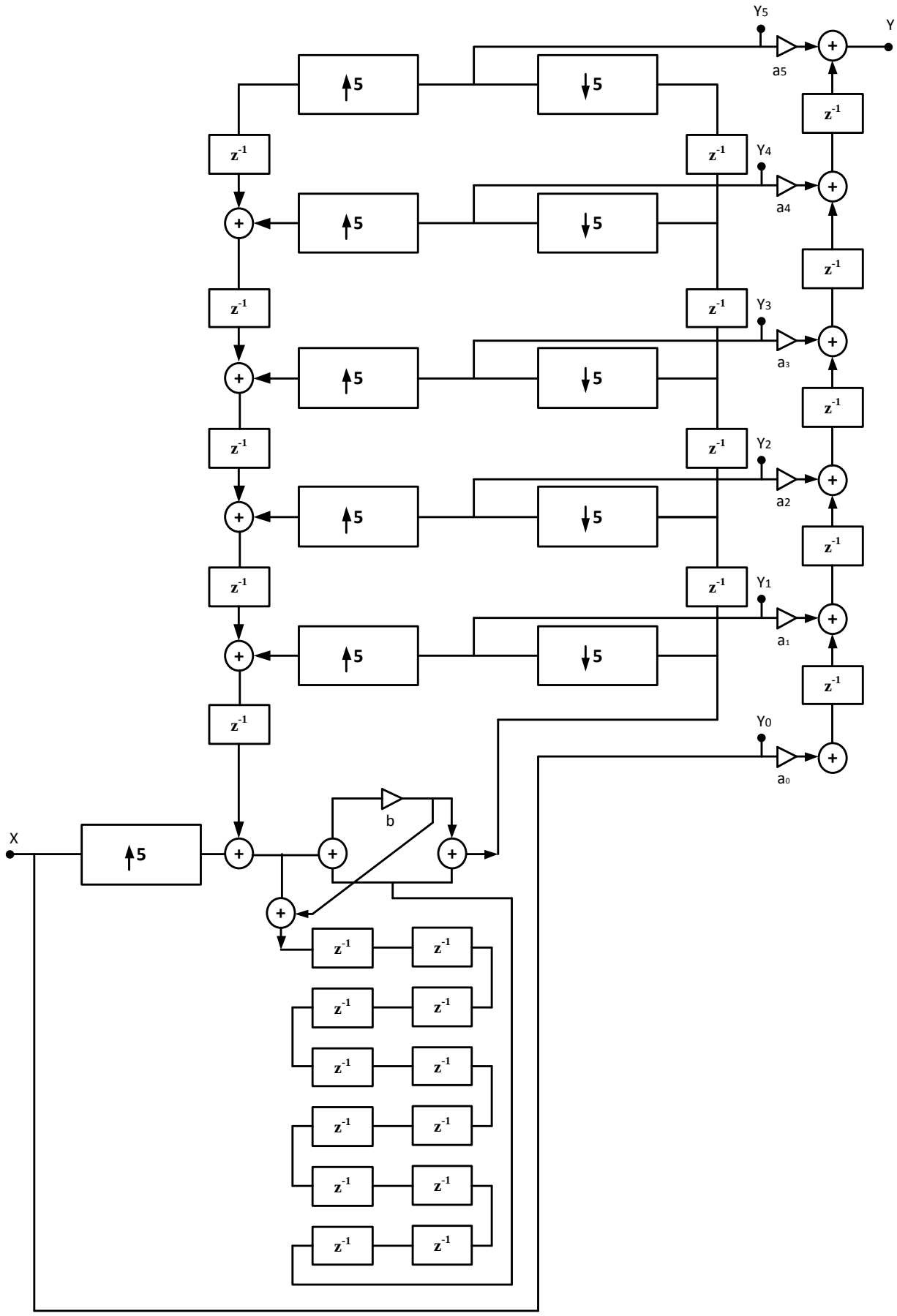
Овако нешто ће бити изведено на тај начин што ће се временски мултиплексирати сигнали на улазу у филтер другог реда, тако да се после сваког улазног одбирка појављује сигнал процесиран једном секцијом другог реда, затим одбирак са два процесирања и тако редом, при чему се сваки излазни одбирак даље обрађује у складу са шемом са Слике 6.3

На Слици 6.6 је приказан комплетан поступак реализације филтра уз коришћење технике са више брзина обраде. На улазни сигнал се примењује операција *upsampling*, чиме се врши убацивање онолико нултих одбирака колики је жељени број обрада тог сигнала истог секција филтра другог реда. Наравно, и број јединичних елемената за кашњење мора бити увећан за исти број нових елемената за кашњење. У посматраном случају, уколико је потребно да се сигнал пет пута процесира истом секцијом другог реда, број нових елемената за кашњење ће бити пет.

На излазни сигнал из филтра се примењује операција *downsampling*, чиме се издвајају они одбирци који су настали од улазног сигнала *X*. Операцијом *upsampling* се убацују нулте вредности а затим се цео сигнал закасни за један одбирак. Сада се овакав сигнал сабира са улазним сигналом на који је већ примењена операција *upsampling*. Тиме се добија да сигнал на улазу у филтер другог реда, који има после сваког улазног одбирка сигнала *X* одбирак сигнала добијен једним процесирањем секцијом другог реда, односно одбирак сигнала Y_1.

Описани поступак понавља се затим још четири пута, при чему се сваки излаз Y_1.....Y_5 множи одговарајућом константом a_0.....a_5 и обрађује сагласно Слици 6.6 да би се добио излазни сигнал Y_a.

Испитајмо сада понашање наше нове *PI* структуре филтра у реалном времену. У претходним поглављима, анализу изведених *PI* структура је вршена снимањем амплитудских карактеристика одбирак по одбирак користећи програмски пакет *Matlab*. Анализа филтра за велике брзине обраде са Слике 6.6 биће извршена на други начин. Користећи програмски пакет *Matematica 6* [16, 17] и одговарајући софтверски алат *SchematicSolver Version 2* [18], за задати улазни сигнал, биће одређени сигнали у тачкама Y_1.....Y_5 за обе структуре, тј. основне са Слике 6.1 и структуре изведене у *PI* техници са Слике 6.6, а затим извршена провера њихове идентичности.



Слика 6.6 Комплетан филтер реализован у *PI* техници

Софтверски алат *SchematucSolver Versuon 2* омогућава кориснику добијање вредности сигнала у било којој тачки на основу дефинисања елемената и њихових међусобних веза, што знатно упрошћава поступак анализе (у посматраном случају улазни сигнал ће прво узети симболичке вредности). Оваква особина *SchematucSolver Versuon 2* нам омогућава елегантну анализу и упоређивање *PI* структуре филтра за велике брзине обраде са Слике 6.6 и полазне структуре филтра са Слике 6.1.

Да би обрада сигнала била извршена потребно је да се генерише улазни сигнал у виду секвенце одбирака. Међутим за филтер са Слике 6.6 потребно је да буду познати како улазни сигнал, тако и они одбирци чије се вредности могу одредити тек након обраде неких од улазних одбирака. Из тог разлога ће бити коришћено симболичко процесирање, тј. најпре ће бити генерисани општи одбирци улазног сигнала а накнадно ће бити одређене и њихове вредности. Ово је могуће у симболичком процесирању зато што се одбирци дефинишу симболима, а стварне вредности могу бити одређене у оном тренутку када су неопходне. На пример, најпре се генерише првих 20 одбирака улазног сигнала на следећи начин:

```
UnитSymbolисSequence[20,r]
```

тако да се добије секвенца чије су вредности симболичке:

```
{{r1},{r2},{r3},{r4},{r5},{r6},{r7},{r8},…,{r20}}
```

Ако је потребно да се симболи замене другим вредностима, то се може урадити тако што се генерише друга секвенца, на пример импулсна побуда

```
x UnитИmpulseSequence[20]
```

што ће дати:

```
{{x},{0},{0},{0},{0},{0},{0},{0},{0},{0},…,{0}}.
```

Команда *UpsampleSequence* убацује потребан број одбирака нулте вредности. Када су познате симболичке вредности одбирака и оне које би требало добити на основу неког правила, листа замена вредности може бити реализована:

```
subs1=ToRules[seq1==seq0]
```

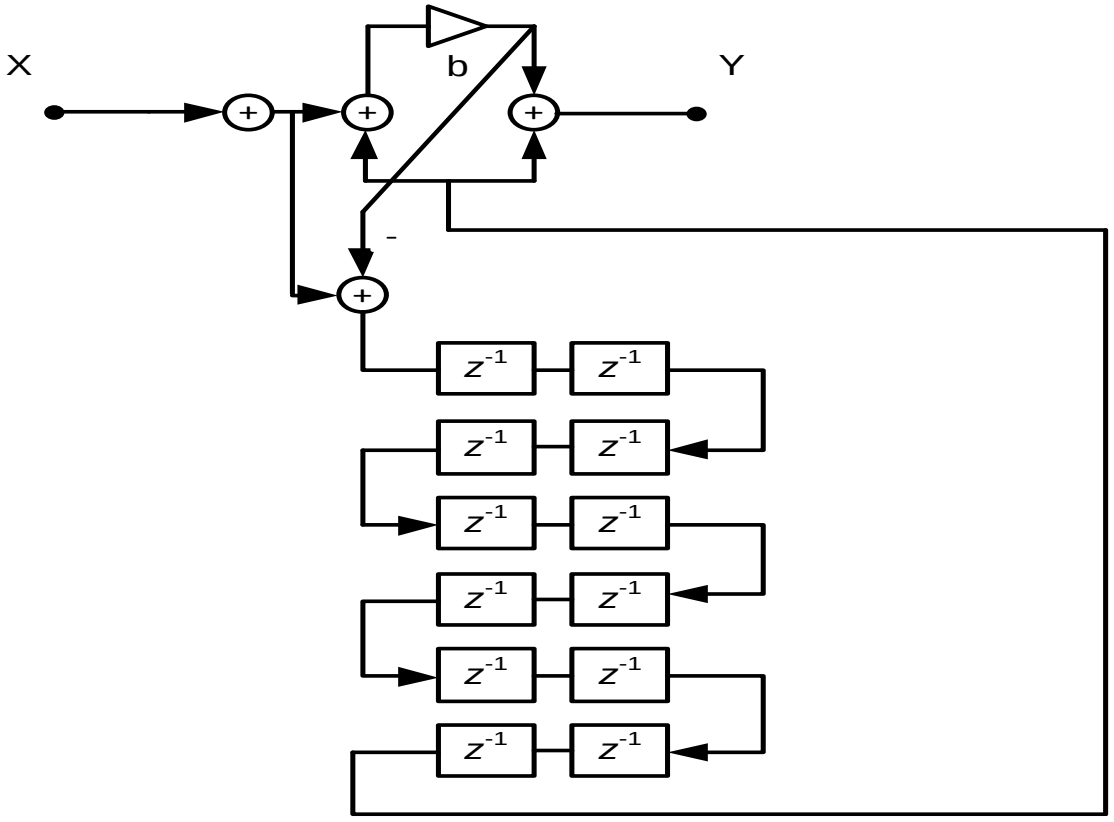
Добија се листа која ће заменити r_1 са x а r_2..... са 0.

```
{r1→x, r2→0, ,…, r20→0}
```

На сличан начин се генеришу остали одбирци улазног сигнала за филтер са Слике 6.6, за 20 одбирака импулсне побуде, тако да је улазни сигнал следећег облика:

```
{{x},{p1},{q1},{w1},{v1},{u1},{0},{p2},{q2},…,{w18}
,{v18},{u18},{0},{p19},{q19},{w19},{v19},{u19},{0},
{p20},{q20},{w20},{v20},{u20}}
```

У добијеном побудном сигналу су познате вредности x и нулте вредности импулсне побуде, а све остале вредности сигнала нису познате јер тек треба да се добију обрадом кроз филтер другог реда. Овако генерисани сигнал се процесира филтром десетог реда, који је приказан на Слици 6.7, добијен од филтра другог реда са додатним кашњењима.



Слика 6.7 Филтер десетог реда

После уношења улазних сигнала и дефинисања елемената структуре, сигнали у тачкама Y_1.....Y_5 се добијају помоћу команде *DownsampleSequence* уз одговарајуће кашњење. Применом операције *upsampling* се добијају сигнали које је потребно поново заменити у побудни сигнал. Тако се на пример за сигнал Y_1 добија:

```
Out0={{b*x},{0},{x-b^2*x},{0},{-(b*x)+b^3*x},{0},
{b^2*x-b^4*x},{0},{-(b^3*x)+b^5*x},{0},
{b^4*x-b^6*x},{0},{-(b^5*x)+b^7*x},{0},
{b^6*x-b^8*x},{0},{-(b^7*x)+b^9*x},{0},
{b^8*x-b^10*x},{0}}
```

Сигнал out0 је сигнал који је у побудном сигналу означен симболом p.

```
{{x},{p1},…,{p2},…,{p19},…,{p20},…{u20}}.
```

Сигнал Y_2 појављује се у следећем облику:

```
out1={b*p1},{b*p2},{p1-b^2*p1+b*p3},
{p2-b^2*p2+b*p4},
{-(b*p1)+b^3*p1+p3-b^2*p3+b*p5},
{-(b*p2)+b^3*p2+p4-b^2*p4+b*p6}, …}
```

Сигнал *out1* је сигнал који је у побудном сигналу означен словом q. Како све вредности морају бити исказане у функцији симбола x и коефицијената филтра, коришћењем команде *ToRules* добија се листа замена и коначне вредности након обраде филтром са Слике 6.6. Тако се на пример за сигнал Y_1 добија:

```
subs1=ToRules[UnitSymbolicSequence[20,p]==out0]
{p1→b x,p2→0,p3→x-b^2 x,p4→0,p5→-b x+b^3 x,p6→0,p7→b^2 x-b^4
x,p8→0,p9→-b^3 x+b^5 x,p10→0,p11→b^4 x-b^6 x,p12→0,p13→-b^5 x+b^7
x,p14→0,p15→b^6 x-b^8 x,p16→0,p17→-b^7 x+b^9 x,p18→0,p19→b^8 x-b^10
x,p20→0}
```

За посматране зачке Y_1.....Y_5 након упоређивања резултата добијених за обе реализације добија се:

```
outseqHS1-out0
{{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0}
,{0},{0},{0},{0}};

outseqHS2-out1/.subs1//Expand
{{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0}
,{0},{0},{0},{0}};

outseqHS3-out2/.subs2/.subs1//Expand
{{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0}
,{0},{0},{0},{0}};

outseqHS4-out3/.subs3/.subs2/.subs1//Expand
{{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0}
,{0},{0},{0},{0}}

outseqHS5-out4/.subs4/.subs3/.subs2/.subs1//Expand
{{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0}
,{0},{0},{0},{0}};
```

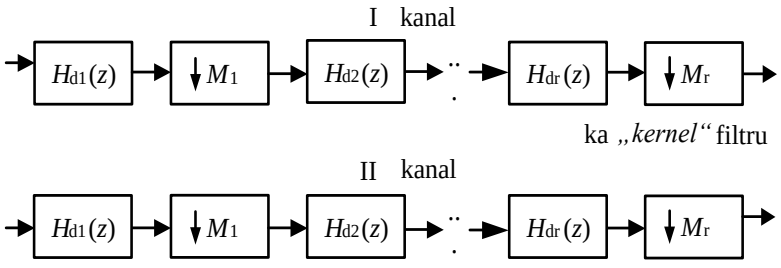
Види се да је симболичко процесирање дало очекивани резултат који је веома важан. Упоређивање резултата у тачкама Y_1.....Y_5 показује да су симболичке вредности добијене процесирањем филтрима са Слика 6.1 и 6.6 потпуно идентичне. Такође, показује се да сигнали у посматраним тачкама садрже искључиво симболичке вредности из одговарајућег канала временски мултиплексованог сигнала, што доказује да нема преслушавања канала.

На основу добијених резултата можемо да се закључи да је предложени начин реализације филтра за велике брзине обраде у *PI* техници исправан и да су вредности сигнала у карактеристичним тачкама структуре идентичне онима који би били добијени класичним поступком. Поред овога, јако важна особина предложеног поступка је та што је потребна само једна основна филтерска секција другог реда уместо пет, тако да се постиже значајна уштеда потребних елемената за реализацију.

6.2 УСКОПОЈАСНИ ФИЛТРИ СА *IIR* КОМПЛЕМЕНТАРНИМ ФИЛТЕРСКИМ ПАРОВИМА

Ускопојасни дигитални филтри, код којих је ширина пропусног опсега реда 1/20 учестаности одабирања представљају велики изазов за пројектанте дигиталних филтара, и поред значајног технолошког развоја дигиталних чипова на којима се врши њихова имплементација. Разлог за то је веома висок ред филтара који је потребан да би се остварила жељена карактеристика [19, 20]. У случају стандардних *FIR* структура, такве карактеристике није могуће реализовати, обзиром да би ред филтра износио неколико стотина. Тако висок ред *FIR* филтра у комбинацији са реализацијом у техници са фиксним зарезом би генерисао грешку која је за већину примена неприхватљива. Такође, претерано велики број елемената за кашњење би условио веома спор одзив филтра. Код реализација оваквих ускопојасних филтара користећи *IIR* структуре, ред филтра би био вишеструко мањи, али би и даље остао неприхватљиво висок за већину примена. Такође, захтев за очувањем линеарне фазне карактеристике не би могао бити остварен.

Уобичајено решење у последње време, за реализацију овако ускопојасних филтара је примена неких од *multirate* метода [21], која ће омогућити ефикаснију реализацију, и смањити број аритметичких операција потребних за имплементацију филтра. Као што јео показано у поглављу 5.1, једно од решења за јако ускопојасне филтре могу бити *multistage* дигитални филтри, чијом применом се ред филтра може значајно смањити [22]. Ако би било настављено увећавање децимационог/интерполационог фактора, ширина пропусног опсега би била све ужа и ужа. У случају да је децимациони/интерполациони фактор велики (на пример већи од 8), показало се [1] да је ефикаснија реализација у више степена, на пример за $M = 8$ реализација са три степана $M = 2$, као што је то приказано на Слици 6.8.



Слика 6.8 Вишестепена децимација

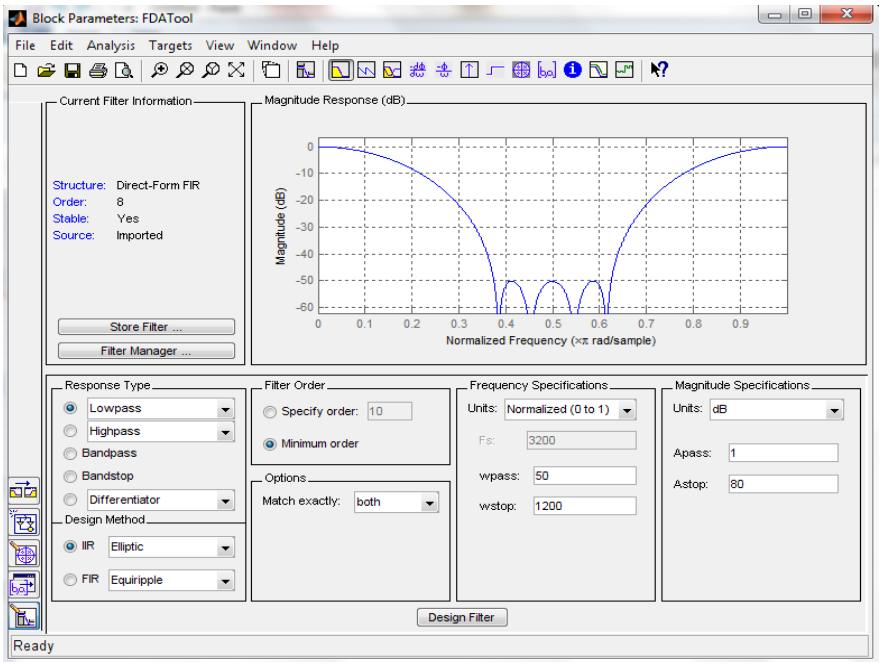
Са даљим увећањем децимационог/интерполационог фактора, као што ће бити показано, осим смањења пропусног опсега, јавља се и деградација амплитудске карактеристике у прелазној зони, а која је условљена грешком коју уносе децимациони/интерполациони филтри $H_{d1}(z).....H_{in}(z)$, који су неопходни код вишеструке децимације и интерполације ради спречавања *aliysing*-а код децимације и уклањања периодичних одраза код интерполације. Такође, показује се да је осетљивост функције преноса на промену параметара елиптичких *IIR* филтара јако висока [23,24] , тако да се код ове класе филтара ограничења односе на максималну вредност децимационог/интерполационог фактора .

У циљу испитивања реализација вишестепених *multirate* филтара, биће извршена имплементација вишестепеног филтра на *Xilinx*-овом *FPGA* чипу Virtex 7 користећи *Xilin*-ов програмски пакет *System Generator* [25]. Као и у предходним поглављима, снимање амплитуске карактеристике ће бити извршено по принципу одбирак по одбирак, тако што ће се формирати скрипта користећи програмски пакет *Matlab* [26, 27] која ће генерисати одбирак по одбирак за скуп фреквенција из опсега *0* - π, и сваки одбирак ће се процесирати кроз структуру вишестепеног филтра из *System generator*-а . За сваку појединачну фреквенцију, биће одређена једна тачка функције преноса, и за цео скуп тачака на крају ће бити формирана амплитудска карактеристика посматране дигиталне структуре.

На почетку ће бити реализован *Kernel* филтер као елиптички филтер следећих карактеристика:

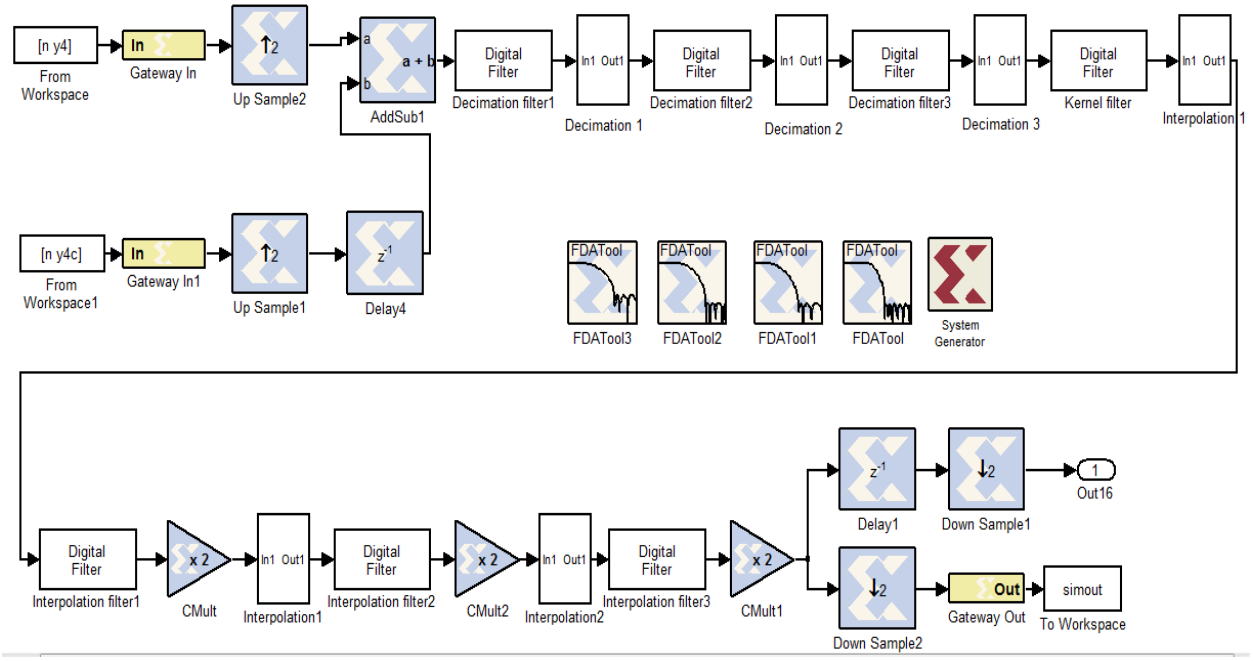
```
[N,wn]=ellupord( 0.2137497191, 0.2909027570,0.1,60);
[b,a]=ellup(N,0.1,60,wn);
```

То ће бити филтер седмог реда, са четири секције другог реда, реализованих као „*Direct-Form II Transposed*" structure. Такви филтри се у програмском пакету *System Generator* могу реализовати користећи софтверски алат *FDA tool* (Слика 6.9), а који директно генерише коефицијенте филтра из задатих карактеристика. Реализација оваквог вишестепеног *multirate* филтра приказана је на Слици 6.10 а амплитудска карактеристика филтра за $M=1$ и $M=5$ приказана је на Слици 6.11.

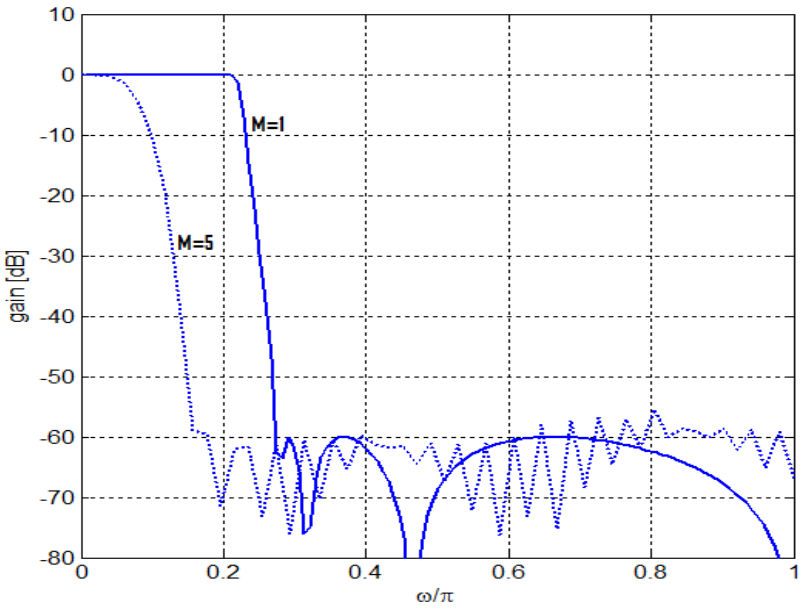


Слика 6.9 Реализација *Kernel* филтра коришћењем алата *FDA tool*

Као што се види са Слике 6.11, са увећањем фактора M долази до деградације прелазне зоне, и даље сужавање карактеристика филтра неће дати очекиване резултате. То значи да *Kernel* филтер, који је једини у оваквој структури подложан модификацији, треба модификовати на такав начин да карактеристика филтра у прелазној зони буде очувана.



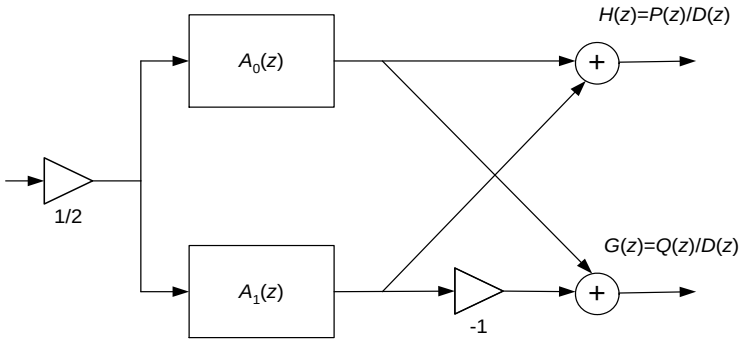
Слика 6.10 *FPGA* реализација вишестепеног филтра за два канала у *PI* техници



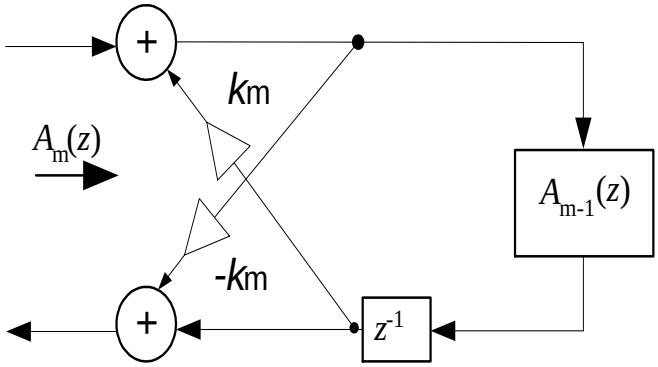
Слика 6.11 Карактеристика вишестепеног филтра са *IIR Kernel* филтром

6.2.1 ВИШЕСТЕПЕНИ ФИЛТРИ СА *ALL-PASS* РЕАЛИЗАЦИЈОМ *KERNEL* ФИЛТРА

Један од начина за побољшање карактеристика вишестепених филтара са великим децимационим/интерполационим фактором у прелазној зони је имплементација *Kernel* филтра користећи паралелне свепропусне секције (Слика 6.12) а које су реализоване као каскадне лествичасте структуре (Слика 6.13) [28]. Обзиром да овакви филтри имају мању осетљивост на промене коефицијената филтра, очекују се бољи резултати код употребе овако реализованог *Kernel* филтра у сложеној структури са Слике 6.



Слика 6.12 Паралелна реализација са свепропусним секцијама



Слика 6.13 Каскадна лествичаста структура

Нека су спецификације идентичне као и у случају директне реализације:

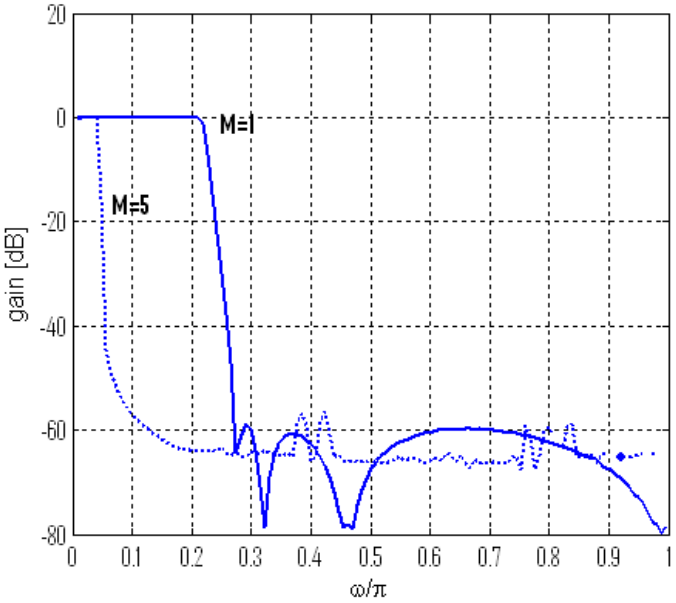
[N,wn]=ellupord(0.2137497191, 0.2909027570,0.1,60);
[b,a]=ellup(N,0.1,60,wn);

Коефицијенти свепропусних секција ће бити:

[k1,k2] = tf2cl(b,a);

Филтер ће бити седмог реда, *cut-off* фреквенције $\omega_c = 0{,}25\pi$, са три секције свепропусника другог реда и једном секцијом првог реда. У поређењу са другим реализацијама (на пример одговарајући *Butterworth*-ов *IIR* филтер би био 26-тог реда), постигнута су одређана побољшања у смислу снижавања реда филтра, и ако би такав филтер био искоришћен као *Kernel* филтер, амплитудска карактеристика вишестепеног филтра би требала бити такође побољшана.

Након снимања амплитудске карактеристике вишестепеног филтра са оваквим *Kernel* филтром, добијају се карактеристике као што је приказано на Слици 6.14. Као што се види, дошло је до побољшања амплитудске карактеристике укупног филтра, мада су још увек таква побољшања недовољна, обзиром на ширину прелазне зоне.



Слика 6.14 Амплитудска карактеристика вишестепеног филтра са *Kernel* филтром са свепропусним секцијама са каскадном лествичастом структуром

6.2.2 ВИШЕСТЕПЕНИ ФИЛТРИ СА *KERNEL* ФИЛТРОМ РЕАЛИЗОВАНИМ СА *IIR* КОМПЛЕМЕНТАРНИМ ФИЛТЕРСКИМ ПАРОВИМА

Овај приступ су прво представили Сарамаки и Ренфорс [29], а касније је техника проширена на реализацију комплементарних филтерских парова [30]. Концепт *IIR* комплементарних филтерских парова има значајну примену код пројектовања основних градивних блокова вишеканалних банака дигиталних филтара [31] , појединачно, као и код примена у пару са техником фреквенцијског маскирања. Популарност примене ове методе, осим због хардверски ефикасне реализације, проистиче од ниске осетљивости амплитудске карактеристике на промене коефицијената филтра [32]. Ова особина је јако битна у случајевима вишеструке обраде сигнала истим филтром, када је грешка услед *fixed-point* имплементације нарочито изражена.

У случају вишестепених филтара, са децимацијом-интерполацијом у више степена, оваква реализација *Kernel* филтра би требало да даје амплитудску карактеристику са знатно смањеном грешком у прелазној зони, где је стрмина карактеристике најизраженија. Из тог разлога ће овакав концепт бити примењен на реализацију ускопојасних вишестепених филтара са истим захтевима као и у предходним случајевима.

Укупни задатак филтрирање сигнала је подељен између два филтра, *FIR* прототип филтра ниског реда $F_{LP}(\omega)$, и *EMQF* (*elliptic minimal Q-factors*) *IIR* филтра $G_{LP}(z)$. Улога *FIR* прототип филтра је да обезбеди захтевано слабљење у пропусном и непропусном опсегу,

док је улога *IIR* прототип филтра да обезбеди граничне учестаности и квалитет прелазне зоне. Ниска осетљивост укупног филтра на промену коефицијената филтра се обезбеђује употребом лествичасте структуре са свепропусним секцијама, а које су реализоване као *Class II* і *Class III* комплементарни филтерски парови [33]. У нашем случају, за реализацију *Kernel* филтра пропусника ниских учестаности ћемо искористити нископропусну грану комплементарног филтерског пара, у циљу корекције грешке услед имплементације са фиксним зарезом.

Комплементарни филтерски пар класе *III* $[H_{LP}(z)]$ $[H_{HP}(z)]$, задовољава оба услова комлементарности, тј. и *All-pass* и амплитудски услов комплементарности. Амплитудска одзив $[H_{HP}(z)]$ осцилује у опсегу $[1,\ 1-\delta]$ у пропусном, и $[0,\ \delta]$ у непропусном опсегу, где је $\delta = 10 - \mathrm{A_s}/20,$ а $\mathrm{A_s}$ минимално слабљење у непропусном опсегу. Функција преноса нископропусне гране комплементарног пара износи:

$$H_{LP}(z) = \sum_{n=0}^{L} a_{LP}(n)[A_0(z)]^n [A_1(z)]^{L-n} \tag{6.7}$$

где је L двострука вредност целобројног умношка, а $A_0(z)$ и $A_1(z)$ су свепропусни филтри који задовољавају услове комплементарности. Коефицијенти $a_{LP}[n]$ су константе *half-band* филтра $F_{LP}(w)$ L-тог реда линеарне фазе, чији је фреквенцијски одзив за нулту фазу не-негативна функција фреквенције.

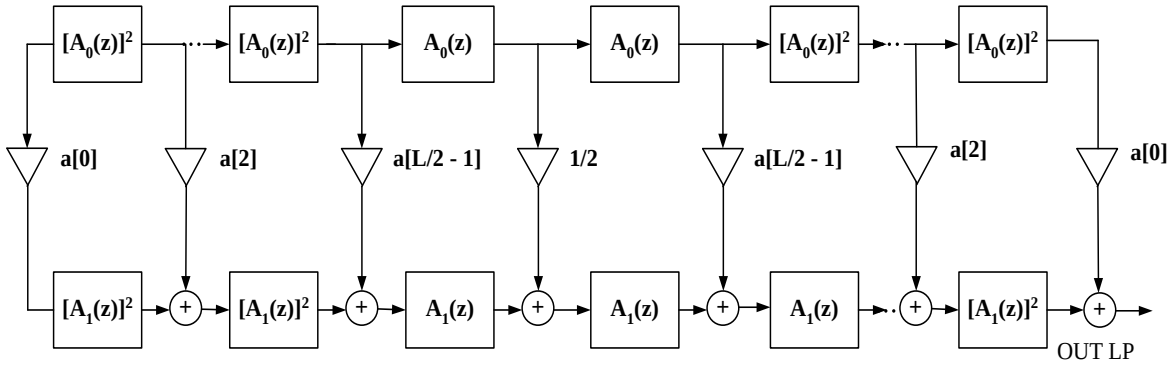
FIR прототип филтер $F_{LP}(w)$ задржава исто минимално слабљење у непропусном опсегу као и свеукупни филтер $H_{LP}(z)$, такође задржавајући и услове комплементарности.

За захтевано минимално слабљење у непропусном опсегу као у предходним примерима $A_{sFIR} = 60dB$, потребно је да *FIR* прототип филтер линеарне фазе буде шестог реда, тј. $L = 6$. За *IIR* прототип филтер се бира да филтер буде трећег реда, за кога је минимално слабљење у непропусном опсегу $A_{sIIR} = 14{,}53dB$, а за граничну фреквенцију $\omega_c = 0{,}1\pi$ коефицијенти свепропусних секција ће износити:

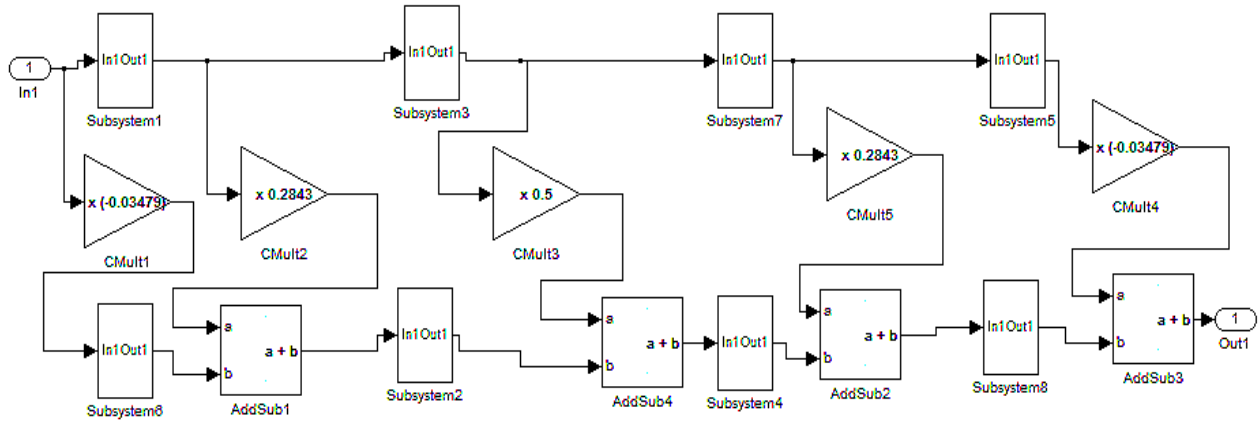
$$A_0(z) = \frac{0.8769377 + 1.7850739\, z^{-1} + z^{-2}}{1 + 1.7850739\, z^{-1} + 0.8769377\, z^{-2}} \tag{6.8}$$

$$A_1(z) = \frac{-0.7265425 + z^{-1}}{1 - 0.7265425\, z^{-1}} \tag{6.9}$$

Имплементациона структура оваквог филтра приказана је на Слици 6.15, а реализација оваквог филтра у *Xilinx*-овом алату *System Generator*-у приказана је на Слици 6.16.



Слика 6.15 Имплементациона структура *Kernel* филтра



Слика 6.16 Реализација *Kernel* филтра у програмском алату *System Generator*

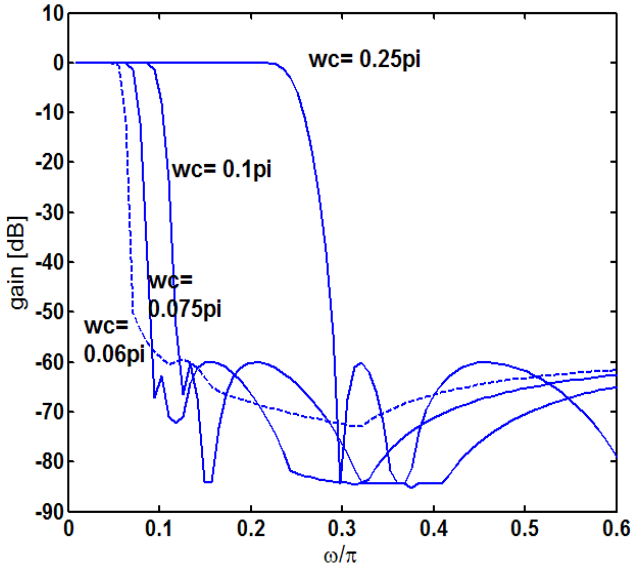
За спецификације филтра као у предходним случајевима и за граничну фреквенцију $\omega_c = 0{,}1\pi$, параметри филтра (коефицијенти *FIR* прототип филтра $a_{LP}[n]$ и коефицијенти *IIR* прототип филтра α, α_1, и β_2) су приказани на Слици 6.1.

За друге граничне фреквенције ω_c, коефицијенти *FIR* прототип филтра задржавају исте вредности, док се параметри *IIR* прототип филтра (α, α_1, и β_2) мењају како би задовољили тражене граничне услове. Карактеристике овако реализованог филтра, за неколико граничних фреквенција ($\omega_c = 0{,}25\pi$, $\omega_c = 0{,}1\pi$, $\omega_c = 0{,}075\pi$ и $\omega_c = 0{,}06\pi$) реализованог према Слици 6.16 за *Xilinx FPGA Virtex7* серију и 32-битну прецизност су приказане на Слици 6.17.

Коефицијенти *FIR* прототип филтра	
$a_{LP}[0] = -a_{HP}[0] = a_{LP}[6] = -a_{HP}[6] = -0.034777744$ $a_{LP}[2] = -a_{HP}[2] = a_{LP}[4] = -a_{HP}[4] = 0.284277744$ $a_{LP}[3] = a_{HP}[3] = 0.5$ $a_{LP}[1] = a_{HP}[1] = a_{LP}[5] = a_{HP}[5]=0$	
Коефицијенти *IIR* прототип филтра	
ω_c	$0.1\ \pi$
ω_p	$0.084450\ \pi$
ω_s	$0.118301\ \pi$
A_{pIIIR}	$0.156\ dB$
A_{sIIIR}	$14.53\ dB$
α_1	-0.726542528
α	-0.951056516
β_2	0.876937785

Табела 6.1 Коефицијенти *Kernel*филтра реализованог преко *Class III* комплементарних филтерских парова

За друге граничне фреквенције ω_c, коефицијенти *FIR* прототип филтра задржавају исте вредности, док се параметри *IIR* прототип филтра (α, α_1, и β_2) мењају како би задовољили тражене граничне услове. Карактеристике овако реализованог филтра, за неколико граничних фреквенција ($\omega_c = 0{,}25\pi$, $\omega_c = 0{,}1\pi$, $\omega_c = 0{,}075\pi$ и $\omega_c = 0{,}06\pi$) реализованог према Слици 6.16 за *Xilinx FPGA Virtex7* серију и 32-битну прецизност су приказане на Слици 6.17.

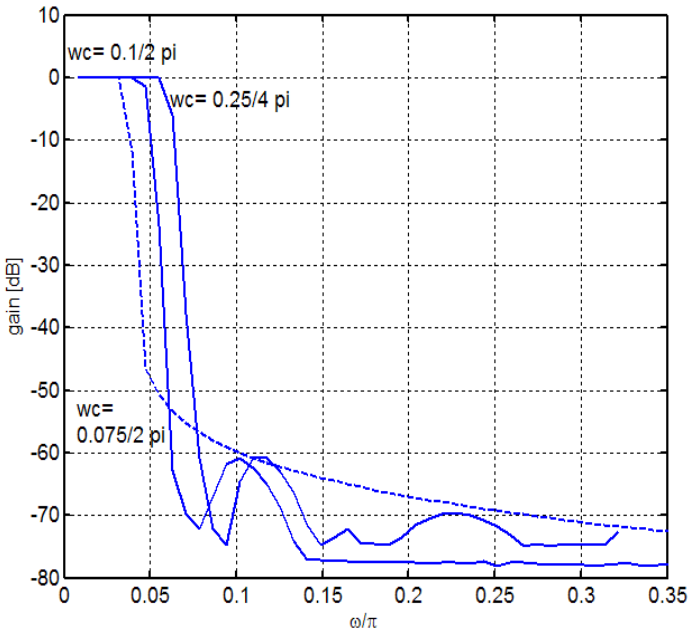


Слика 6.17 Карактеристике *Class III Kernel* филтра за $\omega_c = 0{,}25\pi$, $\omega_c = 0{,}1\pi$, $\omega_c = 0{,}075\pi$ и $\omega_c = 0{,}06\pi$

Као што се са Слике 6.17 види, грешка у прелазној зони овако реализованог филтра је знатно повољнија него у случају стандардне реализације са елиптичким филтром (Слика 6.11), тако да треба очекивати да када се укључи *Class III* комплементарни филтер у вишестепену структуру са Слике 6.10, биће добијена амплитудска карактеристика са мањом деградацијом прелазне зоне код ниских *cut-off* граничних фреквенција.

Из тог разлога ће бити настављено са сужавањем пропусног опсега свеукупног филтра, снижавањем граничне фреквенције *Class III* филтра, а затим ће такав филтер бити искоришћен као *Kernel* филтер у вишестепеној *multirate* реализацији.

Амплитудска карактеристика вишестепеног филтра, за неколико граничних фреквенција и неколико различитих степена децимације\интерполације је приказана на Слици 6.18.



Слика 6.18 Карактеристика свеукупног вишестепеног филтра за $\omega_{co} = 0{,}25\pi / 4$,

$\omega_{co} = 0{,}1\pi / 2$ и $\omega_{co} = 0{,}75\pi / 2$

На Слици 6.18 су приказане карактеристике:

- Вишестепеног филтра са $M = 2$ и граничном фреквенцијом *Kernel* филтра $\omega_c = 0{,}075\pi$ (гранична фреквенција укупног филтра $\omega_{co} = 0{,}0375\pi$),
- Вишестепеног филтра са $M = 2$ и граничном фреквенцијом *Kernel* филтра $\omega_c = 0{,}1\pi$ (гранична фреквенција укупног филтра $\omega_{co} = 0{,}05\pi$),
- Вишестепеног филтра са $M = 2$ и граничном фреквенцијом *Kernel* филтра $\omega_c = 0{,}25\pi$ (гранична фреквенција укупног филтра $\omega_{co} = 0{,}0625\pi$).

Приказане карактеристике говоре да за вишестепени филтер са *Kernel* филтром реализованим са комплементарним филтерским паровима класе *III* на *Xilinx-овом FPGA Virtex7* чипу и са 32-битном прецизношћу, минимална гранична учестаност пропусног опсега износи $\omega_{co} = 0{,}05\pi$. Овакав резултат је веома применљив, обзиром на уштеде хардверских ресурса које су резултат *multirate* реализације у комбинацији са *PI* техником (на Слици 6.10 је приказана реализација за два канала), и обзиром да је показано да би са стандардном реализацијом елиптичким филтром грешка услед реализације са фиксним зарезом била неприхватљиво велика.

6.3 FIR *SHARPENING* МЕТОДА ЗА РЕАЛИЗАЦИЈУ ФИЛТАРА СА ЈАКО УСКОМ ПРЕЛАЗНОМ ЗОНОМ

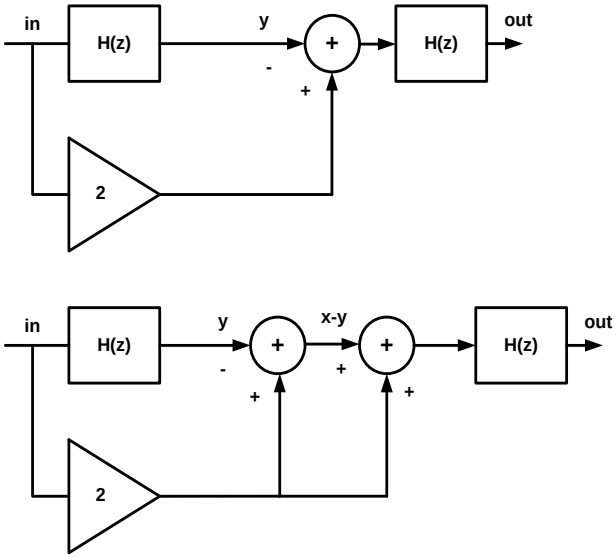
Један од најчешћих захтева које пројектанти дигиталних филтара добијају јесте захтев за очувањем линеарне фазне карактеристике. Такав захтев ставља у први план реализације дигиталних филтара са *FIR* структурама, обзиром да је линеарна фазна карактеристика њихова основна предност. Међутим, ограничавајући фактор код примене *FIR* филтара је претерано високи ред филтра (код ускопојасних реализација, или код филтера са јако уском прелазном зоном). Такви су на пример захтеви код пројектовања гониометара ускопојасних сигнала, где се поред јако уског пропусног опсега опсега захтева слабљење бочних опсега од минимално 80 dB. За тако строге захтеве, ред *FIR* филтра би био несразмерно велики ($\geq$ 100), тако да би кашњење које би такав филтар уносио било неприхватљиво. Такође, због великог броја аритметичко-логичких операција, грешка услед имплементације са фиксним зарезом би била јако изражена.

Једини начин за реализацију оваквих филтара представљала би примена неке од *multirate* метода реализације дигиталних филтара у комбинацији са неком од метода за корекцију амплитудске карактеристике.

Један од најједноставнијих поступака за сужавање прелазне зоне дигиталних филтара је сукцесивно филтрирање сигнала истим филтром. Са сваким филтерским степеном, вредност слабљења у непропусном опсегу би се увећава *к*-пута (*к* је број филтерских степена), док би се међутим и осцилације у пропусном опсегу такође увећале. Метода представљена у [34] се базирала на употреби *amplitude change* функција, и она се такође заснивала на вишеструкој употреби истог филтра, али одређеним редоследом и са скалираним факторима за множење. У суштини, функција преноса целе структуре би представљала полином функције преноса почетног филтра, при чему су коефицијенти

полинома одређени захтеваним слабљењем у непропусном опсегу и потребном ширином прелазне зоне.

Основни принцип реализације *amplitude change* функција је приказан је на Слици 6.19. Улазна секвенца $\{x_n\}$ се процесира кроз основни филтер $H(z)$ на чијем излазу се добија секвенца $\{y_n\}$. Сабирањем секвенце $\{y_n\}$ са скалираном негативном улазном секвенцом добија се секвенца разлике сигнала $x_n - y_n$. Сабирањем секвенце разлике сигнала $x_n - y_n$ са скалираном вреднћу улазног сигнала и филтрирањем идентичним филтром $H(z)$ добија се излазни сигнал са коригованом функцијом преноса (релација 6.10) а након следећег филтерског блока такође и секвенца разлике сигнала $x_n - y_n$.



Слика 6.19 Принцип корекције амплитудске карактеристике дигиталних филтара заснован на примени *amplitude change* функција

$$\begin{aligned} &H(z)[1+1-H(z)] = H(z)[2-H(z)] \\ &= H(z)[1+H_r(z)], \end{aligned} \tag{6.10}$$

где је

$$H_r(z) = 1 - H(z) \tag{6.11}$$

секвенца разлике.

Представљена операција се популарно назива "*twicing*" - дуплирање. На слици 6.20 је приказана *amplitude change* функција другог реда (пуна линија) и неколико функција вишег реда (испракидане линије). Као што видимо, са увећањем реда функције, могу се значајно побољшати карактеристике филтра у прелазној зони. Међутим оно што је веома битно код представљене методе је то што ако почетни филтер $H(z)$ има линеарну фазну

карактеристику (константно групно кашњење), и укупни филтер ће имати линеарну фазну карактеристику, јер његова функција преноса представља полином по $H(z)$.

То значи да је избор филтра $H(z)$ ограничен једино захтевом за линеарном фазном карактеристиком. Уколико је потребно да филтер $H(z)$ буде реализован неком од *multirate* техника како би се повећала ефикасност дигиталног филтрирања и била смањена количина потребних хардверских ресурса, техника фреквенцијског маскирања се намеће као најоптималније решење. Наиме, овом техником се задовољава услов линеарности, а процесирање сигнала се уместо на учестаности одабирања врши на фреквенцији која је нижа неколико пута. Лоша страна њене примене би била увећање времена потребног за процесирање сигнала, обзиром да се у случају затворених повратних спрега ово време акумулира дуж сваког степена.

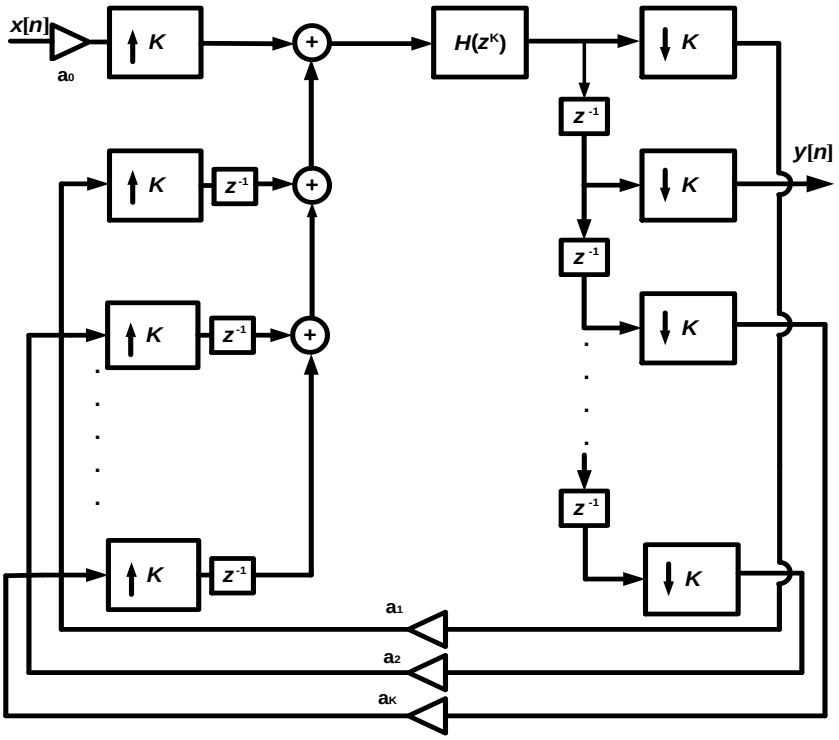
Чињеница да се код поступка корекције амплитудске карактеристике функције преноса употребом *amplutude change* функције исти филтер користи више пута, може бити искоришћена за ефикасну реализацију сукцесивног филтрирања употребом *PI* технике.

У поглављу 3.3 је показан модификовани поступак *PI* технике за реализацију полинома функције $H(z)$, тј.функције:

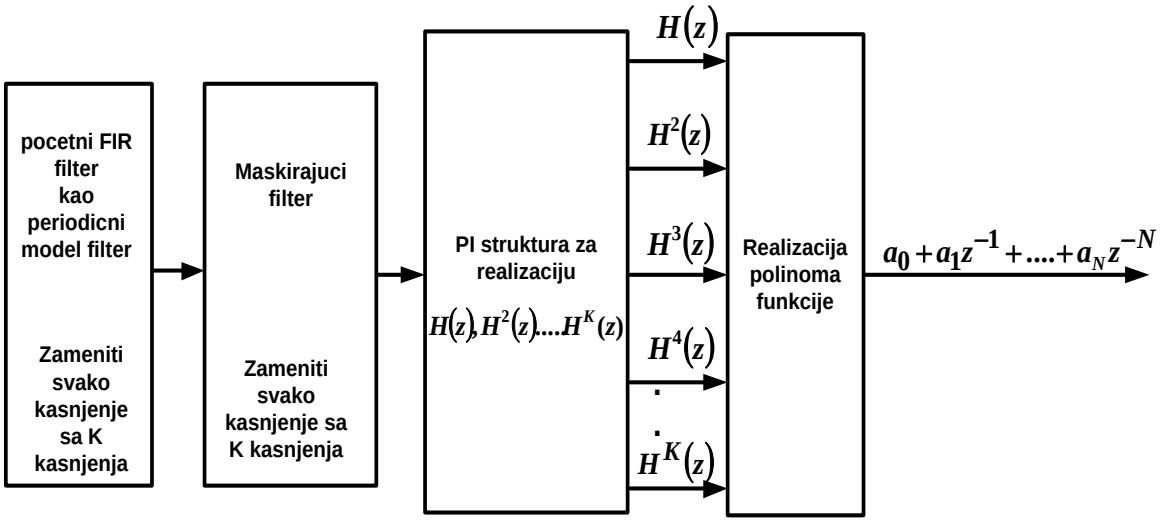
$$H_K(z) = a_K \cdot H^K(z) + a_{K-1} \cdot H^{K-1}(z) + ... + H(z), \tag{6.11}$$

увођењем K повратних спрега за реализацију полинома *K*-тог реда. Велика предност примене модификованог *PI* поступка на *FIR* филтре је што је укупни број повратних спрега једнак реду полинома функције $H(z)$, док је код рекурзивне *IIR* структуре због великог броја повратних спрега, прорачун критичне повратне спреге веома сложен, а за филтре вишег реда модификовани *PI* поступак није могуће применити.

На Слици 6.21 је приказан начин реализације полинома функције $H_K(z)$ *K*-тог реда. За реализацију овог полинома, користи се један филтер $H(z)$ коме је свако кашњење замењено са K кашњења. Избором коефицијената a_K, a_{K-1}, a_{K-2}.....1 одређује се полином којим се реализује *amplitude change* функција и врши корекција амплитудске карактеристике. Како приликом реализације на *FPGA* платформи ови коефицијенти могу бити програмабилни, то се избор реда *amplitude change* функције може вршити директно, на основу захтева за слабљењем у прелазној зони и непропусном опсегу. Укупни алгоритам примене "*sharpening*" методе у комбинацији са техником фреквенцијског маскирања и модификованим *PI* поступком је приказан на Слици 6.22



Слика 6.21 Реализација модификованог *PI* поступка за реализацију полинома *K*-тог реда



Слика 6.22 Реализација модификованог *PI* поступка за реализацију полинома *K*-тог реда

Као и у предходним примерима, имплементација започиње од филтра са спецификацијама једноставним за реализацију:

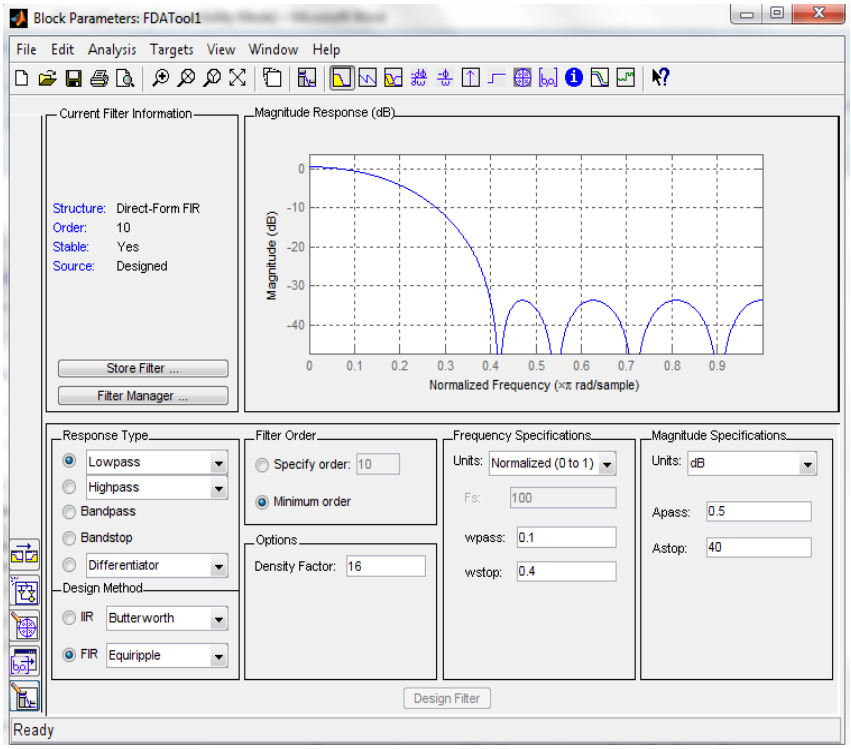
$\omega_{pass} = 0{,}1 \cdot \pi \quad rad/sample$

$\omega_{stop} = 0{,}4 \cdot \pi \quad rad/sample$

$A_{pass} = 0{,}5 \quad dB$

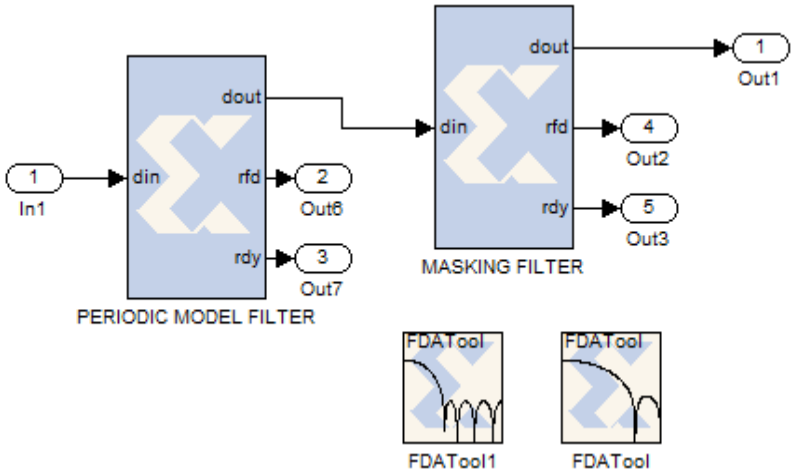
$A_{stop} = 40 \quad dB$

Овакви захтеви могу бити задовољени са *FIR equurupple* филтром ниског реда (r=10) и филтер може бити реализован коришћењем алата *FDA tool*, као што је приказано на Слици 6.23.



Слика 6.23 Реализација почетног *FIR* модел филтра

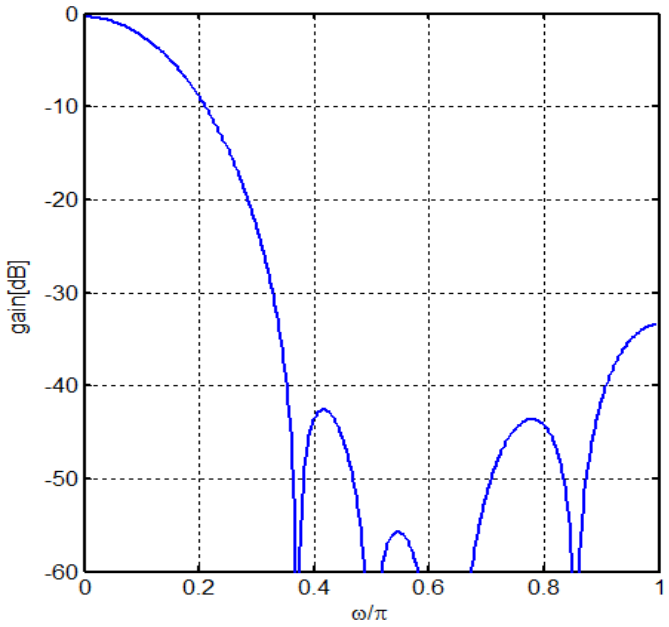
Овакав почетни филтер ће као што је речено бити употребљен као модел филтер приликом примене технике фреквенцијског маскирања, као што је то приказано на Слици 6.24. За елиминисање периодичних одраза је потребан маскирајући филтер, а у посматраном случају за $M = 2$, спецификације маскирајућег филтра ће бити:



Слика 6.24 Реализација укупног филтра техником фреквенцијског маскирања

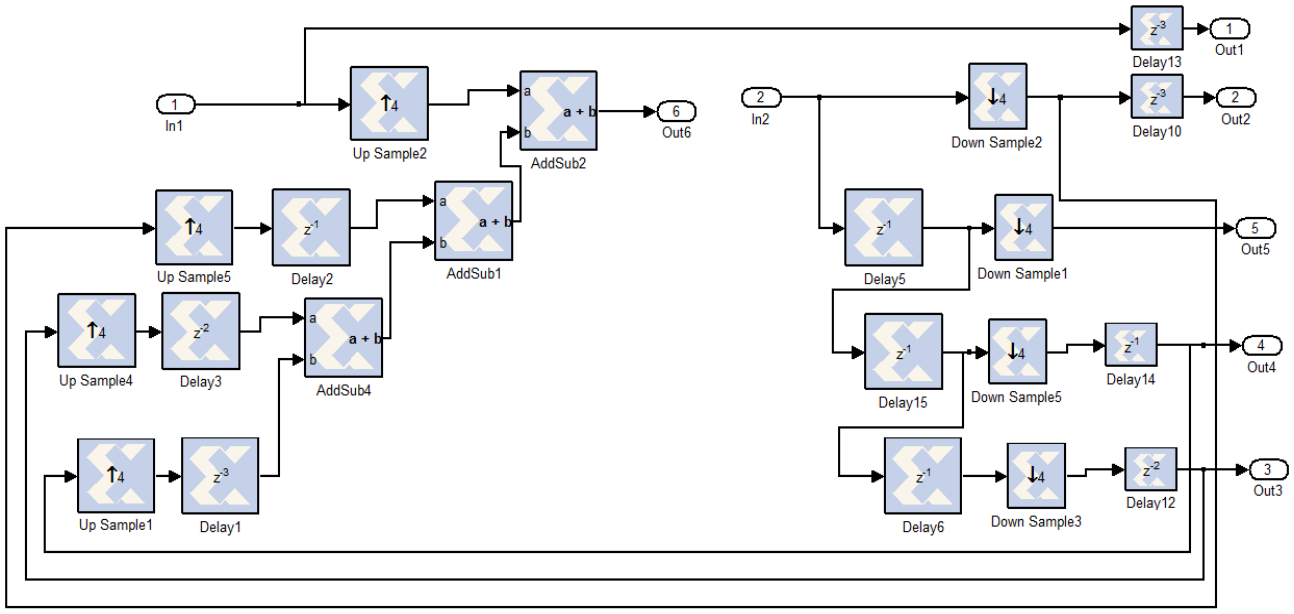
$$\omega_{pass} = 0{,}1 \cdot \pi \quad rad/sample$$
$$\omega_{stop} = 0{,}7 \cdot \pi \quad rad/sample$$
$$A_{pass} = 0{,}5 \quad dB$$
$$A_{stop} = 40 \quad dB$$

а што задовољава *FIR equurupple* филтер трећег реда. Укупна реализација технике фреквенцијског маскирања је приказана на Слици 6.24. Оба филтра су реализована користећи софтверске алате *FIR compiler* и *FDA tool*. Периодични модел филтер се реализује заменом сваког кашњења у модел филтру са два кашњења (за $M = 2$), и то постављајући *FIR compiler* опцију "*Filter type"* на вредност "*interpolated"* и "*zero packing factor"* на вредност "*2*". У овом кораку избором параметара модел филтра одређују се границе пропусног и непропусног опсега укупног филтра, док се даљом применом *sharpening* методе коригује карактеристика филтра у прелазној зони и вредност слабљења у непропусном опсегу. Амплитудска карактеристика филтра приказана је на Слици 6.25. Приказани филтер ће имати линеарну фазну карактеристику јер примена технике фреквенцијског маскирања не деградира фазну карактеристику.



Слика 6.25 Амплитудска карактеристика филтра реализованог техником фреквенцијког маскирања

Реализацију настављамо укључивањем оваквог филтра у структуру за реализацију полинома функције преноса $H(z)$ четвртог реда модификованим *PI* поступком као што је то приказано на Слици 6.26.



Слика 6.26 Реализација полинома четвртог реда

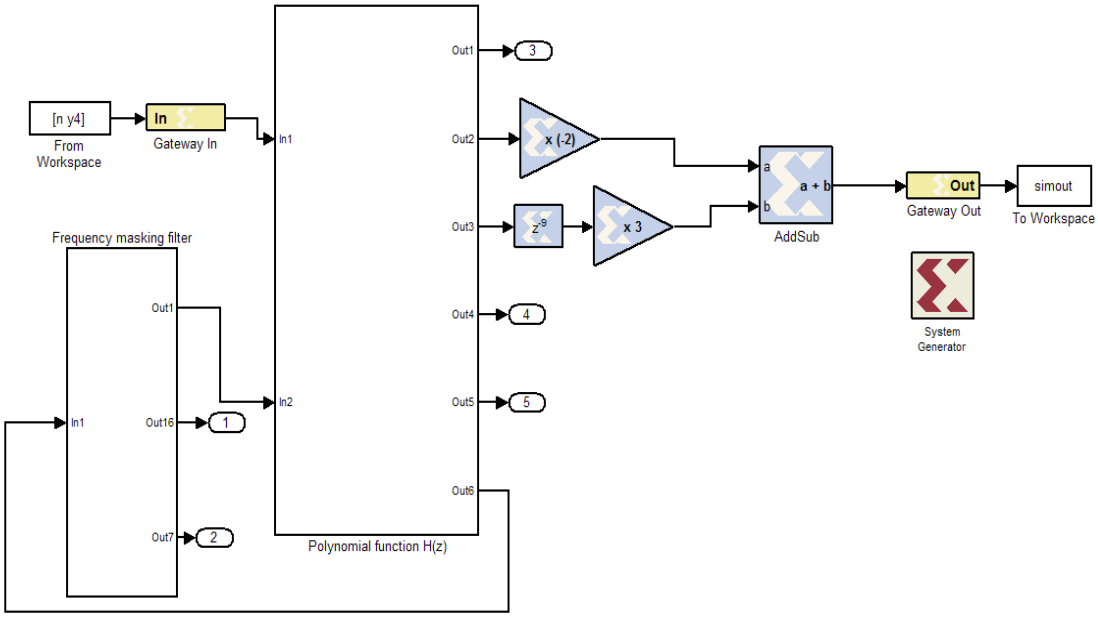
Код модификованог *PI* поступка, за реализацију полинома K-тог реда, свако кашњење у филтру требало би бити замењено са K кашњења, а узимајући у обзир и формирање периодичног модел филтра потребно је урадити следеће:

- Периодични модел филтер – Заменити свако кашњење са $M \cdot K$ кашњења, где је M фактор фреквенцијског маскирања, а K је ред полинома по $H(z)$. Овај поступак се извршава постављањем "*FIR Compiler*" опције "*Filter type* " на вредност "*interpolated*", а опције "*zero packing factor*" на вредност $M \cdot K$.
- Маскирајући филтер - Заменити свако кашњење са K кашњења. Овај поступак се извршава постављањем "*FIR Compiler*" опције "*Filter type*" на вредност "*interpolated*", а опције "*zero packing factor*" на вредност K.

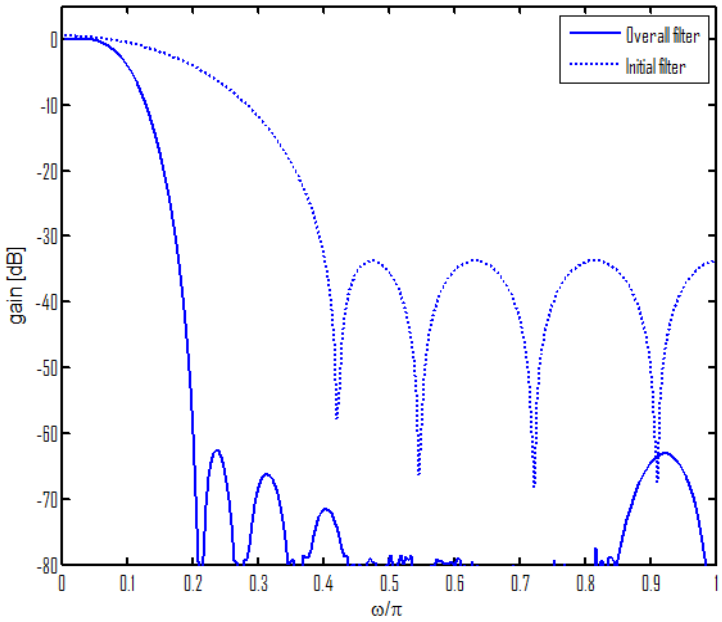
Овако дефинисани филтер сада треба укључити у структуру за реализацију полинома функције $H(z)$ као на Слици 6.26. На излазним портовима *Out1, Out2…Out5* структуре, ће се налазити излази чије су функције преноса $1, H(z), H^2(z), H^3(z)$ и $H^4(z)$ у односу на улаз у структуру. Требало би напоменути, да би због фазног усклађивања требало додати додатна кашњења у гране са нижим вредностима степена полинома.

Након дефинисања почетног *FIR* филтра (реализованог техником фреквенцијског маскирања), сада је потребно реализовати *amplitude change* функцију у циљу корекције амплитудске карактеристике и сужавања прелазне зоне филтра. Из тог разлога сигнали са

излазних портова структуре са Слике 6.26 (додајући потребна кашњења) се употребљавају за реализацију одговарајућег полинома [35, 36]. На Слици 6.27 је приказана *FPGA PI* реализација *amplitude change* функције $F(H) = 3H^2 - 2H^3$. За изабрани филтер дефинисан маскирајућим и модел филтром, и *amplitude change* функцијом трећег реда, добијамо амплитудску карактеристику која је приказана на Слици 6.28.



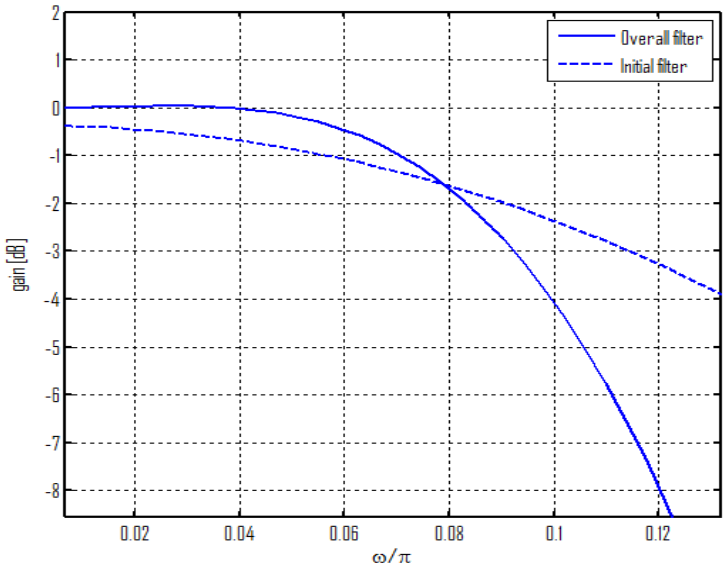
Слика 6.27 *FPGA* реализација укупног филтра



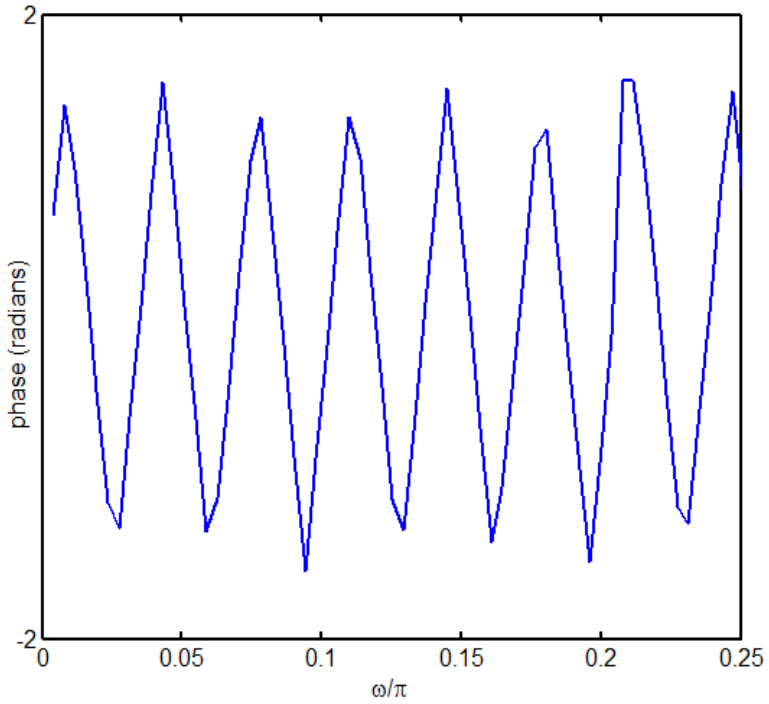
Слика 6.28 Амплитудска карактеристика укупног филтра

Као што видимо, побољшања амплитудске карактеристике су значајна, како у смислу слабљења у непропусном опсегу, тако и у смислу сужавања прелазне зоне. Осим тога,

амплитудска карактеристика (Слика 6.29) није деградирана у пропусном опсегу, (ефекат вишеструког филтрирања идентичним филтром не долази до изражаја), уз истовремено очување линеарне фазне карактеристике, што је приказано на Слици 6.30.



Слика 6.29 Амплитудска карактеристика укупног филтра – пропусни опсег



Слика 6.30 Фазна карактеристика укупног филтра

Даље сужавање карактеристика филтра, као и сужавање прелазне зоне филтра се може извести на два начина, а то су:

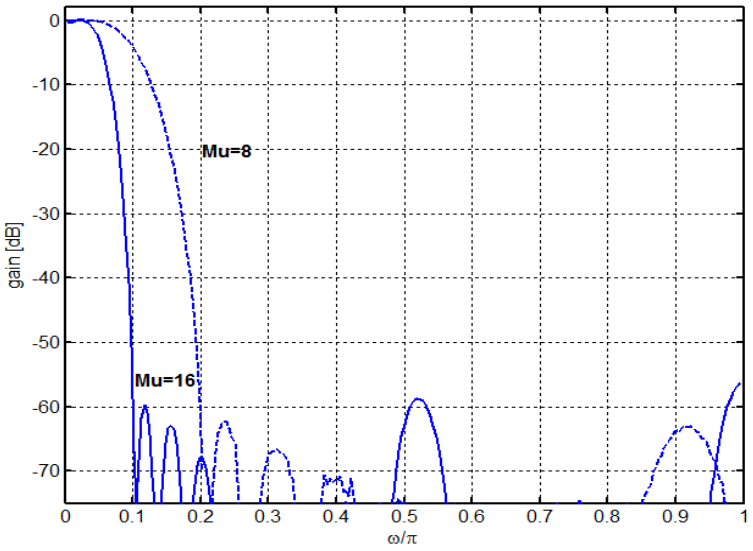
- Употребом *amplitude change* функције вишег реда,

- Коришћење технике фреквенцијског маскирања са већим степеном промене учестаности одабирања.

Користећи *amplitude change* функције вишег реда може се смањити ширина прелазне зоне као и увећати слабљење у непропусном опсегу. Лоша страна оваквог приступа је усложњавање структуре филтра, јер се са већим редом полинома по $H(z)$ увећава број основних секција филтра у укупној петљи. Грешка услед имплементације са фиксним зарезом се увећава са сваким наредним степеном а увећава се и време пропагације сигнала због увећања броја елемената за кашњење.

Употребом технике фреквенцијског маскирања са већим степеном промене учестаности одабирања врши се сужавање пропусног опсега филтра сразмерно степену промене [37, 38]. Међутим, и овакав приступ је ограничен величином грешке у пропусном опсегу и прелазној зони која се увећава сразмерно увећању степена промене учестаности одабирања. Имајући то на уму, реализација ускопојасних филтара код којих је ширина пропусног опсега мања од на пример 1/20 учестаности одабирања би требало да се изводи комбинацијом ове две методе, с'тим да се ширина пропусног одзива филтра дефинише променом степена конверзије учестаности одабирања, а ширина прелазне зоне редом *amplitude change* функције.

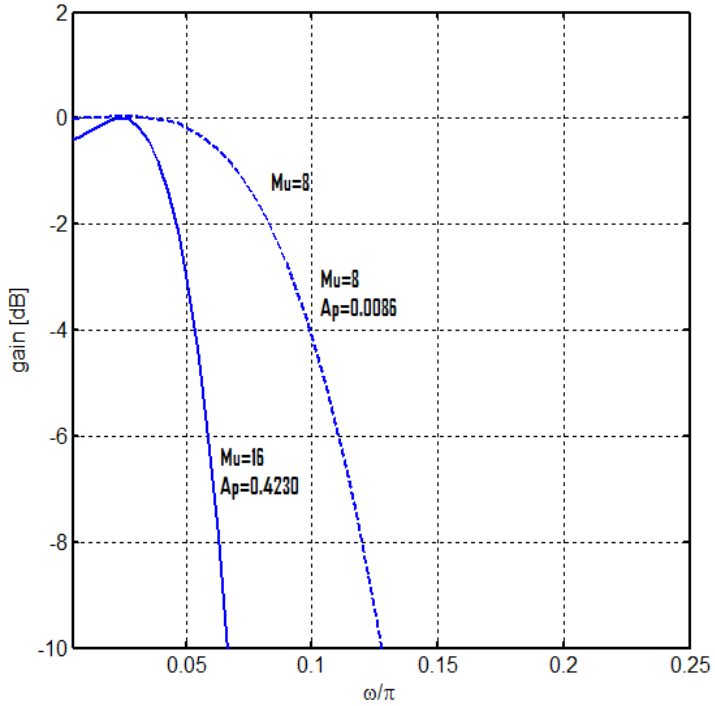
На Слици 6.31 је приказана амплитудска карактеристика ускопојасног филтра који је настао увећањем фактора конверзије филтра из предходног примера, и то за вредност $M=8$ и $M=16$.



Слика 6.31 Карактеристика ускопојасног филтра
са увећаним фактором конверзије

Као што видимо са Слике 6.32 на којој је приказан пропусни опсег филтра, за вредност $M=16$, грешка у пропусном опсегу излази из дозвољених оквира, и такав ускопојасни филтер није прихватљив.

Треба нагласити да је целокупна метода примењена на почетни *FIR* филтер који је укључен у методу фреквенцијског маскирања, имајући на уму захтеве за линеарношћу фазне карактеристике. Као што је познато, са *IIR* филтерским структурама је постигнута много боља амплитудска карактеристика и уз мање трошење хардверских ресурса, али услов линеарности фазне карактеристике не би могао да буде задовољен.



Слика 6.32 Пропусни опсег ускопојасног филтра
са увећаним фактором конверзије

6.4 *CIC* ФИЛТРИ ЗА ПРОМЕНУ УЧЕСТАНОСТИ ОДАБИРАЊА СА ВЕЛИКИМ ФАКТОРОМ КОНВЕРЗИЈЕ

CIC (*engl. Curcuut Integrated Comb*) филтри се често користе у вишебрзинским дигиталним системима код пројектовања кола за промену учестаности одабирања. Њихова особина да могу вршити филтрирање без употребе множача је веома битна код процесирања сигнала великих учестаности одабирања. Осим ове особине, веома важна карактеристика *CIC* филтара је узак пропусни опсег, тако да су веома атрактивни код промене учестаности одабирања са високим фактором промене. У случајевима вишеструке децимације, *CIC* филтер се најчешће користи као први степен, док се у случајевима вишеструке

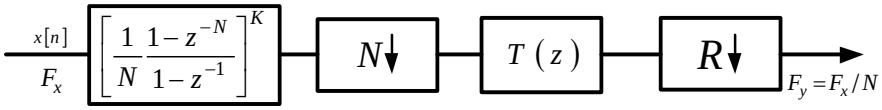
интерполације *CIC* филтер најчешће користи као завршни степен. У оба случаја, улога *CIC* филтара је обезбеђивање конверзије учестаности одабирања високог реда, док наредни степени обезбеђују довољно слабљење у непропусном опсегу и смањују осцилације у пропусном опсегу.

6.4.1 НОВЕ ТЕХНИКЕ РЕАЛИЗАЦИЈЕ *CIC* ФИЛТРАРА ЗА ПРОМЕНУ УЧЕСТАНОСТИ ОДАБИРАЊА СА ВЕЛИКИМ ФАКТОРОМ КОНВЕРЗИЈЕ

Једна од првих примена *CIC* филтара код вишеструке конверзије учестаности одабирања је каскадно повезивање *CIC* и *FIR* дециматора (интерполатора). У таквој конфигурацији улога *CIC* дециматора (интерполатора) је конверзија учестаности одабирања са великим фактором конверзије, а улога *FIR* дециматора (интерполатора) је компензација амплитудске карактеристике у непропусном опсегу и обезбеђивање захтеване ширине прелазне зоне. Укупни фактор конверзије M се може изразити као производ фактора конверзије *CIC* и *FIR* степена као:

$$M = N \times R, \tag{6.12}$$

где су N и R фактори конверзија *CIC* и *FIR* степена респективно. Блок шема реализације овакве структуре у случају имплементације дециматора је приказана на Слици 6.33. Аналогно овоме, у случају увећања учестаности одабирања, *CIC* интерполатор би се налазио у завршном степену.



Слика 6.33 Децимација у два степена, *CIC* дециматор и *FIR* дециматор

Промена учестаности одабирања која је изведена као редна веза *CIC* и *FIR* дециматора (интерполатора) је веома ефикасна у смислу уштеде хардверских ресурса и захвална за реализацију у техници са фиксним зарезом, обзиром на то да не садржи елементе за множење. Међутим, велики недостаци овакве реализације су:

1. Секција интегратора ради на максималној учестаности одабирања, тако да је сагласно томе утрошак снаге велики,

2. За велике вредности фактора конверзије N може доћи до препуњења регистара,

3. Фреквенцијски одзив је одређен са само два параметра, N и R, тако да структура не може да испуни захтеве код строжијих захтева.

Јовановић-Долечек и Митра [39] су предложили реализацију по којој би фактор конверзије *N*, *CIC* дециматора био подељен на два степена, тј. $N = N_1 \times N_2$. У овом случају, трансфер функција децимационог филтра *H*(*z*) би представљала производ две филтерске секције. Улога прве секције би била сужавање прелазне зоне, а друге побољшавање слабљења у непропусном опсегу.

Једно од решења за снижење утрошка снаге приликом децимације са великим фактором конверзије би била примена неке од нерекурзивних техника. Наиме, показано је да се применом добро познате полифазне декомпозиције на реализацију функције преноса *CIC* филтра, утрошак снаге укупног филтра знатно смањује у односу на рекурзивне реализације.

Као што је познато, нерекурзивна представа функције преноса *CIC* филтра $G_c^K(z)$ (*K* је број *CIC* степена) је дата са:

$$G_c^K(z) = \left[\frac{1}{N} \sum_{n=0}^{N-1} z^{-n} \right]^K . \tag{6.13}$$

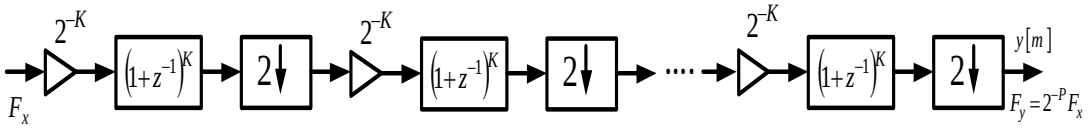
Уколико се коефицијент *N* може изразити као умножак броја 2, тј. ако важи $N = 2 \cdot P$ где је *P* целобројна вредност, тада се једначина 6.13 може изразити у следећој форми:

$$G_c^K(z) = 2^{-PK} \prod_{i=0}^{P-1} \left(1 + z^{2^i}\right)^K . \tag{6.14}$$

Тако на пример, ако је $N = 16$, $G_c^K(z)$ се може изразити на следећи начин:

$$G_c^K(z) = 2^{-16K} \left(1 + z^{-1}\right)^K \left(1 + z^{-2}\right)^K \left(1 + z^{-4}\right)^K \left(1 + z^{-8}\right)^K . \tag{6.15}$$

Из једначине 6.15 се може закључити да се дециматор изведен каскадном везом *CIC* филтра и кола за снижење учестаности одабирања са фактором конверзије који је умножак броја 2, може изразити као каскадна веза *P* дециматора. Сваки дециматор би се састојао из нерекурзивног субфилтера $\left(1 + z^{-1}\right)^K$ и кола за снижење учестаности одабирања са фактором 2. Блок шема тако реализованог дециматора је приказана на Слици 6. 34.

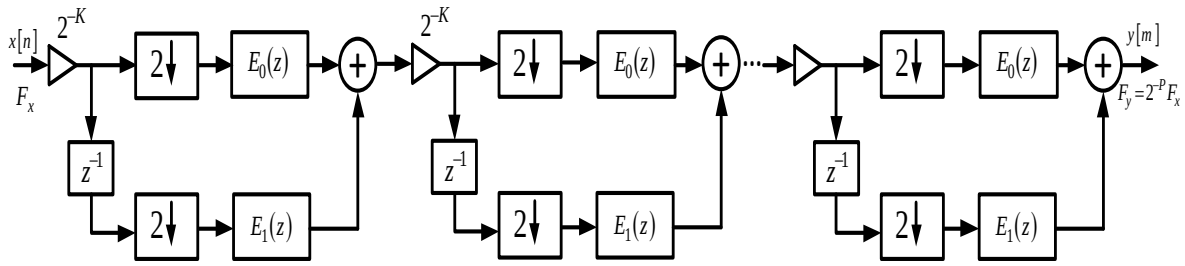


Слика 6.34 Каскадна нерекурзивна имплементација *CIC* дециматора

Са Слике 6.34 видимо да само први степен дециматора ради на максималној учестаности одабирања, а да је затим, са сваким наредним степеном, учестаност одабирања дупло мања. Даље снижавање радне учестаности одабирања за фактор 2 се може извршити , уколико се *FIR* субфилтер $\left(1+z^{-1}\right)^K$ изрази у функцији његових полифазних компонената:

$$\left(1+z^{-1}\right)^K = E_0\left(z^2\right) + z^{-1}E_1\left(z^2\right). \tag{6.16}$$

Ако се кола за промену учестаности одабирања транслирају на место пре филтара, добија се структура која је позната као полифазна имплементациона форма (Слика 6.35). Предност овакве реализације је, поред сниженог утрошка снаге и то што регистри нису у опасности од препуњавања, јер је дужина кодне речи, за сваки степен (*i)*, и за улазну реч дужине W_0 бита, ограничена на $\left(W_0 + K \cdot i\right)$ бита [40].



Слика 6.35 Полифазна нерекурзивна имплементација *CIC* дециматора

Најважнији недостатак полифазне реализације је увођење кола за множење, тј, једног множача по секцији. Имајући на уму да код рекурзивне реализације овај проблем не постоји, пресудно ограничавајући фактор код полифазне имплементације је ред фактора конверзије дециматора (интерполатора), тако да метода није применљива код великих вредности фактора конверзије.

Једна од веома атрактивних модификација *CIC* дециматора (интерполатора), а која је погодна код ΣΔ A/D конверзије је тзв. концепт ротације природних нула. Нови децимациони филтер је пројектован тако да побољша слабљење шума квантизације у "aliasing'' областима првог децимационог степена. Резултат примене концепта ротације

природних нула је генерисање нових нула, а које су постављене тако да је сваки пар нових нула смештен симетрично у односу на природне нуле *CIC* филтра.
Функција преноса *CIC* филтра са једним степеном износи:

$$G_c(z)=\frac{1}{N}\frac{1-z^{-N}}{1-z^{-1}}. \tag{6.17}$$

Нуле функције преноса *CIC* филтра у Z равни, $z_i, i=1,2..(N-1)$ су униформно распоређене у односу на јединични круги налазе се у тачкама:

$$e^{j\pi/N}, e^{2j\pi/N}....e^{(N-1)j\pi/N}. \tag{6.18}$$

Уколико се нуле функције преноса ротирају у смеру кретања казаљки на часовнику за угао α, нова функција таквог *CIC* филтра ће бити [41]:

$$G_{q+}(z)=\frac{1}{N}\frac{1-z^{-N}e^{j\alpha N}}{1-z^{-1}e^{j\alpha N}}. \tag{6.19}$$

Аналогно овоме, код ротације природних нула у супротном смеру за угао α, нова функција преноса ће износити:

$$G_{q-}(z)=\frac{1}{N}\frac{1-z^{-N}e^{-j\alpha N}}{1-z^{-1}e^{-j\alpha N}}. \tag{6.20}$$

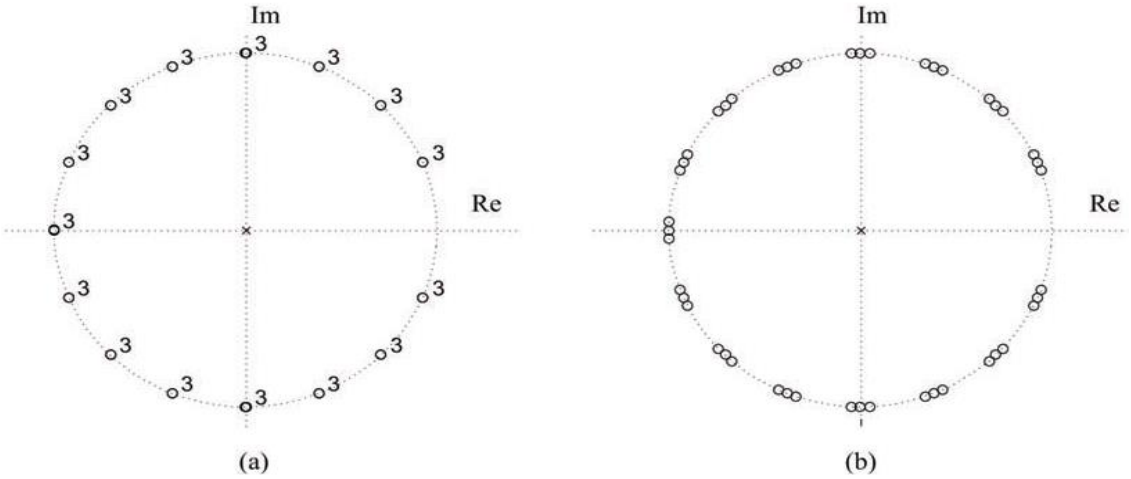
Множењем функција преноса основне *CIC* филтерске секције $G_c(z)$ са функцијама преноса секција ротираних природних нула $G_{q+}(z)$ и $G_{q-}(z)$, добија се функција преноса укупног филтра чија је функција преноса:

$$G_e(z)=\frac{1}{N^3}\frac{1-z^{-N}}{1-z^{-1}}\frac{1-2\cos(\alpha N)z^{-N}+z^{-2N}}{1-2\cos(\alpha)z^{-1}+z^{-2}}, \tag{6.21}$$

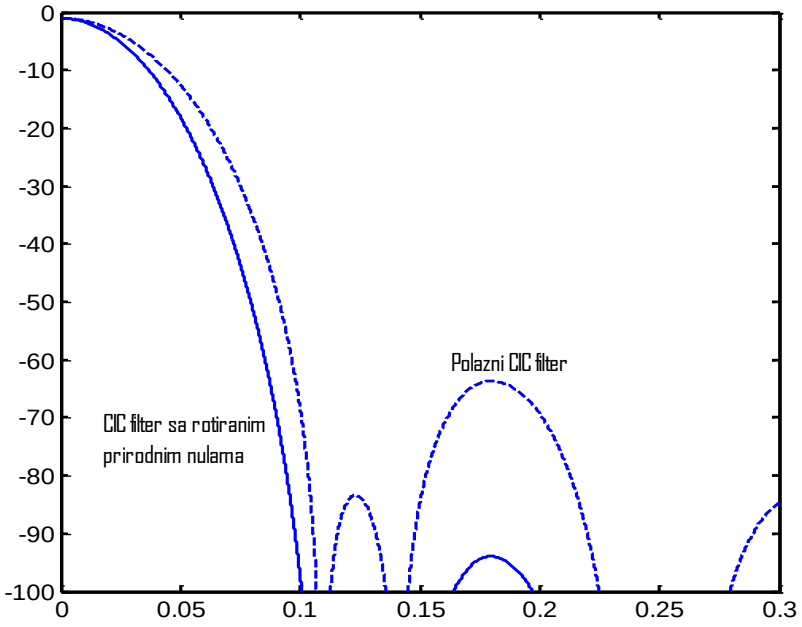
а амплитудска карактеристика:

$$G_e(f)=\frac{1}{N^3}\frac{\sin(\pi n\, f/2)}{\sin(\pi\, f/2)}\frac{\sin(\pi f+\alpha)N/2}{\sin(\pi f+\alpha)/2}\frac{\sin(\pi f-\alpha)N/2}{\sin(\pi f-\alpha)N/2}e^{-j3\pi f(N-1)/2}, \tag{6.22}$$

Као што се види из релације 6.22, амплитудска функција укупног филтра представља призвод три *sinc* функције. Како је основна замисао методе да амплитудска функција новог филтра има нуле у *aliasing* областима полазног филтра, вредност α се бира да такав услов буде задовољен. У радовима [42, 43] је показано, да ће тражени услов бити задовољен ако је $\alpha = q\pi f_m$, где је f_m највиша фреквенција у спектру улазног сигнала. Тада, положај нула новог филтра у *Z* равни треба да изгледа као на Слици 6.36, а амплитудска карактеристика тако модификованог филтра као на Слици 6.37.



Слика 6.36 Положај нула у *Z* равни а: *CIC* филтра са три степена, б: Модификованог *CIC* филтра са три степена



Слика 6.37 Амплитудска карактеристика модификованог филтра

6.4.2 *CIC* ФИЛТРИ ЗА ПРОМЕНУ УЧЕСТАНОСТИ ОДАБИРАЊА СА ВЕЛИКИМ ФАКТОРОМ КОНВЕРЗИЈЕ РЕАЛИЗОВАНИ ТЕХНИКОМ ФРЕКВЕНЦИЈСКОГ МАСКИРАЊА И *SHARPENING* МЕТОДОМ

Методе описане у предходном поглављу су имале за циљ да унапреде процесе децимације односно интерполације у случајевима високог фактора конверзије, када се захтева од децимационих (интерполационих) филтара јако узак пропусни опсег и стрма амплитуска карактеристика у прелазној зони. При томе се захтева да филтри задрже линеарну фазну карактеристику (како не би дошло до фазних изобличења) као и да реализација буде или без множача, или са ограниченим бројем елемената за множење.

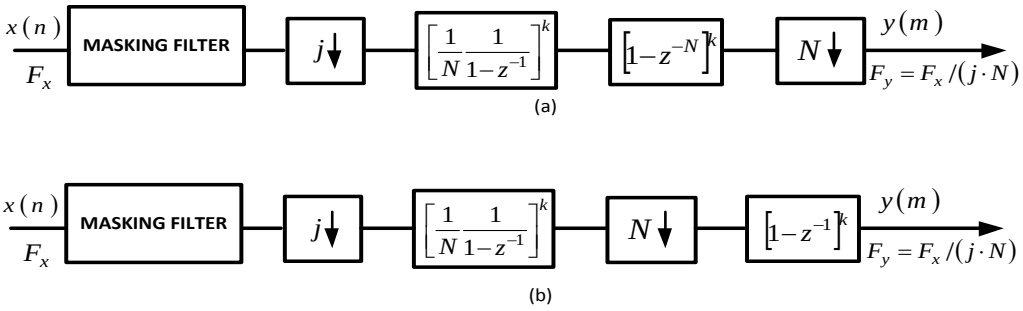
Овде ће бити представљена нова метода реализације децимационих (интерполационих) филтара која комбинује технику фреквенцијског маскирања са тзв. "*filter sharpening*" методом како би се добио дециматор (интерполатор) јако високог реда са стрмом прелазном зоном, линеарне фазне карактеристике и са малим бројем елемената за кашњење.

Директна примена технике фреквенцијског маскирања на један *CIC* дециматор је приказана као на Слици 6.38. Поступак технике фреквенцијског маскирања предвиђа замену сваког кашњења у децимационом филтру са l кашњења (извођење периодичног модел филтра) као и увођење додатног филтра као маскирајућег филтра, како би се уклонили периодични одрази. Након тога, укупни степен конверзије се мења и износи:

$$N_{mask} = N \cdot l\,, \tag{6.22}$$

Уколико је фактор фреквенцијског маскирања l доста мањи од првобитног фактора конверзије N, маскирајући филтер се може транслирати пре периодичног модел филтра. Ако је услов $l<<N$ испуњен, онда ће ред маскирајућег филтра бити минималан, што је веома битно за смањење утрошка снаге обзиром да ће маскирајући филтер радити на максималној учестаности одабирања (Слика 6.39).

Слика 6.38 *CIC* дециматор реализован техником фреквенцијког маскирања



Слика 6.39 Трансформација *CIC* дециматора реализованог техником фреквенцијког маскирања а. транслација периодичног модел филтра, б. транслација блока интегратора

Генерално, спецификације маскирајућег филтра, које обезбеђују уклањање периодичних одраза уз услов да ред филтра буде минималан су [44]:

$$\omega_p = \pi/(l \cdot N),$$
$$\omega_s = (2 \cdot \pi/l) - (\pi/(l \cdot N)) = \frac{\pi}{l}\left(2 - \frac{1}{N}\right), \tag{6.23}$$

Представљени поступак, даје добра резултате у случајевима транслације спектра корисног сигнала код дигиталне "down"-конверзије, компензујући већину недостатака као што су смањење утрошка снаге, увећање броја парааметара којима се одређује амплитудска карактеристика, препуњење регистара итд. Међутим, недостатак овакве примене је мала стрмина прелазне зоне у односу на укупан пропусни одзив филтра, тако да у применама где је таква особина од пресудне важности, корисна је примена неке од метода сужења прелазне зоне. Једна од таквих метода је тзв. *Filter sharpening* метода. У комбинацији са техником фреквенцијског маскирања могу се реализовати ефикасни дециматори (интерполатори) са веома стрмом карактеристиком децимационих (интерполационих) филтара.

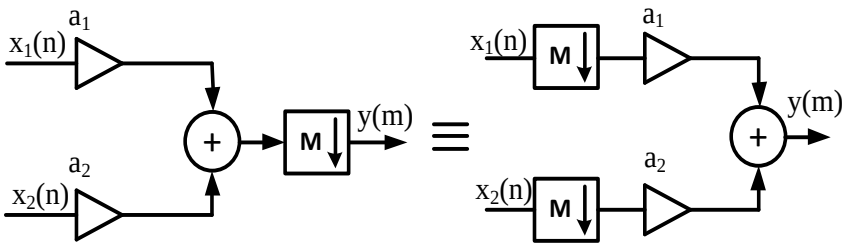
Основни облик "amplitude change" функције, која је у општем случају полимом *n*-тог реда износи:

$$H_{\mathrm{mod}}(z) = \sum_{i=1}^{M} A_i \cdot H^i(z), \tag{6.24}$$

CIC дециматор ће бити реализован као паралелна структура полинома *i* тог реда према релацији 6.24 , при чему ће на сваку паралелну грану бити примењена техника

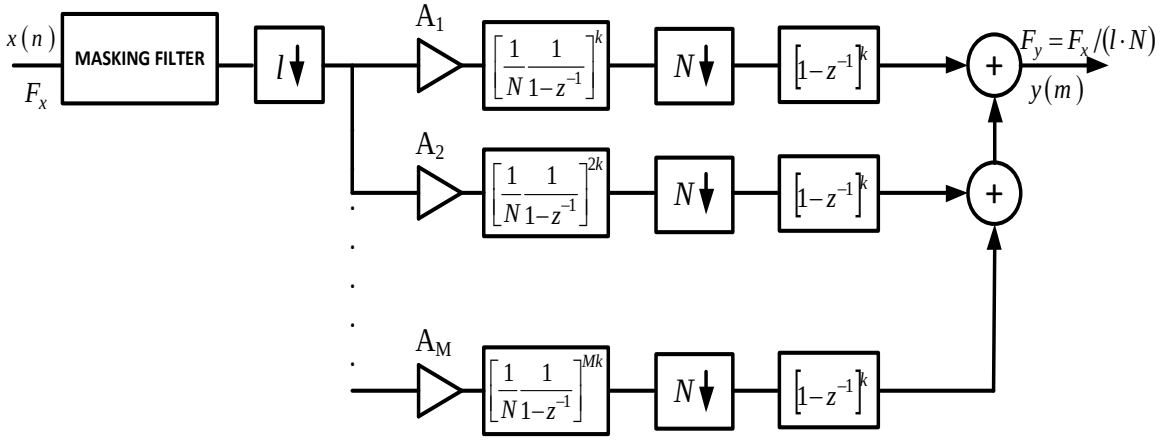
фреквенцијског маскирања са трансформацијама транслирања маскирајућег филтра и блока интегратора (Слика 6.39).

Затим, модификација може да се настави транслацијом множача A_1, A_2..$A_и$, иза кола за снижење учестаности одабирања за фактор l према Слици 6.40, тако да ће они сада радити на нижој учестаности одабирања.



Слика 6.40 Транслација множача на нижу учестаност одабирања

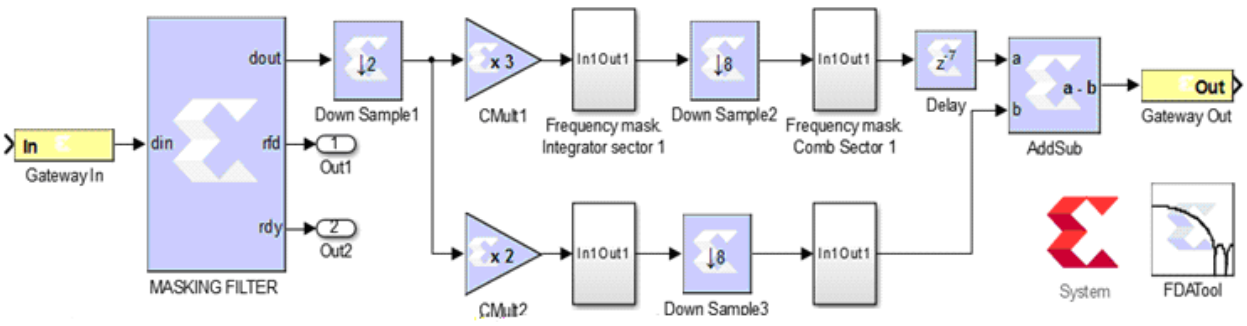
На крају, укупна имплементациона структура *CIC* дециматора, реализованог техником фреквенцијског маскирања и "*filter sharpening*" методом са увећаним фактором l *x* N приказана је на Слици 6.41.



Слика 6.41 Имплементациона структура *CIC* дециматора са фактором, реализованог техником фреквенцијског маскирања и "*filter sharpening*" методом

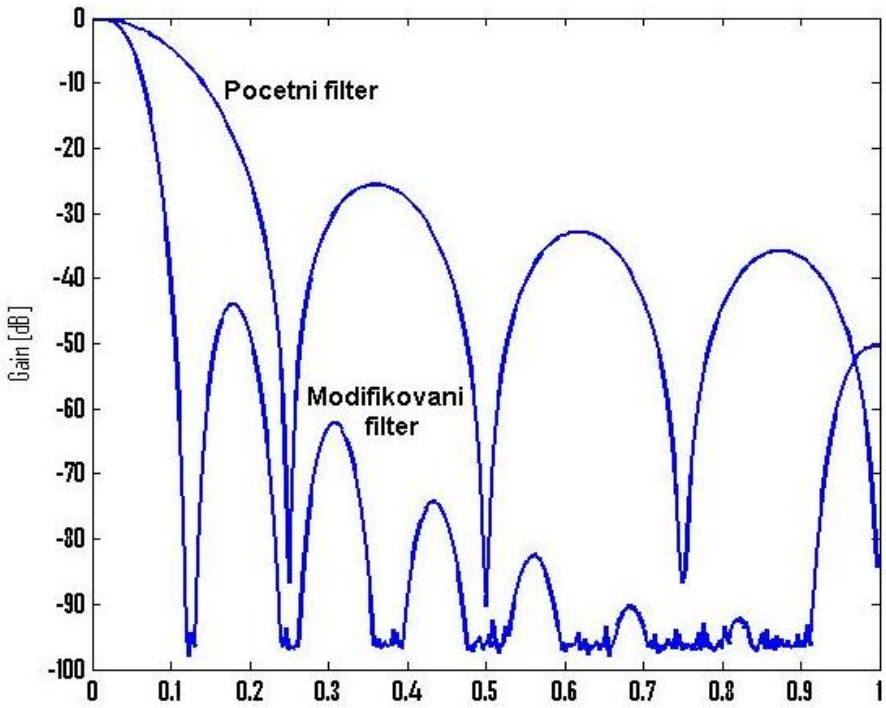
И у овом случају, на максималној учестаности одабирања ради само маскирајући филтер, због чега је и битно да буде испуњен услов $l<<N$, како би утрошак снаге био минималан.

Погледајмо сада пример реализације оваквог дециматора на *FPGA* чипу типа *Virtex* 5, ако су параметари N=8, j=2 ($N_{mod}=16$), и код примене *amplitude change* функције трећег реда $F(H)=H^2(3-2H)$. Имплементациона шема реализована софтверским алатом *System generator* је приказана на Слици 6.42.



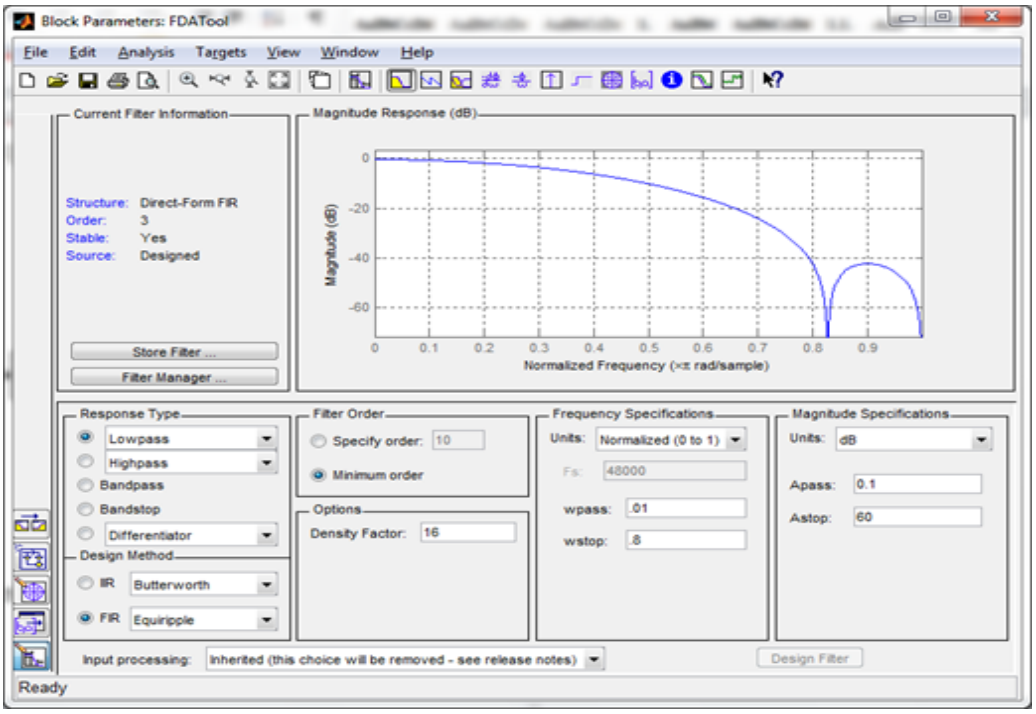
Слика 6.42 *System generator* модел укупног *CIC* дециматора са фактором $N_{mod}=16$

Након извршеног снимања амплитудске карактеристике предложене реализације по принципу одбирак по одбирак, добија се карактеристика која је приказана на Слици 6.43, на којој је упоредо приказана и карактеристика почетног *CIC* филтра трећег реда (к=3).



Слика 6.43 Амплитудска карактеристика модификованог и почетног *CIC* филтра

Са слике се закључује да су обе примењене методе дале добре резултате, јер је применом фреквенцијског маскирања пропусни опсег филтра сужен, а *filter sharpening* методом је увећано слабљење у непропусном опсегу и сужена прелазна зона. На масксималној учестаности одабирања ради само маскирајући филтер, али је он у овом случају трећег реда (Слика 6.44) јер је испоштован услов $l<<N$.



Слика 6.44 Реализација маскирајућег филтра софтверским алатом *FDA tool*

6.4.3 ПРИМЕНА *CIC* ФИЛТАРА ЗА ПРОМЕНУ УЧЕСТАНОСТИ ОДАБИРАЊА СА ВЕЛИКИМ ФАКТОРОМ КОНВЕРЗИЈЕ НА РЕАЛНЕ СИГНАЛЕ

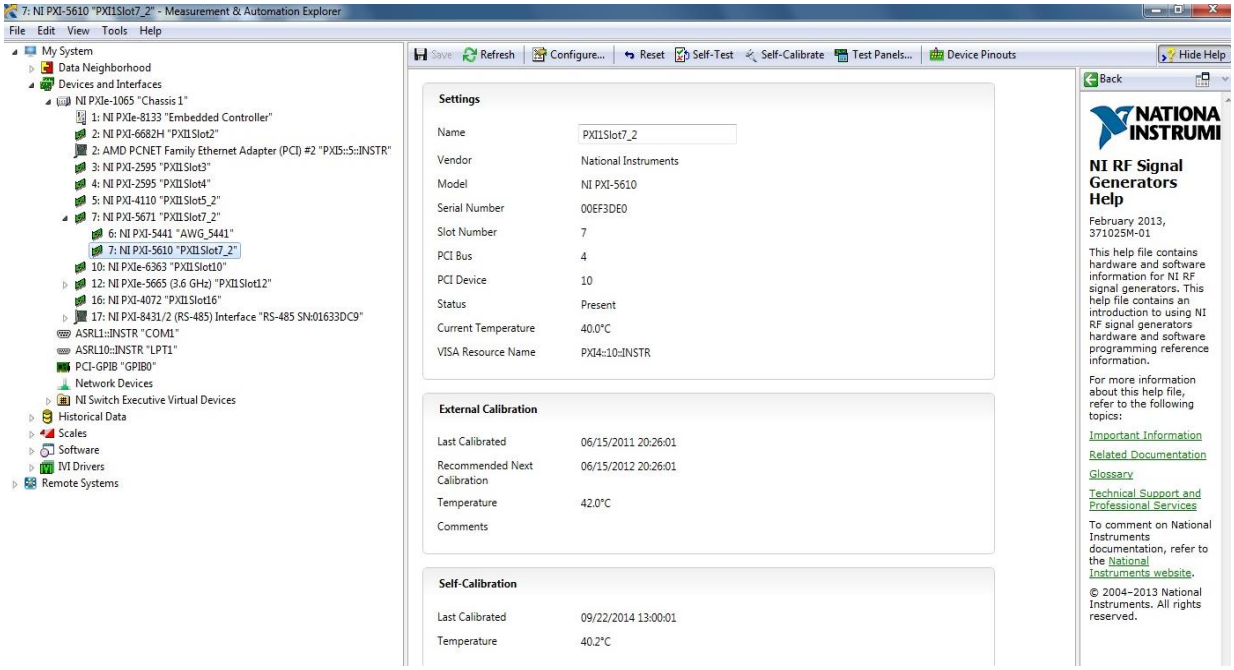
Најреалније испитивање филтерских структура као и кола за промену учестаности одабирања представља снимање сигнала на реалним хардверским платформама. Једна од таквих платформи је NI 7965R FPGA Flex Rio картица (Слика 6.46), смештена у *PXI Express* кућишту произвођача *National Instruments*. На картици се налази *Virtex 5 SX95T FPGA* картица са 512 Mb *DRAM* меморије.

Слика 6.45 *NI 7965R FPGA Flex Rio* картица

Картица у комплету са адаптер модулом (*NI 5734*) и *PXI Express* кућиштем чини комплет са којим можемо проверити структуре предложене у предходном поглављу.

Компатибилност програмског пакета *LabView* са *Xilinx*-овим развојним окружењима овај процес чини једноставним, обзиром да је целокупни процес аутоматизован. Након генерисања *VHDL* кода из *Xilinx* -овог *System Generator* модела, једноставним избором *IP invoke nod-a*, цео модел увозимо у *LabView*, а одатле вршимо завршно програмирање *FPGA* чипа на картици.

Обзиром да је адаптер модул 16-bit., 120 MS/s A/D конвертор, као и да су доступни и аналогни и дигитални улази/излази на конектор блоку, комплет може бити искоришћен за израду дигиталног *down converter*-а, користећи *System Generator* модел са Слике 6.42. Једноставним пребацивањем у програмски пакет *LabView* и програмирањем *FPGA* картице, добијамо могућност да на аналогне улазе конектор блока доведемо неки реални сигнал, а да на њему одговарајућем аналогном излазу добијемо улазни сигнал на фреквенцији *l* пута нижој од фреквенције улазног сигнала (*l* је коефицијент конверзије).

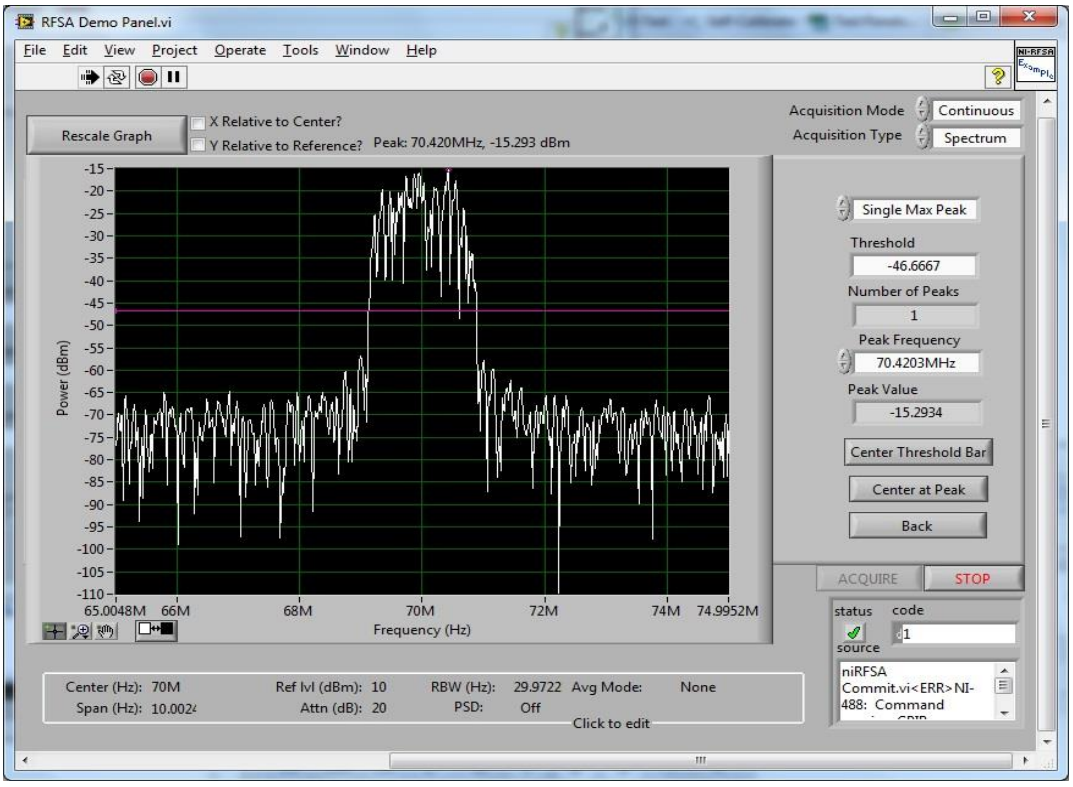


Слика 6.46 Идентификација и провера хардвера

Како би проверу била што приближнија реалним условима, као улазни сигнал ће бити искоришћен *16QAM* модулисани сигнал који се на радио релејном уређају ГРЦ-408Е произвођача ТАДИРАН, јавља на модулу пријемника. На овом модулу се сигнал са фреквенције 70 MHz транслира на међуфреквенцију, а затим врши *16QAM* демодулација и враћање сигнала у основни опсег. На радио-релејном уређају се овај процес врши процесом фреквенцијског мешања, док ћемо ми искористити *FPGA* платформу и

извршити дигиталну *down* конверзију са модификованим *CIC* дециматором са Слике 6.42, и при томе би требало да добијемо резултат идентичан оригиналном.

Процес почињемо идентификацијом расположивог хардвера користећи *MAX* (*Measurement and Automation Explorer*) алат, као што је то приказано на Слици 6.46. Након идентификације и тестирања хардвера, на конектор блок се доводи *16QAM* модулисани сигнал на фреквенцији 70 MHz, са радио релејног уређаја (Слика 6.47).

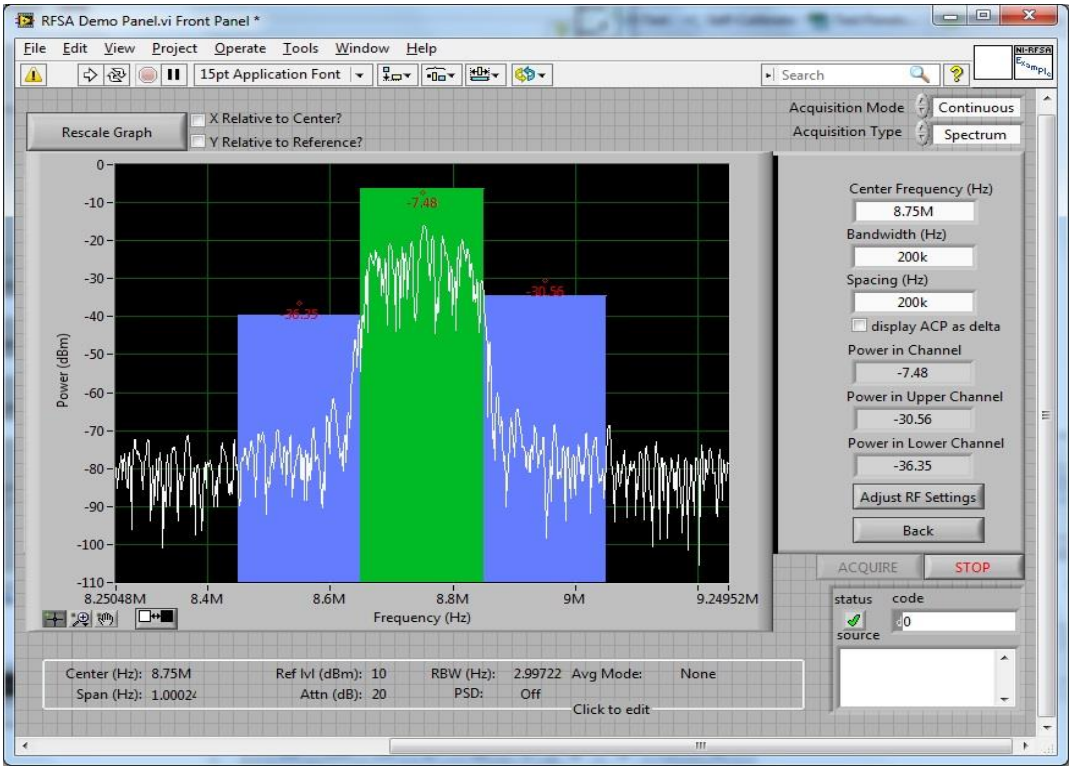


Слика 6.47 *70 MHz 16QAM* модулисани сигнал са радио релејног уређаја

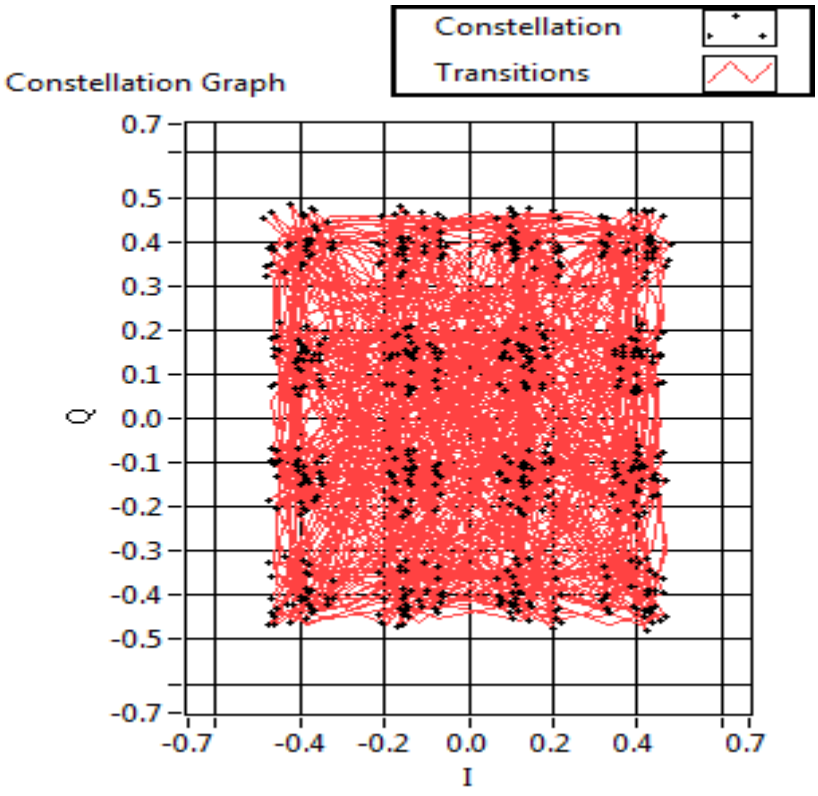
Затим се овакав сигнал води на улаз дигиталног *down* конвертора са фактором 16 (према Слици 6.42). Након дигиталне *down* конверзије, на одговарајућем аналогном излазу је снимљен спектар излазног сигнала на анализатору спектра, као и одговарајућа снага у каналу и ширина пропусног опсега. Као што видимо са Слике 6.48, транслирање спектра је извршено без деградације сигнала по амплитуди и фазних изобличења на фреквенцију 70MHz / 16 = 8,75 MHz.

Затим је искоришћена могућност *16QAM* демодулације сигнала користећи картицу *NI 5610* која је уграђена у кућиште инструмента, и снимљен је констелациони (*I*Q) дијаграм демодулисаног сигнала након модификованог *CIC down* конвертера, који је приказана на Слици 6.49. Као што видимо, *IQ* дијаграм савршено одговара дијаграму *16QAM*

модулисаног сигнала, што говори да се предложена метода успешно може применити и у реалним условима.



Слика 6.48 *70 MXз 16QAM* модулисани сигнал са радио релејног уређаја



Слика 6.49 Констелациони дијаграм

7. ЗАКЉУЧАК

Дигитално филтрирање добија све значајнију улогу у модерним системима дигиталног процесирања сигнала. Овако нешто условљено је све већим бројем апликација које се јављају у области телекомуникација. Потребе за све више хардверских ресурса постају све израженије и поред развоја нових технолошких решења. Из тог разлога структуре дигиталних филтара се мењају и прилагођавају како би се извршила рационализација хардверских ресурса и како би се што ефикасније реализовале захтеване апликације у реалном времену.

Поред увећања и снижавања учестаности одабирања, кола за конверзију учестаности одабирања заједно са другим основним елементима (сабирачима, множачима, колима за кашњење) могу бити корисна за побољшање перформанси дигиталних система. У том смислу је развијен поступак проточне обраде сигнала (енгл. *Pipelining –Interleaving)* тј. *PI* поступак са циљем да се побољша ефикасност дигиталног филтрирања. Његовом употребом у реализацији различитих задатака, поред тога што се омогућава једноставније решавање различитих проблема, постижу се значајне уштеде у хардверским ресурсима. Наиме, применом проточне обраде омогућено је да се сигнали обрађују паралелно, уместо да се та обрада обавља секвенцијално. Редак је случај да су учестаност одабирања и такт дигиталног система усаглашени и то нарочито у случајевима када се дигитални филтри реализују програмабилним хардвером. Тако, уколико је учестаност одабирања знатно нижа од учестаности такта на којој раде елементи система, примена паралелног хардвера непотребно троши ресурсе расположивих елемената на чипу из тог разлога што када се обаве неопходне операције елементи чекају долазак наредног одбирка који би требао да се процесира. Иако би хардвер у међувремену могао да се користи и за друге улоге, то се не чини јер то не дозвољава конфигурација система и редослед операција.

У овом раду су развијена нека решења и изложена методологија развоја нових реализација дигиталних филтара применом појединих *multirate* метода. Поред тога, извршена је примена *PI* технике на реализацију дигиталних структура са различитим брзинама обраде. За свако предложено решење извршена је провера рада и анализа нове структуре у реалном времену. Испитивања су вршена снимањем и упоређивањем амплитудских карактеристика старих и нових структура коришћењем програмског пакета *Matlab*, изузев анализе *IIR/FIR* филтара за велике брзине обраде која је вршена помоћу програмског пакета *Matematica*.

Извршено је испитивање директне примене *PI* поступка као и примена модификованог PI поступка [1] за реализацију каскадне везе идентичних филтара на дигиталне филтре са коначним и филтре са бесконачним импулсним одзивом. Поред овога извршена је провера утицаја ефекта коначне дужине кодне речи. Резултати су показали да се *PI* поступак успешно може применити на овакве структуре, с тим што код филтара са бесконачним импулсним одзивом морамо применити одређена решења при избору филтара, како би задовољили услове критичне повратне петље, а што је условљено сложенијом структуром *IIR* филтара на које примењујемо *PI* поступак.

Затим је извршена примена *PI* техника на вишестепене ускопојасне дигиталне филтре. Предложено је решење за реализацију вишестепених филтара када је број степени велики (на пример $M>10$) изведено је прекомбиновањем операција структуре. Показало се да је примена оваквог решења нарочито ефикасна у случају реализације више оваквих канала, али да и у овом случају због величине грешке (шума реализације) морамо водити рачуна о броју канала и о строгости захтева код ускопојасних филтара, како ова грешка не би постала превелика.

Показано је затим да се *PI* поступак успешно може применити и на фреквенцијско маскирање за случај када се филтрирање врши за више идентичних канала. На начин сличан као и код вишестепених филтара, комбиновањем операција које проистичу из *PI* поступка и операција фреквенцијског маскирања, може се извести ефикасна структура која захтеване карактеристике остварује са минималном грешком.

QMF банке са структуром стабла су такође добар кандидат за примену *PI* поступка. Захваљујући својој структури, оваквим банкама није потребно никакво прекомбиновање операција, већ се *PI* поступак може директно применити тако да нема никаквих ограничења у погледу броја канала. Показано је да одступања од оригиналне карактеристике зависе једино од избора филтара у банкама анализе и синтезе.

На *FIR* филтер са *IIR* филтерским секцијама другог реда за велике брзине обраде успешно се може применити принцип проточности. Приказано је једно оригинално решење реализације оваквог филтра применом *PI* технике. Анализа оваквог филтра извршена је програмским пакетом *Matematica* и софтверским алатом *SchematicSolver Version 2* применом симболичког процесирања. Софтверски алати који омогућавају манипулацију симболима, а не само бројевима веома су ефикасни у развоју нових и неконвенцијалних алгоритама. У датом примеру је њихова употреба била неопходна јер је било потребно познавање како улазног сигнала тако и оних одбирака до чијих вредности се долази тек након обраде неких од улазних одбирака. Посебно је важно да се неки од

параметара система, чије је познавање неопходно пре процесирања сигнала, могу одредити тек на крају обраде.

Затим је показано да се предложене методе успешно могу применити и на реализацију сложенијих дигиталних структура, чија реализација није могућа коришћењем стандардних метода и софтверских алата које нуде произвођачи програмабилног хардвера. Дигиталне структуре су реализоване на реалном хардверу и у техници са фиксним зарезом. При томе није дошло до деградације захтеваних карактеристика, а извршене су значајне уштеде хардверких ресурса. Провера предложених структура осим симулацијом, извршена је и са реалним сигналима и приказан је одзив једне такве структуре на *16QAM* модулисани сигнал.

8. ЛИТЕРАТУРА

[1] Zhongnong Jiang; Willson, A.N., Jr., "Efficient digital filtering architectures using pipelining/interleaving," *Circuits and Systems II: Analog and Digital Signal Processing, IEEE Transactions on* , vol.44, no.2, pp.110,119, Feb 1997 doi: 10.1109/82.554438

[2] Lim, Y.C., "Parallel and pipelined implementations of injected numerator lattice digital filters," *Circuits and Systems II: Analog and Digital Signal Processing, IEEE Transactions on* , vol.42, no.7, pp.480,486, Jul 1995, doi: 10.1109/82.401173.

[3] P.P Vaidyanathan, "Quadrature Mirror Filter Banks, M-band Extensions and Perfect Reconstruction Techniques" *IEEE ASSP Magazine,* Vol. 4, No.3, pp. 4-20, July 1987 ISSN 0740-3224.

[4] Vaidyanathan, P.P., "Multirate digital filters, filter banks, polyphase networks, and applications: a tutorial," *Proceedings of the IEEE* , vol.78, no.1, pp.56,93, Jan 1990, doi: 10.1109/5.52200.

[5] Parhi, K.K.; Messerschmitt, D.G., "Pipeline interleaving and parallelism in recursive digital filters. I. Pipelining using scattered look-ahead and decomposition," *Acoustics, Speech and Signal Processing, IEEE Transactions on* , vol.37, no.7, pp.1099,1117, Jul 1989, doi: 10.1109/29.32286.

[6] K. K. Parhi and D. G. Messerschmitt, "Pipeline interleaving and parallelism in recursive digital filters-Part II**:** Pipelined incremental block filtering," *IEEE Trans. Acoust., Speech, Signal Processing,* 08/1989; 37(7):1118 – 1134, doi: 10.1109/29.32287.

[7] Jiang, Z.; Willson, A.N., Jr., "A pipelined/interleaved IIR digital filter architecture," *Acoustics, Speech, and Signal Processing, 1997. ICASSP-97., 1997 IEEE International Conference on* , vol.3, no., pp.2217,2220 vol.3, 21-24 Apr 1997, doi: 10.1109/ICASSP.1997.599491.

[8] Schwartz, D.; Barnwell, T.P., "Increasing the parallelism of filters through transformation to block state variable form," *Acoustics, Speech, and Signal Processing, IEEE International*

Conference on ICASSP '84., vol.9, no., pp.610,613, Mar 1984, doi: 10.1109/ICASSP.1984.1172622.

[9] Ciric, M.; Radonjic, V., "Realization of multistage FIR digital filters using pipelininginterleaving," *Telecommunications Forum (TELFOR), 2011 19th* , vol., no., pp.758,761, 22-24 Nov. 2011, doi: 10.1109/TELFOR.2011.6143655.

[10] Vaidyanathan, P.P., "Multirate digital filters, filter banks, polyphase networks, and applications: a tutorial," *Proceedings of the IEEE* , vol.78, no.1, pp.56,93, Jan 1990 doi: 10.1109/5.52200.

[11] Ekanayake, M.M.; Premaratne, K., "Two-channel IIR QMF banks with approximately linear-phase analysis and synthesis filters," *Signal Processing, IEEE Transactions on* , vol.43, no.10, pp.2313,2322, Oct 1995, doi: 10.1109/78.469858.

[12] Lee, J.-H.; Niu, I.-C., "Minimax design of two-channel IIR QMF banks with arbitrary group delay," *Vision, Image and Signal Processing, IEE Proceedings* - , vol.148, no.6, pp.384,390, Dec 2001, doi: 10.1049/ip-vis:20010673.

[13] Meng, Teresa H.-Y.; Messerschmitt, D.G., "Implementations of arbitrarily fast adaptive lattice filters with multiple slow processing elements," *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP '86.* , vol.11, no., pp.1153,1156, Apr 1986 doi: 10.1109/ICASSP.1986.1168832.

[14] Jingjing Tian; Guangjun Li; Qiang Li, "Hardware-efficient parallel structures for linear-phase FIR digital filter," *Circuits and Systems (MWSCAS), 2013 IEEE 56th International Midwest Symposium on* , vol., no., pp.995,998, 4-7 Aug. 2013,
doi: 10.1109/MWSCAS.2013.6674819.

[15] Moyer, A., "An efficient parallel algorithm for digital IIR filters," *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP '76.* , vol.1, no., pp.525,528, Apr 1976, doi: 10.1109/ICASSP.1976.1170109.

[16] Lutovac, M.D.; Tosic, D.V., "High-Speed Filter Design using Mathematica," *Computer as a Tool, 2005. EUROCON 2005.The International Conference on* , vol.2, no., pp.1626,1629, 21-24 Nov. 2005, doi: 10.1109/EURCON.2005.1630281.

[17] Evans, B.L.; McClellan, J.H.; Trussell, H.J., "Investigating signal processing theory with Mathematica," *Acoustics, Speech, and Signal Processing, 1993. ICASSP-93., 1993 IEEE International Conference on* , vol.1, no., pp.12,15 vol.1, 27-30 April 1993 doi: 10.1109/ICASSP.1993.319041.

[18] M. Lutovac, D. Tošić, SchematicSolver Version 2, *http://www.wolfram.com/ products/ applications/ schematicsolver*.

[19] Rabiner, L.; Crochiere, R.E., "A novel implementation for narrow-band FIR digital filters," *Acoustics, Speech and Signal Processing, IEEE Transactions on* , vol.23, no.5, pp.457,464, Oct 1975, doi: 10.1109/TASSP.1975.1162727

[20] Haddad, K.C.; Stark, H.; Galatsanos, N.P., "Constrained FIR filter design by the method of vector space projections," *Circuits and Systems II: Analog and Digital Signal Processing, IEEE Transactions on* , vol.47, no.8, pp.714,725, Aug 2000, doi: 10.1109/82.861404T.

[21] Laddomada, M.; Jovanovic Dolecek, G.; Lim Yong Ching; FA-LONG Luo; Renfors, M.; Wanhammar, L., "Advanced techniques on multirate signal processing for digital information processing [Editorial],"*Signal Processing, IET* , vol.5, no.3, pp.313,315, June 2011, doi: 10.1049/iet-spr.2011.9058.

[22] Ramstad, T.A.; Saramaki, T., "Multistage, multirate FIR filter structures for narrow transition-band filters," *Circuits and Systems, 1990., IEEE International Symposium on* , vol., no., pp.2017,2021 vol.3, 1-3 May 1990, doi: 10.1109/ISCAS.1990.112143.

[23] Lutovac, M.D.; Milic, L.D., "Design of computationally efficient elliptic IIR filters with a reduced number of shift-and-add operations in multipliers," *Signal Processing, IEEE Transactions on* , vol.45, no.10, pp.2422,2430, Oct 1997, doi: 10.1109/78.640708.

[24] Trirat, A.; Chivapreecha, S.; Khunaworawet, T.; Ruangrangsan, T.; Dejhan, K., "Design of multiplierless elliptic narrowband IIR digital filter based on sensitivity analysis," *Communications and Information Technology, 2004. ISCIT 2004. IEEE International Symposium on* , vol.1, no., pp.171,177 vol.1, 26-29 Oct. 2004, doi: 10.1109/ISCIT.2004.1412473.

[25] *System Generator for DSP*, release 10.1, March, 2010, www.xilinx.com

[26] Filter design toolbox for use with MATLAB. User's guide, *The MathWorks Inc., 3* Apple Hill Drive, Natick, MA, 2006.

[27] Signal processing toolbox for use with MATLAB. User's guide, *The MathWorks Inc., 3* Apple Hill Drive, Natick, MA, 2006.

[28] Jiang, Z.; Willson, A.N., Jr., "A pipelined/interleaved IIR digital filter architecture," *Acoustics, Speech, and Signal Processing, 1997. ICASSP-97., 1997 IEEE International Conference on* , vol.3, no., pp.2217,2220 vol.3, 21-24 Apr 1997 doi: 10.1109/ICASSP.1997.599491.

.

[29] Saramaki, T., "Design of FIR filters as a tapped cascaded interconnection of identical subfilters," *Circuits and Systems, IEEE Transactions on* , vol.34, no.9, pp.1011,1029, Sep 1987 doi: 10.1109/TCS.1987.1086263.

[30] Vaidyanathan, P.P.; Regalia, P.; Mitra, S.K., "Design of doubly-complementary IIR digital filters, using a single complex allpass filter," *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP '86.* , vol.11, no., pp.2547,2550, Apr 1986 doi: 10.1109/ICASSP.1986.1169285.

[31] Mitra, S.K.; Neuvo, Y.; Vaidyanathan, P.P., "Complementary IIR digital filter banks," *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP '85.* , vol.10, no., pp.529,532, Apr 1985, doi: 10.1109/ICASSP.1985.1168359.

[32] Vaidyanathan, P.P.; Regalia, P.; Mitra, S.K., "Design of doubly-complementary IIR digital filters using a single complex allpass filter, with multirate applications," *Circuits and*

Systems, IEEE Transactions on , vol.34, no.4, pp.378,389, Apr 1987 doi: 10.1109/TCS.1987.1086156.

[33] Milic, L.; Saramaki, T., "Three classes of IIR complementary filter pairs with an adjustable crossover frequency," *Circuits and Systems, 2003. ISCAS '03. Proceedings of the 2003 International Symposium on* , vol.4, no., pp.IV-145,IV-148 vol.4, 25-28 May 2003 doi: 10.1109/ISCAS.2003.1205794.

[34] Kaiser, J.; Hamming, R., "Sharpening the response of a symmetric nonrecursive filter by multiple use of the same filter," *Acoustics, Speech and Signal Processing, IEEE Transactions on* , vol.25, no.5, pp.415,422, Oct 1977, doi: 10.1109/TASSP.1977.1162980.

[35] V. M. Poucki and A. Žemva and M. D. Lutovac and T. Karčnik "Elliptic IIR filter sharpening implemented on FPGA," *Digital Signal Processing* 20 (2010) 13–22, May 2009, 10.1016/j.dsp.2009.04.012.

[36] Lutovac, M.D.; Certic, J.; Milic, L., "A class of digital filters with variable cut-off based on EMQF filter sections and sharpening method," *Circuit Theory and Design (ECCTD), 2011 20th European Conference on* , vol., no., pp.13,16, 29-31 Aug. 2011, doi: 10.1109/ECCTD.2011.6043298.

[37] Lim, Y.C.; Yong Lian, "Frequency-response masking approach for digital filter design: complexity reduction via masking filter factorization," *Circuits and Systems II: Analog and Digital Signal Processing, IEEE Transactions on* , vol.41, no.8, pp.518,525, Aug 1994, doi: 10.1109/82.318940.

[38] Yong Ching Lim; Rui Yang, "On the synthesis of very sharp decimators and interpolators using the frequency-response masking technique," *Signal Processing, IEEE Transactions on* , vol.53, no.4, pp.1387,1397, April 2005, doi: 10.1109/TSP.2005..

[39] Jovanovic-Dolecek, G.; Mitra, S.K., "A new two-stage sharpened comb decimator," *Circuits and Systems I: Regular Papers, IEEE Transactions on* , vol.52, no.7, pp.1414,1420, July 2005, doi: 10.1109/TCSI.2005.851390.

[40] Aboushady, H.; Dumonteix, Y.; Louerat, M.M.; Mehrez, H., "Efficient polyphase decomposition of Comb decimation filters in ΣΔ analog-to-digital converters," *Circuits and Systems, 2000. Proceedings of the 43rd IEEE Midwest Symposium on* , vol.1, no., pp.432,435 vol.1, 2000, doi: 10.1109/MWSCAS.2000.951676.

[41] Lo Presti, L., "Efficient modified-sinc filters for sigma-delta A/D converters," *Circuits and Systems II: Analog and Digital Signal Processing, IEEE Transactions on* , vol.47, no.11, pp.1204,1213, Nov 2000, doi: 10.1109/82.885128.

[42] Laddomada, M., "Generalized Comb Decimation Filters for ΣΔ A/D Converters: Analysis and Design," *Circuits and Systems I: Regular Papers, IEEE Transactions on* , vol.54, no.5, pp.994,1005, May 2007, doi: 10.1109/TCSI.2007.895528.

[43] Laddomada, M., "Comb-Based Decimation Filters for A/D Converters: Novel Schemes and Comparisons," *Signal Processing, IEEE Transactions on* , vol.55, no.5, pp.1769,1779, May 2007, doi: 10.1109/TSP.2006.890822.

[44] Smitha, K.G.; Vinod, A.P., "A new low complexity reconfigurable channel filter architecture for software radio handsets," *Communication Systems, 2008. ICCS 2008. 11th IEEE Singapore International Conference on* , vol., no., pp.1334,1338, 19-21 Nov. 2008 doi: 10.1109/ICCS.2008.4737400.

9. РЕЧНИК СКРАЋЕНИЦА И ИЗРАЗА

16QAM – 16-то нивовска квадратурна амплитудска модулација.

Aliasung – Ефекат који се јавља приликом компресије сигнала када услед преклапања периодичних спектара није могуће издвојити корисни спектар.

Allpass filters – Свепропусни филтри – Филтри пропусници целокупног опсега учестаности.

Downsampler – kompresor – коло за снижење учестаности одабирања.

DDC Digutal down converter– дигитални *down* конвертер – коло за снижење радне фреквенције.

Frequency response masking technique – Техника фреквенцијског маскирања.

FFT – Fast Fourier Transform – Брза Фуријеова трансформација.

FIR filters – Finute Impulse Response filters - Филтри са коначним импулсним одзивом.

Halfband filters – Полуопсежни филтри.

IIR fulters – Infinute Impulse Response filters – Филтри са бесконачним импулсним одзивом.

Imaguig – Ефекат који се јавља приликом експанзије сигнала када се експандовањем сигнала у временском домену за фактор L проузрокује појава L-1 реплика у фреквенцијском домену.

Kernel filtar – (*kernel* – срж, суштина) – Један од ступњева вишестепених (*multistage*) филтара пројектован за филтрирање улазног сигнала али на нижој учестаности одабирања.

Masking filters – Маскирајући филтри – Филтри који се користе код технике фреквенцијског маскирања за елиминисање периодичних одраза проузрокованих периодичним модел филтром.

Multirate systems – Вишебрзински системи, тј, системи са различитим учестаностима одабирања у одређеним тачкама система.

Multistage filters - Вишестепени филтри пројектовани тако да се филтрирање врши на нижој учестаности одабирања а затим се врши повратак на стару учестаност.

PI technique – Pipelinig Interleaving technique – Техника проточне обраде сигнала.

Tree structured QMF filters bank – QMF банке са структуром стабла.

Upsampler – Експандор - Коло за увећање учестаности одабирања.

10. ПРЕГЛЕД КОРИШЋЕНИХ ПРОГРАМА

У циљу анализе сваке примене технике проточне обраде сигнала описане у овом раду извршено је формирање симулационог модела. За све примене изузев реализације филтра за велике брзине обраде, симулациони модел је израђен у програмском пакету *Matlab*. Израда модела је вршена по принципу „одбирак по одбирак“ тако да су програми релативно обимни. Симулациони модел за анализу примене *PI* технике на филтер за велике брзине обраде израђен је у програму *Matematica*. За овај пример, поред наредби, приказане су и пропратне слике које су генерисане у току израде модела као и резултати.
У поглављу 6, сви модели су реализовани користећи *Xilinx*-ов алат System Generator који представља графички развојни програм код кога је код садржан у елементима као и у везама између модула и не представља текстуални програмски језик. Из тог разлога, код структуре је је приказан на сликама модела.

10.1 *FIR PI* СТРУКТУРА

```
clear all
% Izabracemo proizvoljni FIR filtar:
aa=35;
ap=0.2;
dev=[(10^(ap/20)-1)/(10^(ap/20)+1),10^(-aa/20)];
[Nf,fp,mag,wt]=remezord([50 70],[1 0],dev,256);
b1=remez(Nf,fp,mag,wt);
[H1,f1]=freqz(b1,1,256,256);
plot(f1/127,20*log10(abs(H1))), xlabel('\omega/\pi'),
ylabel('pojacanje [dB]')
axis([0 1 -60 10])
% Pa ćemo da formiramo Hz(z2)ubacivanjem po jedne nule:
for t=1:2*length(b1)-1
if ((-1).^t)+1==0
b(1,t)=b1(1,t-(t-1)/2);
else
b(1,t)=0;
end
end
% Formiramo ulazni signal frekvencije 1:127
n=0:1023;
Nx=1024;
R=length(b);
for k=1:127
xa=sin(2*pi*n*k/256);
% Ulazni signal ce sad da traje do Nx+R-1
N=Nx+length(b1)-1;
y1(1,1)=xa(1,1);
y2(1,1)=0;
y4(1,1)=y1(1,1)+y2(1,1);
y6(1,1)=b(1,1)*xa(1,1);
y8(1,1)=0;
y9(1,1)=0;
y7(1,1)=y6(1,1);
for m=2:2*N+R
if m<=2*N
if m<=2*Nx
if ((-1).^m)+1==0
y1(1,m)=xa(1,m-(m-1)/2);
else
y1(1,m)=0;
end
else
y1(1,m)=0;
end
y2(1,m)=y7(1,m-1);
y4(1,m)=y1(1,m)+y2(1,m);
x=y4(1:m);
y5=conv(x,b);
y6(1,m)=y5(1,m);
```

```
else
if m<=2*N+R-1
y6(1,m)=y5(1,m);
else
end
end
if m<=2*N+R-1
if ((-1).^m)+1==0
y7(1,m)=y6(1,m);
else
y7(1,m)=0;
end
else
end
y8(1,m)=y6(1,m-1);
if ((-1).^m)+1==0
y9(1,m-(m-1)/2)=y8(1,m);
else
end
end
y9e=y9(256:768);
Y9e=fft(y9e,256);
Xa=fft(xa,256);
% Sad formiramo matricu reda 127*256 (127 frekvencija, a 256-
broj tacaka FFT)
for d=1:length(Y9e)
Y(k,d)=Y9e(1,d);
X(k,d)=Xa(1,d);
end
% Matrica izlaza u vremenskom domenu
for t=1:length(y9e)
yi(k,t)=y9e(1,t);
end
end
% Za svaku frekvenciju racunamo izlaz na toj frekvenciji
for r=1:127
Yi(1,r)=Y(r,r+1);
Xu(1,r)=X(r,r+1);
end
j=1:127;
% Karakteristika racunata odbirak po odbirak
figure (2)
plot(j/127,20*log10(abs(Yi)./abs(Xu))), xlabel('\omega/\pi'),
ylabel('pojacanje [dB]')
axis([0 1 -120 20])
g(1)=axes('Position',[0.56 0.70 0.32 0.2]);
plot(j/127,20*log10(abs(Yi)./abs(Xu))), axis([0 0.4 -0.5
0.5]),grid
% Izracunavanje H(z)*H(z)
bu=conv(b1,b1);
[H,f]=freqz(bu,1,256,256);
figure (3)
```

```
plot(f/127,20*log10(abs(H))), xlabel('\omega/\pi'),
ylabel('pojacanje [dB]')
axis([0 1 -120 20])
g(2)=axes('Position',[0.56 0.70 0.32 0.2]);
plot(f/127,20*log10(abs(H))), axis([0 0.4 -0.5 0.5]),grid
% prikaz u vremenskom domenu za nekoliko ulaznih signala
xa1=sin(2*pi*n*1/256);
xa10=sin(2*pi*n*10/256);
xa60=sin(2*pi*n*60/256);
xa120=sin(2*pi*n*120/256);
ya1=conv(xa1,bu);
ya10=conv(xa10,bu);
ya60=conv(xa60,bu);
ya120=conv(xa120,bu);
% prikazacemo sada na istom grafiku ove izlaze sa izlazima na
istim frekvencijama racunatim prema nasem algoritmu
figure (4)
% Pomeranje izlaza iz filtra za jedan odbirak
ya1k=[0 ya1];
ya10k=[0 ya10];
ya60k=[0 ya60];
ya120k=[0 ya120];
ya1ke=ya1k(256:768);
ya10ke=ya10k(256:768);
ya60ke=ya60k(256:768);
ya120ke=ya120k(256:768);
b=1:length(ya1ke);
v=1:length(y9e);
subplot(411), plot(b,ya1ke,'r.',v,yi(1,:),'g'),
title('x=sin(2*pi*n*1/256)')
subplot(412), plot(b,ya10ke,'r.',v,yi(10,:),'g'),
title('x=sin(2*pi*n*10/256)')
subplot(413), plot(b,ya60ke,'r.',v,yi(60,:),'g'),
title('x=sin(2*pi*n*60/256)')
subplot(414), plot(b,ya120ke,'r.',v,yi(120,:),'g'),
title('x=sin(2*pi*n*120/256)')
for q=1:127
xa=sin(2*pi*n*q/256);
yai1=conv(xa,b1);
yai2=conv(yai1,b1);
yai3=[0 yai2];
yai4=yai3(256:768);
razlika=yai4-yi(q,:);
greska(1,q)=max(razlika);
end
maxgreska=max(greska)
```

10.2 *IIR PI* СТРУКТУРА

```
clear ll
[N,wn]=ellipord(0.4,0.5,0.5,36);
```

```
[b,a]=ellip(5,0.5,36,0.4);
 [Z,L,K]=tf2zp(b,a);
den1=conv([1,-L(1,1)],[1,-L(2,1)]);
den2=conv([1,-L(3,1)],[1,-L(4,1)]);
k0=poly2rc(den1);
k1=poly2rc(den2);
knew0=fliplr(k0);
knew1=fliplr(k1);
n=0:1023;
N=1024;
for k=1:127
x1=sin(2*pi*n*k/256);
y1(1,1)=x1(1,1);
y2(1,1)=0;
y4(1,1)=y1(1,1)+y2(1,1);
q1(1,1)=0.5*y4(1,1);
q2(1,1)=q1(1,1);
q3(1,1)=0;
q4(1,1)=knew0(1,1)*q2(1,1);
q5(1,1)=0;
q6(1,1)=knew0(2,1)*q1(1,1);
q7(1,1)= q6(1,1);
q8(1,1)=0;
y6a(1,1)=-L(5,1)*q7(1,1);
p1(1,1)=0.5*y4(1,1);
p2(1,1)=p1(1,1);
p3(1,1)=0;
p4(1,1)=knew1(1,1)*p2(1,1);
p5(1,1)=0;
y6b(1,1)=knew1(2,1)*p1(1,1);
y6(1,1)=y6a(1,1)+y6b(1,1);
y7(1,1)=y6(1,1);
y8(1,1)=0;
y9(1,1)=y8(1,1);
y1(1,2)=0;
y4(1,2)=y1(1,2);
q1(1,2)=0.5*y4(1,2);
q2(1,2)=q1(1,2);
q3(1,2)=0;
q4(1,2)=knew0(1,1)*q2(1,2);
q5(1,2)=0;
q6(1,2)=knew0(2,1)*q1(1,2);
q7(1,2)= q6(1,2);
q8(1,2)=0;
y6a(1,2)=-L(5,1)*q7(1,2);
p1(1,2)=0.5*y4(1,2);
p2(1,2)=p1(1,2);
p3(1,2)=0;
p4(1,2)=knew1(1,1)*p2(1,2);
p5(1,2)=0;
y6b(1,2)=knew1(2,1)*p1(1,2);
y6(1,2)=y6a(1,2)+y6b(1,2);
```

```
y7(1,2)=0;
y8(1,2)=y6(1,1);
y9(1,2)=0;
for m=3:2*N+4
if m<=2*N
if ((-1).^m)+1==0
y1(1,m)=x1(1,m-(m-1)/2);
else
y1(1,m)=0;
end
else
y1(1,m)=0;
end
y4(1,m)=y1(1,m)+y7(1,m-1);
q1(1,m)=0.5*y4(1,m)-knew0(2,1)*q4(1,m-2);
q2(1,m)=q1(1,m)-knew0(1,1)*q2(1,m-2);
q3(1,m)=q2(1,m-2);
q4(1,m)=q3(1,m)+knew0(1,1)*q2(1,m);
q5(1,m)=q4(1,m-2);
q6(1,m)=q5(1,m)+knew0(2,1)*q1(1,m);
q7(1,m)=q6(1,m)+L(5,1)*q7(1,m-2);
q8(1,m)=q7(1,m-2);
y6a(1,m)=q8(1,m)-L(5,1)*q7(1,m);
p1(1,m)=0.5*y4(1,m)-knew1(2,1)*p4(1,m-2);
p2(1,m)=p1(1,m)-knew1(1,1)*p2(1,m-2);
p3(1,m)=p2(1,m-2);
p4(1,m)=p3(1,m)+knew1(1,1)*p2(1,m);
p5(1,m)=p4(1,m-2);
y6b(1,m)=p5(1,m)+knew1(2,1)*p1(1,m);
y6(1,m)=y6a(1,m)+y6b(1,m);
if ((-1).^m)+1==0
y7(1,m)=y6(1,m);
else
y7(1,m)=0;
end
y2(1,m)=y7(1,m-1);
y4(1,m)=y2(1,m)+y1(1,m);
y8(1,m)=y6(1,m-1);
if ((-1).^m)+1==0
y9(1,m-(m-1)/2)=y8(1,m);
else
end
end
y9e=y9(256:768);
Y9e=fft(y9e,256);
Xa=fft(x1,256);
% Sad formiramo matricu reda 127*256 (127 frekvencija, a 256-
broj tacaka FFT)
for d=1:length(Y9e)
Y(k,d)=Y9e(1,d);
X(k,d)=Xa(1,d);
end
```

```
% Matrica izlaza u vremenskom domenu
for t=1:length(y9e)
yi(k,t)=y9e(1,t);
end
end
% Za svaku frekvenciju racunamo izlaz na toj frekvenciji
for r=1:127
Yi(1,r)=Y(r,r+1);
Xu(1,r)=X(r,r+1);
end
j=1:127;
[H,f]=freqz(b,a,256,256);
figure (1)
plot(f/127,20*log10(abs(H))), xlabel('w/pi'), ylabel('pojacanje
[dB]')
axis([0 1 -80 0])
% Karakteristika racunata odbirak po odbirak
figure (2)
plot(j/127,20*log10(abs(Yi)./abs(Xu))), xlabel('w/pi'),
ylabel('pojacanje [dB]')
g(1)=axes('Position',[0.56 0.70 0.32 0.2]);
plot(j/127,20*log10(abs(Yi)./abs(Xu))), axis([0 0.4 -1.5
0]),grid
% Izracunavanje H(z)*H(z)
b1=conv(b,b);
a1=conv(a,a);
[H1,f]=freqz(b1,a1,256,256);
figure (3)
plot(f/127,20*log10(abs(H1))), xlabel('w/pi'), ylabel('pojacanje
[dB]')
axis([0 1 -140 0])
g(2)=axes('Position',[0.56 0.70 0.32 0.2]);
plot(f/127,20*log10(abs(H1))), axis([0 0.4 -1.5 0]),grid
xa1=sin(2*pi*n*1/256);
xa105=sin(2*pi*n*105/256);
xa120=sin(2*pi*n*120/256);
xa127=sin(2*pi*n*127/256);
ya1=filter(b1,a1,xa1);
ya105=filter(b1,a1,xa105);
ya120=filter(b1,a1,xa120);
ya127=filter(b1,a1,xa127);
% prikazacemo sada na istom grafiku ove izlaze sa izlazima na
istim frekvencijama racunatim prema nasem algoritmu
figure (4)
% Pomeranje izlaza iz filtra za jedan odbirak
ya1k=[0 ya1];
ya105k=[0 ya105];
ya120k=[0 ya120];
ya127k=[0 ya127];
ya1ke=ya1k(256:768);
ya105ke=ya105k(256:768);
ya120ke=ya120k(256:768);
```

```
ya127ke=ya127k(256:768);
c=1:length(ya1ke);
v=1:length(y9e);
subplot(411), plot(c,ya1ke,'r.',v,yi(1,:),'g'),
title('x=sin(2*pi*n*1/256)')
subplot(412), plot(c,ya105ke,'r.',v,yi(105,:),'g'),
title('x=sin(2*pi*n*105/256)')
subplot(413), plot(c,ya120ke,'r.',v,yi(120,:),'g'),
title('x=sin(2*pi*n*120/256)')
subplot(414), plot(c,ya127ke,'r.',v,yi(127,:),'g'),
title('x=sin(2*pi*n*127/256)')
for q=1:127
xa=sin(2*pi*n*q/256);
yai1=filter(b,a,xa);
yai2=filter(b,a,yai1);
yai3=[0 yai2];
yai4=yai3(256:768);
razlika=yai4-yi(q,:);
greska(1,q)=max(razlika);
end
maxgreska=max(greska)
Q= fft(greska,256);
w=1:256;
figure (5)
plot(w/127,20*log10(abs(Q))), xlabel('w/pi'),
ylabel('greska[dB]')
axis([0 1 -285 -245])
maxgreska=max(greska)
```

10.3 ВИШЕСТЕПЕНИ (*MULTISTAGE*) ФИЛТРИ

```
n=0:255;
x=cos(2*pi*n*100/2000)+0.1*sqrt(randn(size(n)));
[X,f]=freqz(x,1,256,2000);
x1=downsample(x,10);
x12=upsample(x1,2);
x2=x(2:256);
x22=downsample(x2,10);
x23=upsample(x22,2);
x24=[0 x23];
x13=[x12 0];
xu=x13+x24;
[Xu,fu]=freqz(xu,1,256,2000);
subplot(211), plot(f,abs(X)), xlabel('frekvencija [Hz]'),
title('spektar ulaznog signala x(n)')
subplot(212), plot(fu,abs(Xu)), xlabel('frekvencija [Hz]'),
title('x(n) posle ekvivalentnog decimatora')

n=0:255;
x=cos(2*pi*n*100/2000)+0.1*sqrt(randn(size(n)));
```

```
[X,f]=freqz(x,1,256,2000);
subplot(211), plot(f,abs(X)), xlabel('frekvencija [Hz]'),
title('spektar ulaznog signala x(n)')
x1=downsample(x,2);
x12=upsample(x1,10);
x2=x(2:256);
x22=downsample(x2,2);
x23=upsample(x22,10);
x24=[0 x23];
x13=[x12 0];
xu=x13+x24;
[Xu,fu]=freqz(xu,1,256,2000);
subplot(212), plot(fu,abs(Xu)), xlabel('frekvencija [Hz]'),
title('spektar signala x(n) na izlazu iz ekvivalentnog
interpolatora')

[N,fp,mag,wt]=remezord([50 100],[1 0],[0.01 0.001],2000);
b=remez(N,fp,mag,wt);
[H,f]=freqz(b,1,256,2000);
plot(f,20*log10(abs(H)));grid
xlabel('frekvencija [Hz]'), ylabel('amplitudska karakteristika
[dB]')


[Nk,fpk,magk,wtk]=remezord([50 100],[1 0],[0.01/3 0.001],400);
bk=remez(Nk,fpk,magk,wtk);
[Hk,fk]=freqz(bk,1,256,400);
plot(fk,20*log10(abs(Hk))), title('kernel filter'),grid
xlabel('frekvencija [Hz]'), ylabel('amplitudska karakteristika
[dB]')


% interpolacioni i decimacioni filter
[Ni,fpi,magi,wti]=remezord([50 300],[1 0],[0.01/3 0.01],2000);
bi=remez(Ni,fpi,magi,wti);
[Hi,fi]=freqz(bi,1,256,2000);
plot(fi,20*log10(abs(Hi))), title('decimacioni i interpolacioni
filter'),grid
xlabel('frekvencija [Hz]'), ylabel('amplitudska karakteristika
[dB]')

%multistage filter
biu=conv(bi,bi);
bku=upsample(bk,5);
bu=conv(biu,bku);
[Hu,fu]=freqz(bu,1,256,2000);
plot(fu,20*log10(abs(Hu))), title('visestepeni filter'),grid
xlabel('frekvencija [Hz]'), ylabel('amplitudska karakteristika
[dB]')
axis([0 1000 -120 20])
```

```
clear all
%definisanje filtara (decimacionih, interpolacionih i kernel
filtra)
%filtri Hd1 i Hi3
[Nd1,fpd1,magd1,wtd1]=remezord([50 1200],[1 0],[0.01/3
0.001],3200);
bkd1=remez(Nd1,fpd1,magd1,wtd1);
%filtri Hd2 i Hi2
[Nd2,fpd2,magd2,wtd2]=remezord([50 600],[1 0],[0.01/3
0.001],1600);
bkd2=remez(Nd2,fpd2,magd2,wtd2);
%filtri Hd3 i Hi1
[Nd3,fpd3,magd3,wtd3]=remezord([50 300],[1 0],[0.01/3
0.001],800);
bkd3=remez(Nd3,fpd3,magd3,wtd3);
%kernel filter
[Nk,fpk,magk,wtk]=remezord([50 100],[1 0],[0.01/3 0.001],400);
bk=remez(Nk,fpk,magk,wtk);
%prosirivanje filtara
bkd1u=zeros(1,2*length(bkd1)-1);
bkd1u(1:2:length(bkd1u))=bkd1;
bkd2u=zeros(1,2*length(bkd2)-1);
bkd2u(1:2:length(bkd2u))=bkd2;
bkd3u=zeros(1,2*length(bkd3)-1);
bkd3u(1:2:length(bkd3u))=bkd3;
bku=zeros(1,2*length(bk)-1);
bku(1:2:length(bku))=bk;
%snimanje karakteristike strukture
for k=1:127
n=0:1023;
N=1024;
x1=sin(2*pi*n*k/256);
x2=zeros(1,2*length(x1));
x2(1:2:length(x2))=x1;
x3=filter(bkd1u,1,x2);
x4=zeros(1,1024);
x4(1:2:length(x4))=x3(1:4:length(x3));
x4(2:2:length(x4))=x3(2:4:length(x3));
x5=filter(bkd2u,1,x4);
x6=zeros(1,512);
x6(1:2:length(x6))=x5(1:4:length(x5));
x6(2:2:length(x6))=x5(2:4:length(x5));
x7=filter(bkd3u,1,x6);
x8=zeros(1,256);
x8(1:2:length(x8))=x7(1:4:length(x7));
x8(2:2:length(x8))=x7(2:4:length(x7));
x9=filter(bku,1,x8);
x10=zeros(1,512);
x10(1:4:length(x10))=x9(1:2:length(x9));
x10(2:4:length(x10))=x9(2:2:length(x9));
x11=filter(4*bkd3u,1,x10);
x12=zeros(1,1024);
```

```
x12(1:4:length(x12))=x11(1:2:length(x11));
x12(2:4:length(x12))=x11(2:2:length(x11));
x13=filter(2*bkd2u,1,x12);
x14=zeros(1,2048);
x14(1:4:length(x14))=x13(1:2:length(x13));
x14(2:4:length(x14))=x13(2:2:length(x13));
x15=filter(2*bkd1u,1,x14);
x16=zeros(1,1024);
x16(1:length(x16))=x15(1:2:length(x15));
Y16=fft(x16,256);
Xa=fft(x1,256);
% Sad formiramo matricu reda 127*256 (127 frekvencija, a 256-
broj tacaka FFT)
for d=1:length(Y16)
Y(k,d)=Y16(1,d);
X(k,d)=Xa(1,d);
end
end
% Za svaku frekvenciju racunamo izlaz na toj frekvenciji
for r=1:127
Yi(1,r)=Y(r,r+1);
Xu(1,r)=X(r,r+1);
end
j=1:127;
% Karakteristika racunata odbirak po odbirak
plot(j*1800/127,20*log10(abs(Yi)./abs(Xu)))
axis([0 1800 -150 0])
grid, xlabel('frekvencija [Hz]'), ylabel('amplitudska
karakteristika [dB]');
```

10.4 ФРЕКВЕНЦИЈСКО МАСКИРАЊЕ

```
clear all
[Nk,fpk,magk,wtk]=remezord([250 500],[1 0],[0.01 0.001],2000);
bk=remez(Nk,fpk,magk,wtk);
[Hk,fk]=freqz(bk,1,256,2000);
plot(fk,20*log10(abs(Hk))),title('model filter')
xlabel('frekvencija [Hz]'), ylabel('amplitudska karakteristika
[dB]');
bpm=upsample(bk,5);
[Hpm,fpm]=freqz(bpm,1,256,2000);
plot(fpm,20*log10(abs(Hpm))),title('periodicni model filter')
xlabel('frekvencija [Hz]'), ylabel('amplitudska karakteristika
[dB]');
[Nm,fpm,magm,wtm]=remezord([50 300],[1 0],[0.01 0.001],2000);
bm=remez(Nm,fpm,magm,wtm);
[Hm,fm]=freqz(bm,1,256,2000);
plot(fm,20*log10(abs(Hm))),title('maskirajuci filter')
```

```
xlabel('frekvencija [Hz]'), ylabel('amplitudska karakteristika
[dB]');
bu=conv(bm,bpm);
[Hu,fu]=freqz(bu,1,256,2000);
plot(fu,20*log10(abs(Hu))), xlabel('frekvencija [Hz]'),
ylabel('amplitudska karakteristika [dB]');



clear all
[Nk,fpk,magk,wtk]=remezord([250 500],[1 0],[0.01 0.001],2000);
bk=remez(Nk,fpk,magk,wtk);
bpm=upsample(bk,5);
 [Nm,fpm,magm,wtm]=remezord([50 300],[1 0],[0.01 0.001],2000);
bm=remez(Nm,fpm,magm,wtm);
bu=conv(bpm,bm);
b=upsample(bu,2);
% Formiramo ulazni signal frekvencije 1:127
n=0:1023;
N=1024;
R=length(b);
% R-dužina filtra, potrebna nam je zbog izracunavanja
konvolucije
xb=zeros(1,N);
for k=1:127
xa=sin(2*pi*n*k/256);
% Kako ce zbog kola za kasnjenje i zbog toga sto izlaz iz filtra
izracunavamo na osnovu konvolucije izlazni signal biti duzi ud
ulaznog, analizu odbiraka moramo nastaviti do 2*N+R
for m=1:2*N+R
if m<=2*N
% Podizanje ucestanosti odabiranja za faktor L=2
if ((-1).^m)+1==0
y1(1,m)=xa(1,m-(m-1)/2);
y3(1,m)=xb(1,m-(m-1)/2);
else
y1(1,m)=0;
y3(1,m)=0;
end
% Kolo za kasnjenje
if m==1
y2(1,m)=0;
else
y2(1,m)=y3(1,m-1);
end
% Sabirac
y4(1,m)=y1(1,m)+y3(1,m);
% Filter, racunanje izlaza odbirak po odbirak
x=y4(1:m);
y5=conv(x,b);
y6(1,m)=y5(1,m);
else
```

```
% Nema signala na ulazu filtra-istekao je y4(n), i sada filter
generise odbirke u zavisnosti od reda filtra i prethodnih ulaza
if m<=2*N+R-1
y6(1,m)=y5(1,m);
else
end
end
% Smanjivanje ucestanosti odabiranja za faktor M=2
if m<=2*N+R-1
if ((-1).^m)+1==0
y7(1,m-(m-1)/2)=y6(1,m);
else
end
else
end
if m==1
% Kolo za kasnjenje
y8(1,m)=0;
else
y8(1,m)=y6(1,m-1);
end
% Smanjivanje ucestanosti odabiranja za faktor M=2
if ((-1).^m)+1==0
y9(1,m-(m-1)/2)=y8(1,m);
else
end
end
y7e=y7(256:768);
Y7e=fft(y7e,256);
Xa=fft(xa,256);
% Sad formiramo matricu reda 127*256 (127 frekvencija, a 256-
broj tacaka FFT)
for d=1:length(Y7e)
Y(k,d)=Y7e(1,d);
X(k,d)=Xa(1,d);
end
% Matrica izlaza u vremenskom domenu
for t=1:length(y7e)
yi(k,t)=y7e(1,t);
end
end
% Za svaku frekvenciju racunamo izlaz na toj frekvenciji
for r=1:127
Yi(1,r)=Y(r,r+1);
Xu(1,r)=X(r,r+1);
end
j=1:127;
% Karakteristika racunata odbirak po odbirak
plot(j*1000/127,20*log10(abs(Yi)./abs(Xu))),xlabel('frekvencija
[Hz]'), ylabel('pojacanje [dB]')
axis([0 1000 -200 50])
for q=1:127
```

```
xa=sin(2*pi*n*q/256);
yai1=conv(xa,bu);
yai3=yai1(1:(length(yai1)-4));
yai3e=yai3(256:768);
razlika=yai3e-yi(q,:);
greska(1,q)=max(razlika);
end
maxgreska=max(greska)
```

10.5 *QMF* БАНКЕ СА СТРУКТУРОМ СТАБЛА

```
clear all
as = 40;
delta = 10^(-as/20);
fs = 0.60;
fp = 1-fs;
h0= firhalfband('minorder',fp,delta);
h1= firhalfband('minorder',fp,delta,'high');
% prosirivanje filtara
for t=1:2*length(h0)-1
if ((-1).^t)+1==0
h02(1,t)=h0(1,t-(t-1)/2);
else
h02(1,t)=0;
end
end
for t=1:2*length(h1)-1
if ((-1).^t)+1==0
h12(1,t)=h1(1,t-(t-1)/2);
else
h12(1,t)=0;
end
end
[H0,w0]=freqz(h0,1,256);
[H1,w1]=freqz(h1,1,256);
plot(w0,20*log10(abs(H0)),'r',w1,20*log10(abs(H1)),'g')
xlabel('frekvencija [rad]'), ylabel('pojacanje [dB]'),
title('karakteristike filtara H0(z) i H1(z)')
n=0:1023;
Nx=1024;
%računanje izlaza iz originalne strukture
for k=1:511
x1=sin(2*pi*n*k/1024);
for m=1:Nx+length(h0)-1+length(h02)-1
if m<= Nx+length(h0)-1
if m<=Nx
x=x1(1:m);
y1=conv(x,h0);
```

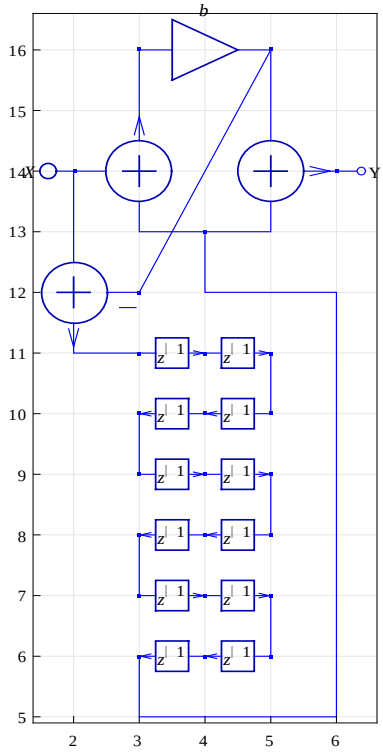
```
y2(1,m)=y1(1,m);
y3=conv(x,h1);
y4(1,m)=y3(1,m);
else
y2(1,m)=y1(1,m);
y4(1,m)=y3(1,m);
end
y5=zeros(1,length(y2));
y5(1:2:length(y5))=y2(1:2:length(y2));
y5(2:2:length(y5))=y4(1:2:length(y2)-1);
else
end
if m<=Nx+length(h0)-1
y6=conv(y5,h02);
y8=conv(y5,h12);
y7(1,m)=y6(1,m);
y9(1,m)=y8(1,m);
else
y7(1,m)=y6(1,m);
y9(1,m)=y8(1,m);
end
y11=downsample(y7,2);
y13=downsample(y11,2);
end
% Matrica izlaza u vremenskom domenu
for t=1:256
yi(k,t)=y13(1,t);
end
end
xa10=sin(2*pi*n*10/1024);
xa150=sin(2*pi*n*150/1024);
xa130=sin(2*pi*n*130/1024);
xa400=sin(2*pi*n*400/1024);
z110=filter(h0,1,xa10);
z1150=filter(h0,1,xa150);
z1130=filter(h0,1,xa130);
z1400=filter(h0,1,xa400);
z210=downsample(z110,2);
z2150=downsample(z1150,2);
z2130=downsample(z1130,2);
z2400=downsample(z1400,2);
z310=filter(h0,1,z210);
z3150=filter(h0,1,z2150);
z3130=filter(h0,1,z2130);
z3400=filter(h0,1,z2400);
z410=downsample(z310,2);
z4150=downsample(z3150,2);
z4130=downsample(z3130,2);
z4400=downsample(z3400,2);
yi10=yi(10,:);
yi150=yi(150,:);
yi130=yi(130,:);
```

```
yi400=yi(400,:);
yi10e=yi10(32:256);
yi150e=yi150(32:256);
yi130e=yi130(32:256);
yi400e=yi400(32:256);
z410e=z410(32:256);
z4150e=z4150(32:256);
z4130e=z4130(32:256);
z4400e=z4400(32:256);
c=1:length(yi10e);
v=1:length(z410e);
figure (2)
subplot(411), plot(c,yi10e,'r.',v,z410e,'g'),
title('x=sin(2*pi*n*10/1024)')
subplot(412), plot(c,yi130e,'r.',v,z4130e,'g'),
title('x=sin(2*pi*n*130/1024)')
subplot(413), plot(c,yi150e,'r.',v,z4150e,'g'),
title('x=sin(2*pi*n*150/1024)')
subplot(414), plot(c,yi400e,'r.',v,z4400e,'g'),
title('x=sin(2*pi*n*400/1024)')
razlika10=yi10e-z410e;
razlika130=yi130e-z4130e;
razlika150=yi150e-z4150e;
razlika400=yi400e-z4400e;
greska10=max(razlika10);
greska130=max(razlika130);
greska150=max(razlika150);
greska400=max(razlika400);
greska=[greska10 greska130 greska150 greska400];
maxgreska=max(greska)
```

10.6 ДИГИТАЛНИ ФИЛТРИ ЗА ВЕЛИКЕ БРЗИНЕ ОБРАДЕ

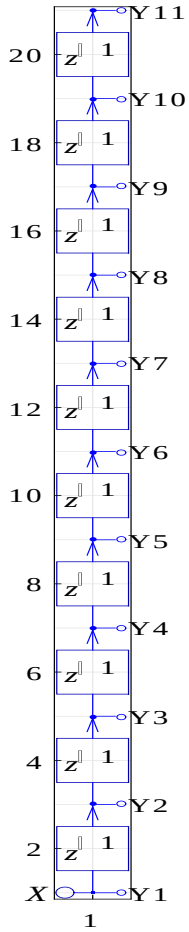
```
Needs["SchematicSolver`"];
 smenak2a={a0→k0 k3,a1→-k0 k1 k3,a2→-k0 k2 (-1+k1 k3),a3→-k0
k2 (k1+k3),a4→k0 k1,a5→k0};
 parameterSubstitution={
   b→1/2+1/16,
   k0→0.24000685,
   k1→2.37428361,
   k2→-0.54068333,
   k3→ 0.10932683};
 smenak2a/.parameterSubstitution
 {a0→0.0262392,a1→-0.0622993,a2→-
0.0960835,a3→0.322292,a4→0.569844,a5→0.240007}

h1Schematic={{"Input",{2,14},X},{"Output",{6,14},"Y"},{"Adder",{
```

```
{1,12},{3, 11},{3, 12},{2, 14}},{0,2,-
1,1}},{"Adder",{{2,14},{4,13},{4,14},{3,16}},{1,1,0,2}},{"Adder"
,{{4,14},{4,13},{6,14},{5,16}},{0,1,2,1}},{"Multiplier",{{3,16},
{5,16}},b},
   {"Delay", {{3, 11}, {4, 11}}, 1, ""},
   {"Delay", {{4, 11}, {5, 11}}, 1, ""},
   {"Delay", {{5, 10}, {4, 10}}, 1, ""},
   {"Delay", {{4, 10}, {3, 10}}, 1, ""},
   {"Delay", {{3, 9}, {4, 9}}, 1, ""},
   {"Delay", {{4, 9}, {5, 9}}, 1, ""},
   {"Delay", {{5, 8}, {4,8}}, 1, ""},
   {"Delay", {{4, 8}, {3,8}}, 1, ""},
   {"Delay", {{3, 7}, {4,7}}, 1, ""},
   {"Delay", {{4, 7}, {5,7}}, 1, ""},
   {"Delay", {{5, 6}, {4, 6}}, 1, ""},
   {"Delay", {{4, 6}, {3, 6}}, 1, ""},
   {"Line",{{3,12},{5,16}}},
   {"Line", {{5, 11}, {5,10}}},
   {"Line", {{3, 9}, {3, 10}}},
   {"Line", {{5, 8}, {5,9}}},
   {"Line", {{3, 7}, {3, 8}}},
   {"Line", {{5, 7}, {5, 6}}},
   {"Line", {{3, 6},{3, 5},{6, 5}, {6, 12},{4, 12}, {4, 13}}}};
 ShowSchematic[h1Schematic];
```




```
inputSequence={{X1},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},
{0},{0},{0},{0},{0},{0}};
procedureName = imp1;
DiscreteSystemImplementation[h1Schematic,
ToString[procedureName]];
initialConditions = {0,0,0,0,0,0,0,0,0,0,0,0};
```

```
systemParameters = {b};
{outputSequence, finalConditions} =
DiscreteSystemImplementationProcessing[inputSequence,
initialConditions, systemParameters, procedureName];
outputSequence//FullSimplify
 Implementation procedure name:  imp1
 Implementation procedure usage:
  {{Y6p14}, {Y3p11, Y4p11, Y5p11, Y4p10, Y3p9, Y4p9, Y5p8, Y4p8, Y3p7, Y4p7, Y5p7,
Y4p6}} = imp1[{Y2p14},{Y4p11, Y5p11, Y4p10, Y3p9, Y4p9, Y5p8, Y4p8, Y3p7, Y4p7, Y5p7,
Y4p6, Y3p6},{b}] is the template for calling the procedure.  The general template is
{outputSamples, finalConditions} = procedureName[inputSamples, initialConditions,
systemParameters]. See also: DiscreteSystemImplementationProcessing
 {{b X1},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{X1-b^2
X1},{0},{0},{0},{0},{0},{0}}
 mySchematic = {
    {"Delay", {{1, 1}, {1, 3}}, 1, ""},{"Delay", {{1, 3}, {1,
5}}, 1, ""},
   {"Delay", {{1, 5}, {1, 7}}, 1, ""},{"Delay", {{1, 7}, {1,
9}}, 1, ""},
   {"Delay", {{1, 9}, {1, 11}}, 1, ""},{"Delay", {{1, 11}, {1,
13}}, 1, ""},
   {"Delay", {{1, 13}, {1, 15}}, 1, ""},{"Delay", {{1, 15}, {1,
17}}, 1, ""},
   {"Delay", {{1, 17}, {1, 19}}, 1, ""},{"Delay", {{1, 19}, {1,
21}}, 1, ""},

   {"Output", {1, 1}, Y1, "", TextOffset →{-1, 0}},
   {"Output", {1, 3}, Y2, "", TextOffset →{-1, 0}},
   {"Output", {1, 5}, Y3, "", TextOffset →{-1, 0}},
   {"Output", {1, 7}, Y4, "", TextOffset →{-1, 0}},
   {"Output", {1, 9}, Y5, "", TextOffset →{-1, 0}},
   {"Output", {1, 11}, Y6, "", TextOffset →{-1, 0}},
   {"Output", {1, 13}, Y7, "", TextOffset →{-1, 0}},
   {"Output", {1, 15}, Y8, "", TextOffset →{-1, 0}},
   {"Output", {1, 17}, Y9, "", TextOffset →{-1, 0}},
   {"Output", {1, 19}, Y10, "", TextOffset →{-1, 0}},
   {"Output", {1, 21}, Y11, "", TextOffset →{-1, 0}},

   {"Input", {1, 1}, X, "", TextOffset → {1, 0}} };
ShowSchematic[%];
```




```
 procName = impD;
DiscreteSystemImplementation[mySchematic, ToString[procName]];
 Implementation procedure name:  impD
 Implementation procedure usage:
```

{{Y1p1, Y1p3, Y1p5, Y1p7, Y1p9, Y1p11, Y1p13, Y1p15, Y1p17, Y1p19, Y1p21}, {Y1p1, Y1p3, Y1p5, Y1p7, Y1p9, Y1p11, Y1p13, Y1p15, Y1p17, Y1p19}} = impD[{Y1p1},{Y1p3, Y1p5, Y1p7, Y1p9, Y1p11, Y1p13, Y1p15, Y1p17, Y1p19, Y1p21},{}] is the template for calling the procedure. The general template is {outputSamples, finalConditions} = procedureName[inputSamples, initialConditions, systemParameters]. See also: DiscreteSystemImplementationProcessing

```
 UnitSymbolicSequence[20,r]

{{r1},{r2},{r3},{r4},{r5},{r6},{r7},{r8},{r9},{r10},{r11},{r12},
{r13},{r14},{r15},{r16},{r17},{r18},{r19},{r20}}
 seq1 = ListToSequence [DiscreteSystemImplementationProcessing[
UpsampleSequence[ UnitSymbolicSequence[20,r],6 ],
{0,0,0,0,0,0,0,0,0,0}, {}, procName][[1]][[All,1]]]

{{r1},{0},{0},{0},{0},{0},{r2},{0},{0},{0},{0},{0},{r3},{0},{0},
{0},{0},{0},{r4},{0},{0},{0},{0},{0},{r5},{0},{0},{0},{0},{0},{r
6},{0},{0},{0},{0},{0},{r7},{0},{0},{0},{0},{0},{r8},{0},{0},{0}
,{0},{0},{r9},{0},{0},{0},{0},{0},{r10},{0},{0},{0},{0},{0},{r11
},{0},{0},{0},{0},{0},{r12},{0},{0},{0},{0},{0},{r13},{0},{0},{0
},{0},{0},{r14},{0},{0},{0},{0},{0},{r15},{0},{0},{0},{0},{0},{r
16},{0},{0},{0},{0},{0},{r17},{0},{0},{0},{0},{0},{r18},{0},{0},
{0},{0},{0},{r19},{0},{0},{0},{0},{0},{r20},{0},{0},{0},{0},{0}}
```

```
x UnitImpulseSequence[20]
```

```
{{x},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0}
,{0},{0},{0},{0}}
```

```
 seq0   =x  UpsampleSequence[ UnitImpulseSequence[20],6 ]
```

```
{{x},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0}
,{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0}
,{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0}
,{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0}
,{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0}
,{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0}
,{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0}
,{0},{0},{0},{0},{0},{0},{0},{0}}
```

```
seq2 = ListToSequence [DiscreteSystemImplementationProcessing[
UpsampleSequence[ UnitSymbolicSequence[20,p],6 ],
{0,0,0,0,0,0,0,0,0,0}, {}, procName][[1]][[All,2]]]
```

```
{{0},{p1},{0},{0},{0},{0},{0},{p2},{0},{0},{0},{0},{0},{p3},{0},
{0},{0},{0},{0},{p4},{0},{0},{0},{0},{0},{p5},{0},{0},{0},{0},{0
},{p6},{0},{0},{0},{0},{0},{p7},{0},{0},{0},{0},{0},{p8},{0},{0}
,{0},{0},{0},{p9},{0},{0},{0},{0},{0},{p10},{0},{0},{0},{0},{0},
{p11},{0},{0},{0},{0},{0},{p12},{0},{0},{0},{0},{0},{p13},{0},{0
},{0},{0},{0},{p14},{0},{0},{0},{0},{0},{p15},{0},{0},{0},{0},{0
},{p16},{0},{0},{0},{0},{0},{p17},{0},{0},{0},{0},{0},{p18},{0},
{0},{0},{0},{0},{p19},{0},{0},{0},{0},{0},{p20},{0},{0},{0},{0}}
```

```
seq3 = ListToSequence [DiscreteSystemImplementationProcessing[
UpsampleSequence[ UnitSymbolicSequence[20,q],6 ],
{0,0,0,0,0,0,0,0,0,0}, {}, procName][[1]][[All,3]]]
```

```
{{0},{0},{q1},{0},{0},{0},{0},{0},{q2},{0},{0},{0},{0},{0},{q3},
{0},{0},{0},{0},{0},{q4},{0},{0},{0},{0},{0},{q5},{0},{0},{0},{0
},{0},{q6},{0},{0},{0},{0},{0},{q7},{0},{0},{0},{0},{0},{q8},{0}
,{0},{0},{0},{0},{q9},{0},{0},{0},{0},{0},{q10},{0},{0},{0},{0},
{0},{q11},{0},{0},{0},{0},{0},{q12},{0},{0},{0},{0},{0},{q13},{0
},{0},{0},{0},{0},{q14},{0},{0},{0},{0},{0},{q15},{0},{0},{0},{0
},{0},{q16},{0},{0},{0},{0},{0},{q17},{0},{0},{0},{0},{0},{q18},
{0},{0},{0},{0},{0},{q19},{0},{0},{0},{0},{0},{q20},{0},{0},{0}}
```

```
seq4= ListToSequence [DiscreteSystemImplementationProcessing[
UpsampleSequence[ UnitSymbolicSequence[20,w],6 ],
{0,0,0,0,0,0,0,0,0,0}, {}, procName][[1]][[All,4]]]
```

```
{{0},{0},{0},{w1},{0},{0},{0},{0},{0},{w2},{0},{0},{0},{0},{0},{
w3},{0},{0},{0},{0},{0},{w4},{0},{0},{0},{0},{0},{w5},{0},{0},{0
},{0},{0},{w6},{0},{0},{0},{0},{0},{w7},{0},{0},{0},{0},{0},{w8}
,{0},{0},{0},{0},{0},{w9},{0},{0},{0},{0},{0},{w10},{0},{0},{0},
{0},{0},{w11},{0},{0},{0},{0},{0},{w12},{0},{0},{0},{0},{0},{w13
},{0},{0},{0},{0},{0},{w14},{0},{0},{0},{0},{0},{w15},{0},{0},{0
},{0},{0},{w16},{0},{0},{0},{0},{0},{w17},{0},{0},{0},{0},{0},{w
18},{0},{0},{0},{0},{0},{w19},{0},{0},{0},{0},{0},{w20},{0},{0}}
```

```
 seq5 = ListToSequence [DiscreteSystemImplementationProcessing[
UpsampleSequence[ UnitSymbolicSequence[20,v],6 ],
{0,0,0,0,0,0,0,0,0,0}, {}, procName][[1]][[All,5]]]
```

```
{{0},{0},{0},{0},{v1},{0},{0},{0},{0},{0},{v2},{0},{0},{0},{0},{
0},{v3},{0},{0},{0},{0},{0},{v4},{0},{0},{0},{0},{0},{v5},{0},{0
},{0},{0},{0},{v6},{0},{0},{0},{0},{0},{v7},{0},{0},{0},{0},{0},
{v8},{0},{0},{0},{0},{0},{v9},{0},{0},{0},{0},{0},{v10},{0},{0},
{0},{0},{0},{v11},{0},{0},{0},{0},{0},{v12},{0},{0},{0},{0},{0},
{v13},{0},{0},{0},{0},{0},{v14},{0},{0},{0},{0},{0},{v15},{0},{0
},{0},{0},{0},{v16},{0},{0},{0},{0},{0},{v17},{0},{0},{0},{0},{0
},{v18},{0},{0},{0},{0},{0},{v19},{0},{0},{0},{0},{0},{v20},{0}}
```

```
 seq6 = ListToSequence [DiscreteSystemImplementationProcessing[
UpsampleSequence[ UnitSymbolicSequence[20,u],6 ],
{0,0,0,0,0,0,0,0,0,0}, {}, procName][[1]][[All,6]]]
```

```
{{0},{0},{0},{0},{0},{u1},{0},{0},{0},{0},{0},{u2},{0},{0},{0},{
0},{0},{u3},{0},{0},{0},{0},{0},{u4},{0},{0},{0},{0},{0},{u5},{0
},{0},{0},{0},{0},{u6},{0},{0},{0},{0},{0},{u7},{0},{0},{0},{0},
{0},{u8},{0},{0},{0},{0},{0},{u9},{0},{0},{0},{0},{0},{u10},{0},
{0},{0},{0},{0},{u11},{0},{0},{0},{0},{0},{u12},{0},{0},{0},{0},
{0},{u13},{0},{0},{0},{0},{0},{u14},{0},{0},{0},{0},{0},{u15},{0
},{0},{0},{0},{0},{u16},{0},{0},{0},{0},{0},{u17},{0},{0},{0},{0
},{0},{u18},{0},{0},{0},{0},{0},{u19},{0},{0},{0},{0},{0},{u20}}
```

```
 inps=seq0+seq2+seq3+seq4+seq5+seq6
```

```
{{x},{p1},{q1},{w1},{v1},{u1},{0},{p2},{q2},{w2},{v2},{u2},{0},{
p3},{q3},{w3},{v3},{u3},{0},{p4},{q4},{w4},{v4},{u4},{0},{p5},{q
5},{w5},{v5},{u5},{0},{p6},{q6},{w6},{v6},{u6},{0},{p7},{q7},{w7
},{v7},{u7},{0},{p8},{q8},{w8},{v8},{u8},{0},{p9},{q9},{w9},{v9}
,{u9},{0},{p10},{q10},{w10},{v10},{u10},{0},{p11},{q11},{w11},{v
11},{u11},{0},{p12},{q12},{w12},{v12},{u12},{0},{p13},{q13},{w13
},{v13},{u13},{0},{p14},{q14},{w14},{v14},{u14},{0},{p15},{q15},
{w15},{v15},{u15},{0},{p16},{q16},{w16},{v16},{u16},{0},{p17},{q
17},{w17},{v17},{u17},{0},{p18},{q18},{w18},{v18},{u18},{0},{p19
},{q19},{w19},{v19},{u19},{0},{p20},{q20},{w20},{v20},{u20}}
```

```
 {outputSequence, finalConditions} =
DiscreteSystemImplementationProcessing[inps,
{0,0,0,0,0,0,0,0,0,0,0,0,0}, {b}, imp1];
  out0 = DownsampleSequence [outputSequence,6]//Expand
```

$$\{\{b\ x\},\{0\},\{x-b^2\ x\},\{0\},\{-b\ x+b^3\ x\},\{0\},\{b^2\ x-b^4\ x\},\{0\},\{-b^3\ x+b^5\ x\},\{0\},\{b^4\ x-b^6\ x\},\{0\},\{-b^5\ x+b^7\ x\},\{0\},\{b^6\ x-b^8\ x\},\{0\},\{-b^7\ x+b^9\ x\},\{0\},\{b^8\ x-b^{10}\ x\},\{0\}\}$$

```
  out0//InputForm
 {{b*x}, {0}, {x - b^2*x}, {0}, {-(b*x) + b^3*x}, {0}, {b^2*x -
b^4*x}, {0}, {-(b^3*x) + b^5*x}, {0}, {b^4*x - b^6*x}, {0}, {-
(b^5*x) + b^7*x}, {0}, {b^6*x - b^8*x}, {0},
 {-(b^7*x) + b^9*x}, {0}, {b^8*x - b^10*x}, {0}}
  out1 = DownsampleSequence [outputSequence,6,1]//Expand
```

$$\{\{b\ p1\},\{b\ p2\},\{p1-b^2\ p1+b\ p3\},\{p2-b^2\ p2+b\ p4\},\{-b\ p1+b^3\ p1+p3-b^2\ p3+b\ p5\},\{-b\ p2+b^3\ p2+p4-b^2\ p4+b\ p6\},\{b^2\ p1-b^4\ p1-b\ p3+b^3\ p3+p5-b^2\ p5+b\ p7\},\{b^2\ p2-b^4\ p2-b\ p4+b^3\ p4+p6-b^2\ p6+b\ p8\},\{-b^3\ p1+b^5\ p1+b^2$$

```
p3-b^4 p3-b p5+b^3 p5+p7-b^2 p7+b p9},{b p10-b^3 p2+b^5 p2+b^2 p4-b^4 p4-b
p6+b^3 p6+p8-b^2 p8},{b^4 p1-b^6 p1+b p11-b^3 p3+b^5 p3+b^2 p5-b^4 p5-b
p7+b^3 p7+p9-b^2 p9},{p10-b^2 p10+b p12+b^4 p2-b^6 p2-b^3 p4+b^5 p4+b^2 p6-
b^4 p6-b p8+b^3 p8},{-b^5 p1+b^7 p1+p11-b^2 p11+b p13+b^4 p3-b^6 p3-b^3
p5+b^5 p5+b^2 p7-b^4 p7-b p9+b^3 p9},{-b p10+b^3 p10+p12-b^2 p12+b p14-b^5
p2+b^7 p2+b^4 p4-b^6 p4-b^3 p6+b^5 p6+b^2 p8-b^4 p8},{b^6 p1-b^8 p1-b p11+b^3
p11+p13-b^2 p13+b p15-b^5 p3+b^7 p3+b^4 p5-b^6 p5-b^3 p7+b^5 p7+b^2 p9-b^4
p9},{b^2 p10-b^4 p10-b p12+b^3 p12+p14-b^2 p14+b p16+b^6 p2-b^8 p2-b^5
p4+b^7 p4+b^4 p6-b^6 p6-b^3 p8+b^5 p8},{-b^7 p1+b^9 p1+b^2 p11-b^4 p11-b
p13+b^3 p13+p15-b^2 p15+b p17+b^6 p3-b^8 p3-b^5 p5+b^7 p5+b^4 p7-b^6 p7-b^3
p9+b^5 p9},{-b^3 p10+b^5 p10+b^2 p12-b^4 p12-b p14+b^3 p14+p16-b^2 p16+b
p18-b^7 p2+b^9 p2+b^6 p4-b^8 p4-b^5 p6+b^7 p6+b^4 p8-b^6 p8},{b^8 p1-b^10 p1-
b^3 p11+b^5 p11+b^2 p13-b^4 p13-b p15+b^3 p15+p17-b^2 p17+b p19-b^7 p3+b^9
p3+b^6 p5-b^8 p5-b^5 p7+b^7 p7+b^4 p9-b^6 p9},{b^4 p10-b^6 p10-b^3 p12+b^5
p12+b^2 p14-b^4 p14-b p16+b^3 p16+p18-b^2 p18+b^8 p2-b^10 p2+b p20-b^7
p4+b^9 p4+b^6 p6-b^8 p6-b^5 p8+b^7 p8}}
```

out1//InputForm

```
 {{b*p1}, {b*p2}, {p1 - b^2*p1 + b*p3}, {p2 - b^2*p2 + b*p4}, {-
(b*p1) + b^3*p1 + p3 - b^2*p3 + b*p5}, {-(b*p2) + b^3*p2 + p4 -
b^2*p4 + b*p6},
 {b^2*p1 - b^4*p1 - b*p3 + b^3*p3 + p5 - b^2*p5 + b*p7}, {b^2*p2
- b^4*p2 - b*p4 + b^3*p4 + p6 - b^2*p6 + b*p8},
 {-(b^3*p1) + b^5*p1 + b^2*p3 - b^4*p3 - b*p5 + b^3*p5 + p7 -
b^2*p7 + b*p9}, {b*p10 - b^3*p2 + b^5*p2 + b^2*p4 - b^4*p4 -
b*p6 + b^3*p6 + p8 - b^2*p8},
 {b^4*p1 - b^6*p1 + b*p11 - b^3*p3 + b^5*p3 + b^2*p5 - b^4*p5 -
b*p7 + b^3*p7 + p9 - b^2*p9},
 {p10 - b^2*p10 + b*p12 + b^4*p2 - b^6*p2 - b^3*p4 + b^5*p4 +
b^2*p6 - b^4*p6 - b*p8 + b^3*p8},
 {-(b^5*p1) + b^7*p1 + p11 - b^2*p11 + b*p13 + b^4*p3 - b^6*p3 -
b^3*p5 + b^5*p5 + b^2*p7 - b^4*p7 - b*p9 + b^3*p9},
 {-(b*p10) + b^3*p10 + p12 - b^2*p12 + b*p14 - b^5*p2 + b^7*p2 +
b^4*p4 - b^6*p4 - b^3*p6 + b^5*p6 + b^2*p8 - b^4*p8},
 {b^6*p1 - b^8*p1 - b*p11 + b^3*p11 + p13 - b^2*p13 + b*p15 -
b^5*p3 + b^7*p3 + b^4*p5 - b^6*p5 - b^3*p7 + b^5*p7 + b^2*p9 -
b^4*p9},
 {b^2*p10 - b^4*p10 - b*p12 + b^3*p12 + p14 - b^2*p14 + b*p16 +
b^6*p2 - b^8*p2 - b^5*p4 + b^7*p4 + b^4*p6 - b^6*p6 - b^3*p8 +
b^5*p8},
 {-(b^7*p1) + b^9*p1 + b^2*p11 - b^4*p11 - b*p13 + b^3*p13 + p15
- b^2*p15 + b*p17 + b^6*p3 - b^8*p3 - b^5*p5 + b^7*p5 + b^4*p7 -
b^6*p7 - b^3*p9 + b^5*p9},
 {-(b^3*p10) + b^5*p10 + b^2*p12 - b^4*p12 - b*p14 + b^3*p14 +
p16 - b^2*p16 + b*p18 - b^7*p2 + b^9*p2 + b^6*p4 - b^8*p4 -
b^5*p6 + b^7*p6 + b^4*p8 - b^6*p8},
 {b^8*p1 - b^10*p1 - b^3*p11 + b^5*p11 + b^2*p13 - b^4*p13 -
b*p15 + b^3*p15 + p17 - b^2*p17 + b*p19 - b^7*p3 + b^9*p3 +
b^6*p5 - b^8*p5 - b^5*p7 + b^7*p7 + b^4*p9 - b^6*p9},
 {b^4*p10 - b^6*p10 - b^3*p12 + b^5*p12 + b^2*p14 - b^4*p14 -
b*p16 + b^3*p16 + p18 - b^2*p18 + b^8*p2 - b^10*p2 + b*p20 -
b^7*p4 + b^9*p4 + b^6*p6 - b^8*p6 - b^5*p8 + b^7*p8}}
```

out2 = DownsampleSequence [outputSequence,6,2]//Expand

```
 {{b q1},{b q2},{q1-b^2 q1+b q3},{q2-b^2 q2+b q4},{-b q1+b^3 q1+q3-b^2
q3+b q5},{-b q2+b^3 q2+q4-b^2 q4+b q6},{b^2 q1-b^4 q1-b q3+b^3 q3+q5-b^2
q5+b q7},{b^2 q2-b^4 q2-b q4+b^3 q4+q6-b^2 q6+b q8},{-b^3 q1+b^5 q1+b^2
q3-b^4 q3-b q5+b^3 q5+q7-b^2 q7+b q9},{b q10-b^3 q2+b^5 q2+b^2 q4-b^4 q4-b
q6+b^3 q6+q8-b^2 q8},{b^4 q1-b^6 q1+b q11-b^3 q3+b^5 q3+b^2 q5-b^4 q5-b
q7+b^3 q7+q9-b^2 q9},{q10-b^2 q10+b q12+b^4 q2-b^6 q2-b^3 q4+b^5 q4+b^2 q6-
b^4 q6-b q8+b^3 q8},{-b^5 q1+b^7 q1+q11-b^2 q11+b q13+b^4 q3-b^6 q3-b^3
q5+b^5 q5+b^2 q7-b^4 q7-b q9+b^3 q9},{-b q10+b^3 q10+q12-b^2 q12+b q14-b^5
q2+b^7 q2+b^4 q4-b^6 q4-b^3 q6+b^5 q6+b^2 q8-b^4 q8},{b^6 q1-b^8 q1-b q11+b^3
q11+q13-b^2 q13+b q15-b^5 q3+b^7 q3+b^4 q5-b^6 q5-b^3 q7+b^5 q7+b^2 q9-b^4
q9},{b^2 q10-b^4 q10-b q12+b^3 q12+q14-b^2 q14+b q16+b^6 q2-b^8 q2-b^5
q4+b^7 q4+b^4 q6-b^6 q6-b^3 q8+b^5 q8},{-b^7 q1+b^9 q1+b^2 q11-b^4 q11-b
q13+b^3 q13+q15-b^2 q15+b q17+b^6 q3-b^8 q3-b^5 q5+b^7 q5+b^4 q7-b^6 q7-b^3
q9+b^5 q9},{-b^3 q10+b^5 q10+b^2 q12-b^4 q12-b q14+b^3 q14+q16-b^2 q16+b
q18-b^7 q2+b^9 q2+b^6 q4-b^8 q4-b^5 q6+b^7 q6+b^4 q8-b^6 q8},{b^8 q1-b^10 q1-
b^3 q11+b^5 q11+b^2 q13-b^4 q13-b q15+b^3 q15+q17-b^2 q17+b q19-b^7 q3+b^9
q3+b^6 q5-b^8 q5-b^5 q7+b^7 q7+b^4 q9-b^6 q9},{b^4 q10-b^6 q10-b^3 q12+b^5
q12+b^2 q14-b^4 q14-b q16+b^3 q16+q18-b^2 q18+b^8 q2-b^10 q2+b q20-b^7
q4+b^9 q4+b^6 q6-b^8 q6-b^5 q8+b^7 q8}}
```

out3 = DownsampleSequence [outputSequence,6,3]//Expand

```
 {{b w1},{b w2},{w1-b^2 w1+b w3},{w2-b^2 w2+b w4},{-b w1+b^3 w1+w3-b^2
w3+b w5},{-b w2+b^3 w2+w4-b^2 w4+b w6},{b^2 w1-b^4 w1-b w3+b^3 w3+w5-b^2
w5+b w7},{b^2 w2-b^4 w2-b w4+b^3 w4+w6-b^2 w6+b w8},{-b^3 w1+b^5 w1+b^2
w3-b^4 w3-b w5+b^3 w5+w7-b^2 w7+b w9},{b w10-b^3 w2+b^5 w2+b^2 w4-b^4 w4-b
w6+b^3 w6+w8-b^2 w8},{b^4 w1-b^6 w1+b w11-b^3 w3+b^5 w3+b^2 w5-b^4 w5-b
w7+b^3 w7+w9-b^2 w9},{w10-b^2 w10+b w12+b^4 w2-b^6 w2-b^3 w4+b^5 w4+b^2 w6-
b^4 w6-b w8+b^3 w8},{-b^5 w1+b^7 w1+w11-b^2 w11+b w13+b^4 w3-b^6 w3-b^3
w5+b^5 w5+b^2 w7-b^4 w7-b w9+b^3 w9},{-b w10+b^3 w10+w12-b^2 w12+b w14-b^5
w2+b^7 w2+b^4 w4-b^6 w4-b^3 w6+b^5 w6+b^2 w8-b^4 w8},{b^6 w1-b^8 w1-b w11+b^3
w11+w13-b^2 w13+b w15-b^5 w3+b^7 w3+b^4 w5-b^6 w5-b^3 w7+b^5 w7+b^2 w9-b^4
w9},{b^2 w10-b^4 w10-b w12+b^3 w12+w14-b^2 w14+b w16+b^6 w2-b^8 w2-b^5
w4+b^7 w4+b^4 w6-b^6 w6-b^3 w8+b^5 w8},{-b^7 w1+b^9 w1+b^2 w11-b^4 w11-b
w13+b^3 w13+w15-b^2 w15+b w17+b^6 w3-b^8 w3-b^5 w5+b^7 w5+b^4 w7-b^6 w7-b^3
w9+b^5 w9},{-b^3 w10+b^5 w10+b^2 w12-b^4 w12-b w14+b^3 w14+w16-b^2 w16+b
w18-b^7 w2+b^9 w2+b^6 w4-b^8 w4-b^5 w6+b^7 w6+b^4 w8-b^6 w8},{b^8 w1-b^10 w1-
b^3 w11+b^5 w11+b^2 w13-b^4 w13-b w15+b^3 w15+w17-b^2 w17+b w19-b^7 w3+b^9
w3+b^6 w5-b^8 w5-b^5 w7+b^7 w7+b^4 w9-b^6 w9},{b^4 w10-b^6 w10-b^3 w12+b^5
w12+b^2 w14-b^4 w14-b w16+b^3 w16+w18-b^2 w18+b^8 w2-b^10 w2+b w20-b^7
w4+b^9 w4+b^6 w6-b^8 w6-b^5 w8+b^7 w8}}
```

out4 = DownsampleSequence [outputSequence,6,4]//Expand

```
 {{b v1},{b v2},{v1-b^2 v1+b v3},{v2-b^2 v2+b v4},{-b v1+b^3 v1+v3-b^2
v3+b v5},{-b v2+b^3 v2+v4-b^2 v4+b v6},{b^2 v1-b^4 v1-b v3+b^3 v3+v5-b^2
v5+b v7},{b^2 v2-b^4 v2-b v4+b^3 v4+v6-b^2 v6+b v8},{-b^3 v1+b^5 v1+b^2
v3-b^4 v3-b v5+b^3 v5+v7-b^2 v7+b v9},{b v10-b^3 v2+b^5 v2+b^2 v4-b^4 v4-b
v6+b^3 v6+v8-b^2 v8},{b^4 v1-b^6 v1+b v11-b^3 v3+b^5 v3+b^2 v5-b^4 v5-b
v7+b^3 v7+v9-b^2 v9},{v10-b^2 v10+b v12+b^4 v2-b^6 v2-b^3 v4+b^5 v4+b^2 v6-
b^4 v6-b v8+b^3 v8},{-b^5 v1+b^7 v1+v11-b^2 v11+b v13+b^4 v3-b^6 v3-b^3
v5+b^5 v5+b^2 v7-b^4 v7-b v9+b^3 v9},{-b v10+b^3 v10+v12-b^2 v12+b v14-b^5
v2+b^7 v2+b^4 v4-b^6 v4-b^3 v6+b^5 v6+b^2 v8-b^4 v8},{b^6 v1-b^8 v1-b v11+b^3
v11+v13-b^2 v13+b v15-b^5 v3+b^7 v3+b^4 v5-b^6 v5-b^3 v7+b^5 v7+b^2 v9-b^4
v9},{b^2 v10-b^4 v10-b v12+b^3 v12+v14-b^2 v14+b v16+b^6 v2-b^8 v2-b^5
v4+b^7 v4+b^4 v6-b^6 v6-b^3 v8+b^5 v8},{-b^7 v1+b^9 v1+b^2 v11-b^4 v11-b
```

```
v13+b^3 v13+v15-b^2 v15+b v17+b^6 v3-b^8 v3-b^5 v5+b^7 v5+b^4 v7-b^6 v7-b^3
v9+b^5 v9},{-b^3 v10+b^5 v10+b^2 v12-b^4 v12-b v14+b^3 v14+v16-b^2 v16+b
v18-b^7 v2+b^9 v2+b^6 v4-b^8 v4-b^5 v6+b^7 v6+b^4 v8-b^6 v8},{b^8 v1-b^10 v1-
b^3 v11+b^5 v11+b^2 v13-b^4 v13-b v15+b^3 v15+v17-b^2 v17+b v19-b^7 v3+b^9
v3+b^6 v5-b^8 v5-b^5 v7+b^7 v7+b^4 v9-b^6 v9},{b^4 v10-b^6 v10-b^3 v12+b^5
v12+b^2 v14-b^4 v14-b v16+b^3 v16+v18-b^2 v18+b^8 v2-b^10 v2+b v20-b^7
v4+b^9 v4+b^6 v6-b^8 v6-b^5 v8+b^7 v8}}
```

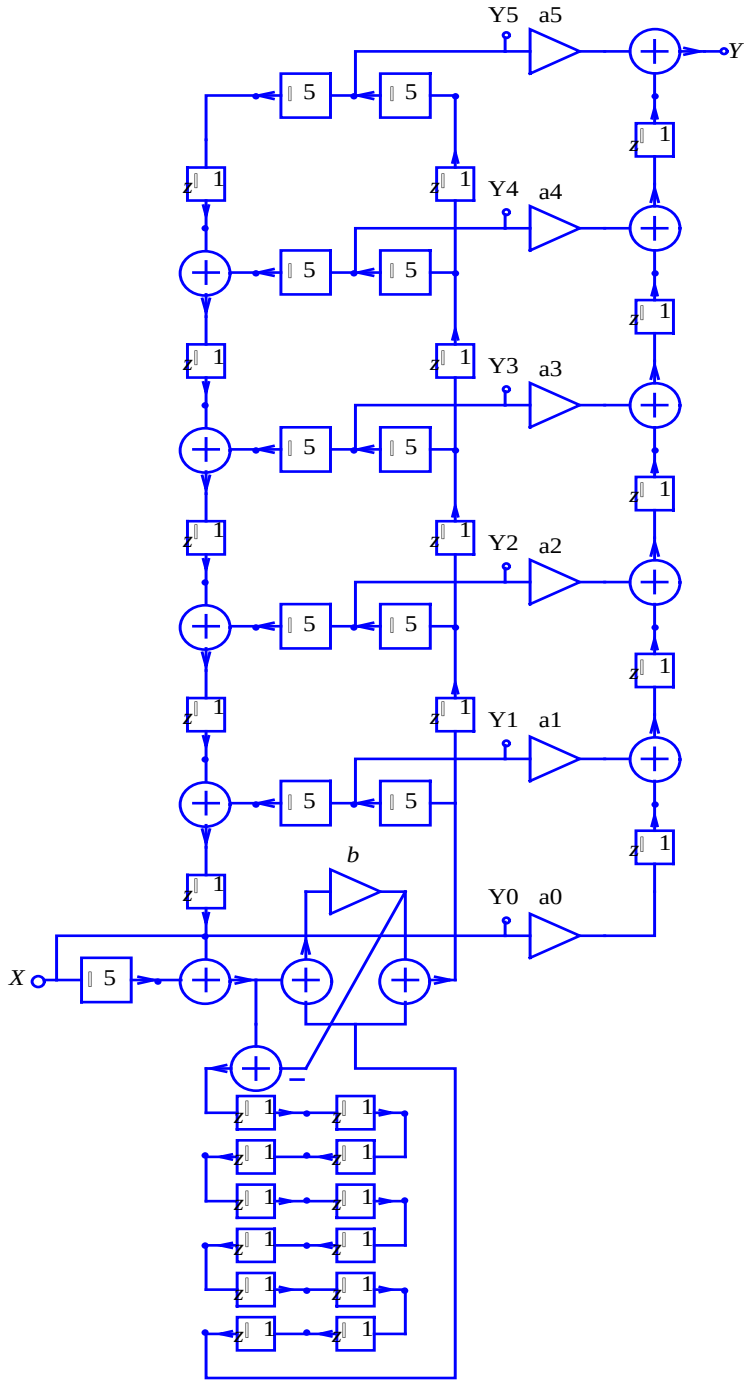
out5 = DownsampleSequence [outputSequence,6,5]//Expand

```
{{b u1},{b u2},{u1-b^2 u1+b u3},{u2-b^2 u2+b u4},{-b u1+b^3 u1+u3-b^2
u3+b u5},{-b u2+b^3 u2+u4-b^2 u4+b u6},{b^2 u1-b^4 u1-b u3+b^3 u3+u5-b^2
u5+b u7},{b^2 u2-b^4 u2-b u4+b^3 u4+u6-b^2 u6+b u8},{-b^3 u1+b^5 u1+b^2
u3-b^4 u3-b u5+b^3 u5+u7-b^2 u7+b u9},{b u10-b^3 u2+b^5 u2+b^2 u4-b^4 u4-b
u6+b^3 u6+u8-b^2 u8},{b^4 u1-b^6 u1+b u11-b^3 u3+b^5 u3+b^2 u5-b^4 u5-b
u7+b^3 u7+u9-b^2 u9},{u10-b^2 u10+b u12+b^4 u2-b^6 u2-b^3 u4+b^5 u4+b^2 u6-
b^4 u6-b u8+b^3 u8},{-b^5 u1+b^7 u1+u11-b^2 u11+b u13+b^4 u3-b^6 u3-b^3
u5+b^5 u5+b^2 u7-b^4 u7-b u9+b^3 u9},{-b u10+b^3 u10+u12-b^2 u12+b u14-b^5
u2+b^7 u2+b^4 u4-b^6 u4-b^3 u6+b^5 u6+b^2 u8-b^4 u8},{b^6 u1-b^8 u1-b u11+b^3
u11+u13-b^2 u13+b u15-b^5 u3+b^7 u3+b^4 u5-b^6 u5-b^3 u7+b^5 u7+b^2 u9-b^4
u9},{b^2 u10-b^4 u10-b u12+b^3 u12+u14-b^2 u14+b u16+b^6 u2-b^8 u2-b^5
u4+b^7 u4+b^4 u6-b^6 u6-b^3 u8+b^5 u8},{-b^7 u1+b^9 u1+b^2 u11-b^4 u11-b
u13+b^3 u13+u15-b^2 u15+b u17+b^6 u3-b^8 u3-b^5 u5+b^7 u5+b^4 u7-b^6 u7-b^3
u9+b^5 u9},{-b^3 u10+b^5 u10+b^2 u12-b^4 u12-b u14+b^3 u14+u16-b^2 u16+b
u18-b^7 u2+b^9 u2+b^6 u4-b^8 u4-b^5 u6+b^7 u6+b^4 u8-b^6 u8},{b^8 u1-b^10 u1-
b^3 u11+b^5 u11+b^2 u13-b^4 u13-b u15+b^3 u15+u17-b^2 u17+b u19-b^7 u3+b^9
u3+b^6 u5-b^8 u5-b^5 u7+b^7 u7+b^4 u9-b^6 u9},{b^4 u10-b^6 u10-b^3 u12+b^5
u12+b^2 u14-b^4 u14-b u16+b^3 u16+u18-b^2 u18+b^8 u2-b^10 u2+b u20-b^7
u4+b^9 u4+b^6 u6-b^8 u6-b^5 u8+b^7 u8}}
```

```
SetOptions[DrawElement,ShowNodes→False,PlotStyle→{{Blue,Thickne
ss[0.004]},{Blue,Thickness[0.004]}}];
 sch3={{"Input",{0,9},X},

{"Adder",{{4,9},{6,8},{6,9},{5,11}},{1,1,0,2}},{"Adder",{{6,9},{
6,8},{8,9},{7,11}},{0,1,2,1}},{"Adder", {{2, 13}, {3, 12}, {4,
13}, {3, 14}}, {0, 2,1, 1}},
   {"Adder", {{2, 17}, {3, 16}, {4, 17}, {3, 18}}, {0, 2,1, 1}},
   {"Adder", {{3, 7}, {4, 6}, {5, 7}, {4, 8}}, {2,0,-1, 1}},
   {"Adder", {{2, 9}, {3, 8}, {4, 9}, {3, 10}}, {1, 0, 2, 1}},
   {"Adder", {{2, 21}, {3, 20}, {4, 21}, {3, 22}}, {0, 2,1, 1}},
   {"Adder", {{2, 25}, {3, 24}, {4, 25}, {3, 26}}, {0, 2,1,1}, "
"},
   {"Multiplier",{{5,11},{7,11}},b},
   {"Block", {{0,9},{2, 9}},U,""},
   {"Block", {{6,13},{4,13}},U,""},{"Block",
{{8,13},{6,13}},D,""},
   {"Block", {{6,17},{4,17}},U,""},{"Block",
{{8,17},{6,17}},D,""},
   {"Block", {{6,21},{4,21}},U,""},{"Block",
{{8,21},{6,21}},D,""},
   {"Block", {{6,25},{4,25}},U,""},{"Block",
{{8,25},{6,25}},D,""},
```

```
   {"Block", {{6,29},{4,29}},U,""},{"Block",
{{8,29},{6,29}},D,""},
   {"Output",{9,10},Y0,"",TextOffset→{0,-1}},
   {"Output",{9,14},Y1,"",TextOffset→{0,-1}},
   {"Output",{9,18},Y2,"",TextOffset→{0,-1}},
   {"Output",{9,22},Y3,"",TextOffset→{0,-1}},
   {"Output",{9,26},Y4,"",TextOffset→{0,-1}},
   {"Output",{9,30},Y5,"",TextOffset→{0,-1}},
   {"Line",{{3, 1},{3,0},{8,0},{8,7},{6,7},{6,8}}},
   {"Line",{{5,7},{7,11}}},{"Line",{{7,5},{7,6}}},
   {"Line",{{3,4},{3,5}}},{"Line",{{7,3},{7,4}}},
   {"Line",{{3,2},{3,3}}},{"Line", {{8, 9}, {8, 13}}},
   {"Line", {{3, 7}, {3, 6}}},{"Line", {{6, 13}, {6, 14}, {9,
14}}},
   {"Line", {{6, 17},{6, 18}, {9, 18}}},{"Line", {{6, 21},{6,
22}, {9, 22}}},
   {"Line", {{6, 25},{6, 26}, {9, 26}}},{"Line", {{0, 9},{0,
10}, {9, 10}}},
   {"Line", {{6, 29},{6, 30}, {9, 30}}},{"Line", {{3, 28},{3,
29}, {4, 29}}},
   {"Delay",{{3,6},{5,6}},1,"",ElementSize → {0.75,0.75}},
   {"Delay",{{5,6},{7,6}},1,"",ElementSize → {0.75,0.75}},
   {"Delay",{{7,5},{5,5}},1,"",ElementSize → {0.75,0.75}},
   {"Delay",{{5,5},{3,5}},1,"",ElementSize → {0.75,0.75}},
   {"Delay",{{3,4},{5,4}},1,"",ElementSize → {0.75,0.75}},
   {"Delay",{{5,4},{7,4}},1,"",ElementSize → {0.75,0.75}},
   {"Delay",{{7,3},{5,3}},1,"",ElementSize → {0.75,0.75}},
   {"Delay",{{5,3},{3,3}},1,"",ElementSize → {0.75,0.75}},
   {"Delay",{{3,2},{5,2}},1,"",ElementSize → {0.75,0.75}},
   {"Delay",{{5,2},{7,2}},1,"",ElementSize →
{0.75,0.75}},{"Delay",{{3,12},{3,10}},1,"",ElementSize→{0.75,0.7
5}},
   {"Delay",{{3,16},{3,14}},1,"",ElementSize→{0.75,0.75}},
   {"Delay",{{8,13},{8,17}},1,"",ElementSize→{0.75,0.75}},
   {"Delay",{{3,20},{3,18}},1,"",ElementSize→{0.75,0.75}},
   {"Delay",{{8,17},{8,21}},1,"",ElementSize→{0.75,0.75}},
   {"Delay",{{3,24},{3,22}},1,"",ElementSize →{0.75,0.75}},
   {"Delay",{{8,21},{8,25}},1,"",ElementSize→{0.75,0.75}},
   {"Delay",{{7,1},{5,1}}, 1,"",ElementSize → {0.75,0.75}},
   {"Delay",{{5,1},{3,1}}, 1,"",ElementSize → {0.75,0.75}},
   {"Delay",{{8,25},{8,29}}, 1,"",ElementSize → {0.75,0.75}},
   {"Delay", {{3, 28}, {3, 26}}, 1, "",ElementSize →
{0.75,0.75}},
   {"Output", {13, 30}, Y, "", TextOffset -> {-1, 0}},
   {"Multiplier", {{9, 10}, {11, 10}}, a0, ""},
   {"Multiplier", {{9, 14}, {11, 14}}, a1, ""},
   {"Multiplier", {{9, 18}, {11, 18}}, a2, ""},
   {"Multiplier", {{9, 22}, {11, 22}}, a3, ""},
```

```
   {"Multiplier", {{9, 26}, {11, 26}}, a4, ""},
   {"Multiplier", {{9, 30}, {11, 30}}, a5, ""},
   {"Adder", {{11, 14}, {12, 13}, {13, 14}, {12, 15}}, {1,
1,0,2}, " "},
   {"Adder", {{11, 18}, {12, 17}, {13, 18}, {12, 19}}, {1,
1,0,2}, " "},
   {"Adder", {{11, 22}, {12, 21}, {13, 22}, {12, 23}}, {1,
1,0,2}, " "},
   {"Adder", {{11, 26}, {12, 25}, {13, 26}, {12, 27}}, {1,
1,0,2}, " "},
   {"Adder", {{11, 30}, {12, 29}, {13, 30}, {12, 31}}, {1, 1,
2,0}, " "},
   {"Line", {{11, 10},{12, 10}, {12, 11}}},
   {"Delay", {{12, 11}, {12, 13}}, 1, "",ElementSize →
{0.75,0.75}},
   {"Delay", {{12, 15}, {12, 17}}, 1, "",ElementSize →
{0.75,0.75}},
   {"Delay", {{12, 19}, {12, 21}}, 1, "",ElementSize →
{0.75,0.75}},
   {"Delay", {{12, 23}, {12, 25}}, 1, "",ElementSize →
{0.75,0.75}},
   {"Delay", {{12, 27}, {12, 29}}, 1, "",ElementSize →
{0.75,0.75}},

   {"Line", {{7,2},{7, 1}}},{"Line",{{4,9},{4,8}}}};

ShowSchematic[sch3/.{U→"\[UpArrow]5",D→"\[DownArrow]5"},FontSiz
e →7,ShowNodes→True,Frame → False,GridLines → None];
```



subs1=ToRules[seq1□seq0]

```
{r1→x,0→0,0→0,0→0,0→0,0→0,r2→0,0→0,0→0,0→0,0→0,0→0,r3→0,
0→0,0→0,0→0,0→0,0→0,r4→0,0→0,0→0,0→0,0→0,0→0,r5→0,0→0,0
→0,0→0,0→0,0→0,r6→0,0→0,0→0,0→0,0→0,0→0,r7→0,0→0,0→0,0→
0,0→0,0→0,r8→0,0→0,0→0,0→0,0→0,0→0,0→0,r9→0,0→0,0→0,0→0,0→0,
0→0,r10→0,0→0,0→0,0→0,0→0,0→0,0→0,r11→0,0→0,0→0,0→0,0→0,0→0,
r12→0,0→0,0→0,0→0,0→0,0→0,0→0,r13→0,0→0,0→0,0→0,0→0,0→0,r14→
0,0→0,0→0,0→0,0→0,0→0,0→0,r15→0,0→0,0→0,0→0,0→0,0→0,r16→0,0→
0,0→0,0→0,0→0,0→0,r17→0,0→0,0→0,0→0,0→0,0→0,r18→0,0→0,0→
0,0→0,0→0,0→0,r19→0,0→0,0→0,0→0,0→0,0→0,r20→0,0→0,0→0,0→
0,0→0,0→0}
```

seq2

{{0},{p1},{0},{0},{0},{0},{0},{p2},{0},{0},{0},{0},{0},{p3},{0},{0},{0},{0},{0},{p4},{0},{0},{0},{0},{0},{p5},{0},{0},{0},{0},{0},{p6},{0},{0},{0},{0},{0},{0},{p7},{0},{0},{0},{0},{0},{p8},{0},{0},{0},{0},{0},{p9},{0},{0},{0},{0},{0},{p10},{0},{0},{0},{0},{0},{p11},{0},{0},{0},{0},{0},{p12},{0},{0},{0},{0},{0},{p13},{0},{0},{0},{0},{0},{p14},{0},{0},{0},{0},{0},{p15},{0},{0},{0},{0},{0},{p16},{0},{0},{0},{0},{0},{p17},{0},{0},{0},{0},{0},{p18},{0},{0},{0},{0},{0},{p19},{0},{0},{0},{0},{0},{p20},{0},{0},{0},{0}}

out0

{{b x},{0},{x-b^2 x},{0},{-b x+b^3 x},{0},{b^2 x-b^4 x},{0},{-b^3 x+b^5 x},{0},{b^4 x-b^6 x},{0},{-b^5 x+b^7 x},{0},{b^6 x-b^8 x},{0},{-b^7 x+b^9 x},{0},{b^8 x-b^{10} x},{0}}

subs1=ToRules[UnitSymbolicSequence[20,p]□out0]

{p1→b x,p2→0,p3→x-b^2 x,p4→0,p5→-b x+b^3 x,p6→0,p7→b^2 x-b^4 x,p8→0,p9→-b^3 x+b^5 x,p10→0,p11→b^4 x-b^6 x,p12→0,p13→-b^5 x+b^7 x,p14→0,p15→b^6 x-b^8 x,p16→0,p17→-b^7 x+b^9 x,p18→0,p19→b^8 x-b^{10} x,p20→0}

subs2=ToRules[UnitSymbolicSequence[20,q]□out1]

{q1→b p1,q2→b p2,q3→p1-b^2 p1+b p3,q4→p2-b^2 p2+b p4,q5→-b p1+b^3 p1+p3-b^2 p3+b p5,q6→-b p2+b^3 p2+p4-b^2 p4+b p6,q7→b^2 p1-b^4 p1-b p3+b^3 p3+p5-b^2 p5+b p7,q8→b^2 p2-b^4 p2-b p4+b^3 p4+p6-b^2 p6+b p8,q9→-b^3 p1+b^5 p1+b^2 p3-b^4 p3-b p5+b^3 p5+p7-b^2 p7+b p9,q10→b p10-b^3 p2+b^5 p2+b^2 p4-b^4 p4-b p6+b^3 p6+p8-b^2 p8,q11→b^4 p1-b^6 p1+b p11-b^3 p3+b^5 p3+b^2 p5-b^4 p5-b p7+b^3 p7+p9-b^2 p9,q12→p10-b^2 p10+b p12+b^4 p2-b^6 p2-b^3 p4+b^5 p4+b^2 p6-b^4 p6-b p8+b^3 p8,q13→-b^5 p1+b^7 p1+p11-b^2 p11+b p13+b^4 p3-b^6 p3-b^3 p5+b^5 p5+b^2 p7-b^4 p7-b p9+b^3 p9,q14→-b p10+b^3 p10+p12-b^2 p12+b p14-b^5 p2+b^7 p2+b^4 p4-b^6 p4-b^3 p6+b^5 p6+b^2 p8-b^4 p8,q15→b^6 p1-b^8 p1-b p11+b^3 p11+p13-b^2 p13+b p15-b^5 p3+b^7 p3+b^4 p5-b^6 p5-b^3 p7+b^5 p7+b^2 p9-b^4 p9,q16→b^2 p10-b^4 p10-b p12+b^3 p12+p14-b^2 p14+b p16+b^6 p2-b^8 p2-b^5 p4+b^7 p4+b^4 p6-b^6 p6-b^3 p8+b^5 p8,q17→-b^7 p1+b^9 p1+b^2 p11-b^4 p11-b p13+b^3 p13+p15-b^2 p15+b p17+b^6 p3-b^8 p3-b^5 p5+b^7 p5+b^4 p7-b^6 p7-b^3 p9+b^5 p9,q18→-b^3 p10+b^5 p10+b^2 p12-b^4 p12-b p14+b^3 p14+p16-b^2 p16+b p18-b^7 p2+b^9 p2+b^6 p4-b^8 p4-b^5 p6+b^7 p6+b^4 p8-b^6 p8,q19→b^8 p1-b^{10} p1-b^3 p11+b^5 p11+b^2 p13-b^4 p13-b p15+b^3 p15+p17-b^2 p17+b p19-b^7 p3+b^9 p3+b^6 p5-b^8 p5-b^5 p7+b^7 p7+b^4 p9-b^6 p9,q20→b^4 p10-b^6 p10-b^3 p12+b^5 p12+b^2 p14-b^4 p14-b p16+b^3 p16+p18-b^2 p18+b^8 p2-b^{10} p2+b p20-b^7 p4+b^9 p4+b^6 p6-b^8 p6-b^5 p8+b^7 p8}

subs3=ToRules[UnitSymbolicSequence[20,w]□out2]

{w1→b q1,w2→b q2,w3→q1-b^2 q1+b q3,w4→q2-b^2 q2+b q4,w5→-b q1+b^3 q1+q3-b^2 q3+b q5,w6→-b q2+b^3 q2+q4-b^2 q4+b q6,w7→b^2 q1-b^4 q1-b q3+b^3 q3+q5-b^2 q5+b q7,w8→b^2 q2-b^4 q2-b q4+b^3 q4+q6-b^2 q6+b q8,w9→-b^3 q1+b^5 q1+b^2 q3-b^4 q3-b q5+b^3 q5+q7-b^2 q7+b q9,w10→b q10-b^3 q2+b^5 q2+b^2 q4-b^4 q4-b q6+b^3 q6+q8-b^2 q8,w11→b^4 q1-b^6 q1+b q11-b^3 q3+b^5 q3+b^2 q5-b^4 q5-b q7+b^3 q7+q9-b^2 q9,w12→q10-b^2 q10+b q12+b^4 q2-b^6 q2-b^3 q4+b^5 q4+b^2 q6-b^4 q6-b q8+b^3 q8,w13→-b^5 q1+b^7

q1+q11-b^2 q11+b q13+b^4 q3-b^6 q3-b^3 q5+b^5 q5+b^2 q7-b^4 q7-b q9+b^3 q9,w14→-b q10+b^3 q10+q12-b^2 q12+b q14-b^5 q2+b^7 q2+b^4 q4-b^6 q4-b^3 q6+b^5 q6+b^2 q8-b^4 q8,w15→b^6 q1-b^8 q1-b q11+b^3 q11+q13-b^2 q13+b q15-b^5 q3+b^7 q3+b^4 q5-b^6 q5-b^3 q7+b^5 q7+b^2 q9-b^4 q9,w16→b^2 q10-b^4 q10-b q12+b^3 q12+q14-b^2 q14+b q16+b^6 q2-b^8 q2-b^5 q4+b^7 q4+b^4 q6-b^6 q6-b^3 q8+b^5 q8,w17→-b^7 q1+b^9 q1+b^2 q11-b^4 q11-b q13+b^3 q13+q15-b^2 q15+b q17+b^6 q3-b^8 q3-b^5 q5+b^7 q5+b^4 q7-b^6 q7-b^3 q9+b^5 q9,w18→-b^3 q10+b^5 q10+b^2 q12-b^4 q12-b q14+b^3 q14+q16-b^2 q16+b q18-b^7 q2+b^9 q2+b^6 q4-b^8 q4-b^5 q6+b^7 q6+b^4 q8-b^6 q8,w19→b^8 q1-b^10 q1-b^3 q11+b^5 q11+b^2 q13-b^4 q13-b q15+b^3 q15+q17-b^2 q17+b q19-b^7 q3+b^9 q3+b^6 q5-b^8 q5-b^5 q7+b^7 q7+b^4 q9-b^6 q9,w20→b^4 q10-b^6 q10-b^3 q12+b^5 q12+b^2 q14-b^4 q14-b q16+b^3 q16+q18-b^2 q18+b^8 q2-b^10 q2+b q20-b^7 q4+b^9 q4+b^6 q6-b^8 q6-b^5 q8+b^7 q8}

subs4=ToRules[UnitSymbolicSequence[20,v]□out3]

{v1→b w1,v2→b w2,v3→w1-b^2 w1+b w3,v4→w2-b^2 w2+b w4,v5→-b w1+b^3 w1+w3-b^2 w3+b w5,v6→-b w2+b^3 w2+w4-b^2 w4+b w6,v7→b^2 w1-b^4 w1-b w3+b^3 w3+w5-b^2 w5+b w7,v8→b^2 w2-b^4 w2-b w4+b^3 w4+w6-b^2 w6+b w8,v9→-b^3 w1+b^5 w1+b^2 w3-b^4 w3-b w5+b^3 w5+w7-b^2 w7+b w9,v10→b w10-b^3 w2+b^5 w2+b^2 w4-b^4 w4-b w6+b^3 w6+w8-b^2 w8,v11→b^4 w1-b^6 w1+b w11-b^3 w3+b^5 w3+b^2 w5-b^4 w5-b w7+b^3 w7+w9-b^2 w9,v12→w10-b^2 w10+b w12+b^4 w2-b^6 w2-b^3 w4+b^5 w4+b^2 w6-b^4 w6-b w8+b^3 w8,v13→-b^5 w1+b^7 w1+w11-b^2 w11+b w13+b^4 w3-b^6 w3-b^3 w5+b^5 w5+b^2 w7-b^4 w7-b w9+b^3 w9,v14→-b w10+b^3 w10+w12-b^2 w12+b w14-b^5 w2+b^7 w2+b^4 w4-b^6 w4-b^3 w6+b^5 w6+b^2 w8-b^4 w8,v15→b^6 w1-b^8 w1-b w11+b^3 w11+w13-b^2 w13+b w15-b^5 w3+b^7 w3+b^4 w5-b^6 w5-b^3 w7+b^5 w7+b^2 w9-b^4 w9,v16→b^2 w10-b^4 w10-b w12+b^3 w12+w14-b^2 w14+b w16+b^6 w2-b^8 w2-b^5 w4+b^7 w4+b^4 w6-b^6 w6-b^3 w8+b^5 w8,v17→-b^7 w1+b^9 w1+b^2 w11-b^4 w11-b w13+b^3 w13+w15-b^2 w15+b w17+b^6 w3-b^8 w3-b^5 w5+b^7 w5+b^4 w7-b^6 w7-b^3 w9+b^5 w9,v18→-b^3 w10+b^5 w10+b^2 w12-b^4 w12-b w14+b^3 w14+w16-b^2 w16+b w18-b^7 w2+b^9 w2+b^6 w4-b^8 w4-b^5 w6+b^7 w6+b^4 w8-b^6 w8,v19→b^8 w1-b^10 w1-b^3 w11+b^5 w11+b^2 w13-b^4 w13-b w15+b^3 w15+w17-b^2 w17+b w19-b^7 w3+b^9 w3+b^6 w5-b^8 w5-b^5 w7+b^7 w7+b^4 w9-b^6 w9,v20→b^4 w10-b^6 w10-b^3 w12+b^5 w12+b^2 w14-b^4 w14-b w16+b^3 w16+w18-b^2 w18+b^8 w2-b^10 w2+b w20-b^7 w4+b^9 w4+b^6 w6-b^8 w6-b^5 w8+b^7 w8}

subs5=ToRules[UnitSymbolicSequence[20,u]□out4]

{u1→b v1,u2→b v2,u3→v1-b^2 v1+b v3,u4→v2-b^2 v2+b v4,u5→-b v1+b^3 v1+v3-b^2 v3+b v5,u6→-b v2+b^3 v2+v4-b^2 v4+b v6,u7→b^2 v1-b^4 v1-b v3+b^3 v3+v5-b^2 v5+b v7,u8→b^2 v2-b^4 v2-b v4+b^3 v4+v6-b^2 v6+b v8,u9→-b^3 v1+b^5 v1+b^2 v3-b^4 v3-b v5+b^3 v5+v7-b^2 v7+b v9,u10→b v10-b^3 v2+b^5 v2+b^2 v4-b^4 v4-b v6+b^3 v6+v8-b^2 v8,u11→b^4 v1-b^6 v1+b v11-b^3 v3+b^5 v3+b^2 v5-b^4 v5-b v7+b^3 v7+v9-b^2 v9,u12→v10-b^2 v10+b v12+b^4 v2-b^6 v2-b^3 v4+b^5 v4+b^2 v6-b^4 v6-b v8+b^3 v8,u13→-b^5 v1+b^7 v1+v11-b^2 v11+b v13+b^4 v3-b^6 v3-b^3 v5+b^5 v5+b^2 v7-b^4 v7-b v9+b^3 v9,u14→-b v10+b^3 v10+v12-b^2 v12+b v14-b^5 v2+b^7 v2+b^4 v4-b^6 v4-b^3 v6+b^5 v6+b^2 v8-b^4 v8,u15→b^6 v1-b^8 v1-b v11+b^3 v11+v13-b^2 v13+b v15-b^5 v3+b^7 v3+b^4 v5-b^6 v5-b^3 v7+b^5 v7+b^2 v9-b^4 v9,u16→b^2 v10-b^4 v10-b v12+b^3 v12+v14-b^2 v14+b v16+b^6 v2-b^8 v2-b^5 v4+b^7 v4+b^4 v6-b^6

v6-b^3 v8+b^5 v8,u17→-b^7 v1+b^9 v1+b^2 v11-b^4 v11-b v13+b^3 v13+v15-b^2 v15+b v17+b^6 v3-b^8 v3-b^5 v5+b^7 v5+b^4 v7-b^6 v7-b^3 v9+b^5 v9,u18→-b^3 v10+b^5 v10+b^2 v12-b^4 v12-b v14+b^3 v14+v16-b^2 v16+b v18-b^7 v2+b^9 v2+b^6 v4-b^8 v4-b^5 v6+b^7 v6+b^4 v8-b^6 v8,u19→b^8 v1-b^10 v1-b^3 v11+b^5 v11+b^2 v13-b^4 v13-b v15+b^3 v15+v17-b^2 v17+b v19-b^7 v3+b^9 v3+b^6 v5-b^8 v5-b^5 v7+b^7 v7+b^4 v9-b^6 v9,u20→b^4 v10-b^6 v10-b^3 v12+b^5 v12+b^2 v14-b^4 v14-b v16+b^3 v16+v18-b^2 v18+b^8 v2-b^10 v2+b v20-b^7 v4+b^9 v4+b^6 v6-b^8 v6-b^5 v8+b^7 v8}

seq2 = ListToSequence [DiscreteSystemImplementationProcessing[UpsampleSequence[UnitSymbolicSequence[20,p]/.subs1,6], {0,0,0,0,0,0,0,0,0,0}, {}, procName][[1]][[All,2]]]

{{0},{b x},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{x-b^2 x},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{-b x+b^3 x},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{b^2 x-b^4 x},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{-b^3 x+b^5 x},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{b^4 x-b^6 x},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{-b^5 x+b^7 x},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{b^6 x-b^8 x},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{-b^7 x+b^9 x},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{b^8 x-b^10 x},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0}}

out1/.subs1//Expand

{{b^2 x},{0},{2 b x-2 b^3 x},{0},{x-4 b^2 x+3 b^4 x},{0},{-2 b x+6 b^3 x-4 b^5 x},{0},{3 b^2 x-8 b^4 x+5 b^6 x},{0},{-4 b^3 x+10 b^5 x-6 b^7 x},{0},{5 b^4 x-12 b^6 x+7 b^8 x},{0},{-6 b^5 x+14 b^7 x-8 b^9 x},{0},{7 b^6 x-16 b^8 x+9 b^10 x},{0},{-8 b^7 x+18 b^9 x-10 b^11 x},{0}}

out2/.subs2/.subs1//Expand

{{b^3 x},{0},{3 b^2 x-3 b^4 x},{0},{3 b x-9 b^3 x+6 b^5 x},{0},{x-9 b^2 x+18 b^4 x-10 b^6 x},{0},{-3 b x+18 b^3 x-30 b^5 x+15 b^7 x},{0},{6 b^2 x-30 b^4 x+45 b^6 x-21 b^8 x},{0},{-10 b^3 x+45 b^5 x-63 b^7 x+28 b^9 x},{0},{15 b^4 x-63 b^6 x+84 b^8 x-36 b^10 x},{0},{-21 b^5 x+84 b^7 x-108 b^9 x+45 b^11 x},{0},{28 b^6 x-108 b^8 x+135 b^10 x-55 b^12 x},{0}}

outseqHS1={b x,0,x-b^2 x,0,-b x+b^3 x,0,b^2 x-b^4 x,0,-b^3 x+b^5 x,0,b^4 x-b^6 x,0,-b^5 x+b^7 x,0,b^6 x-b^8 x,0,-b^7 x+b^9 x,0,b^8 x-b^10 x,0}; outseqHS1-out0

{{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0}}

outseqHS2={b^2 x,0,2 b x-2 b^3 x,0,x-4 b^2 x+3 b^4 x,0,-2 b x+6 b^3 x-4 b^5 x,0,3 b^2 x-8 b^4 x+5 b^6 x,0,-4 b^3 x+10 b^5 x-6 b^7 x,0,5 b^4 x-12 b^6 x+7 b^8 x,0,-6 b^5 x+14 b^7 x-8 b^9 x,0,7 b^6 x-16 b^8 x+9 b^10 x,0,-8 b^7 x+18 b^9 x-10 b^11 x,0}

outseqHS2-out1/.subs1//Expand

{b^2 x,0,2 b x-2 b^3 x,0,x-4 b^2 x+3 b^4 x,0,-2 b x+6 b^3 x-4 b^5 x,0,3 b^2 x-8 b^4 x+5 b^6 x,0,-4 b^3 x+10 b^5 x-6 b^7 x,0,5 b^4 x-12 b^6 x+7 b^8 x,0,-6 b^5 x+14 b^7 x-8 b^9 x,0,7 b^6 x-16 b^8 x+9 b^10 x,0,-8 b^7 x+18 b^9 x-10 b^11 x,0}

{{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0}}

outseqHS3={b^3 x,0,3 b^2 x-3 b^4 x,0,3 b x-9 b^3 x+6 b^5 x,0,x-9 b^2 x+18 b^4 x-10 b^6 x,0,-3 b x+18 b^3 x-30 b^5 x+15 b^7 x,0,6 b^2 x-30 b^4 x+45 b^6 x-21 b^8 x,0,-10 b^3 x+45 b^5 x-63 b^7 x+28 b^9 x,0,15 b^4 x-63 b^6 x+84 b^8 x-36 b^{10} x,0,-21 b^5 x+84 b^7 x-108 b^9 x+45 b^{11} x,0,28 b^6 x-108 b^8 x+135 b^{10} x-55 b^{12} x,0}

outseqHS3-out2/.subs2/.subs1//Expand

{b^3 x,0,3 b^2 x-3 b^4 x,0,3 b x-9 b^3 x+6 b^5 x,0,x-9 b^2 x+18 b^4 x-10 b^6 x,0,-3 b x+18 b^3 x-30 b^5 x+15 b^7 x,0,6 b^2 x-30 b^4 x+45 b^6 x-21 b^8 x,0,-10 b^3 x+45 b^5 x-63 b^7 x+28 b^9 x,0,15 b^4 x-63 b^6 x+84 b^8 x-36 b^{10} x,0,-21 b^5 x+84 b^7 x-108 b^9 x+45 b^{11} x,0,28 b^6 x-108 b^8 x+135 b^{10} x-55 b^{12} x,0}

{{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0}}

outseqHS4={b^4 x,0,4 b^3 x-4 b^5 x,0,6 b^2 x-16 b^4 x+10 b^6 x,0,4 b x-24 b^3 x+40 b^5 x-20 b^7 x,0,x-16 b^2 x+60 b^4 x-80 b^6 x+35 b^8 x,0,-4 b x+40 b^3 x-120 b^5 x+140 b^7 x-56 b^9 x,0,10 b^2 x-80 b^4 x+210 b^6 x-224 b^8 x+84 b^{10} x,0,-20 b^3 x+140 b^5 x-336 b^7 x+336 b^9 x-120 b^{11} x,0,35 b^4 x-224 b^6 x+504 b^8 x-480 b^{10} x+165 b^{12} x,0,-56 b^5 x+336 b^7 x-720 b^9 x+660 b^{11} x-220 b^{13} x,0}

outseqHS4-out3/.subs3/.subs2/.subs1//Expand

{b^4 x,0,4 b^3 x-4 b^5 x,0,6 b^2 x-16 b^4 x+10 b^6 x,0,4 b x-24 b^3 x+40 b^5 x-20 b^7 x,0,x-16 b^2 x+60 b^4 x-80 b^6 x+35 b^8 x,0,-4 b x+40 b^3 x-120 b^5 x+140 b^7 x-56 b^9 x,0,10 b^2 x-80 b^4 x+210 b^6 x-224 b^8 x+84 b^{10} x,0,-20 b^3 x+140 b^5 x-336 b^7 x+336 b^9 x-120 b^{11} x,0,35 b^4 x-224 b^6 x+504 b^8 x-480 b^{10} x+165 b^{12} x,0,-56 b^5 x+336 b^7 x-720 b^9 x+660 b^{11} x-220 b^{13} x,0}

{{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0}}

outseqHS5={b^5 x,0,5 b^4 x-5 b^6 x,0,10 b^3 x-25 b^5 x+15 b^7 x,0,10 b^2 x-50 b^4 x+75 b^6 x-35 b^8 x,0,5 b x-50 b^3 x+150 b^5 x-175 b^7 x+70 b^9 x,0,x-25 b^2 x+150 b^4 x-350 b^6 x+350 b^8 x-126 b^{10} x,0,-5 b x+75 b^3 x-350 b^5 x+700 b^7 x-630 b^9 x+210 b^{11} x,0,15 b^2 x-175 b^4 x+700 b^6 x-1260 b^8 x+1050 b^{10} x-330 b^{12} x,0,-35 b^3 x+350 b^5 x-1260 b^7 x+2100 b^9 x-1650 b^{11} x+495 b^{13} x,0,70 b^4 x-630 b^6 x+2100 b^8 x-3300 b^{10} x+2475 b^{12} x-715 b^{14} x,0}

outseqHS5-out4/.subs4/.subs3/.subs2/.subs1//Expand

{b^5 x,0,5 b^4 x-5 b^6 x,0,10 b^3 x-25 b^5 x+15 b^7 x,0,10 b^2 x-50 b^4 x+75 b^6 x-35 b^8 x,0,5 b x-50 b^3 x+150 b^5 x-175 b^7 x+70 b^9 x,0,x-25 b^2 x+150 b^4 x-350 b^6 x+350 b^8 x-126 b^{10} x,0,-5 b x+75 b^3 x-350 b^5 x+700 b^7 x-630 b^9 x+210 b^{11} x,0,15 b^2 x-175 b^4 x+700 b^6 x-1260 b^8 x+1050 b^{10} x-330 b^{12} x,0,-35 b^3 x+350 b^5 x-1260 b^7 x+2100 b^9 x-1650 b^{11} x+495 b^{13} x,0,70 b^4 x-630 b^6 x+2100 b^8 x-3300 b^{10} x+2475 b^{12} x-715 b^{14} x,0}

{{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0},{0}}