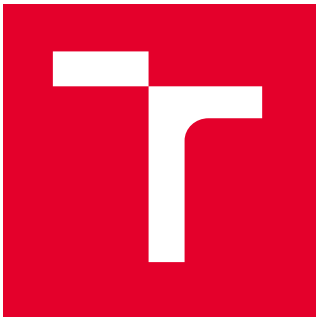


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV MIKROELEKTRONIKY

DEPARTMENT OF MICROELECTRONICS

IMPLEMENTACE LADITELNÉHO ČÍSLICOVÉHO FILTRU DO OBVODU FPGA

IMPLEMENTATION OF TUNABLE DIGITAL FILTER INTO FPGA

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE — **Matěj Štěpán**

AUTHOR

VEDOUCÍ PRÁCE — **Ing. Vojtěch Dvořák, Ph.D.**

SUPERVISOR

BRNO 2023

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní program **Mikroelektronika a technologie**

Ústav mikroelektroniky

Student: Matěj Štěpán ***ID:*** 220843

Ročník: 3 ***Akademický rok:*** 2022/23

NÁZEV TÉMATU:

Implementace laditelného číslicového filtru do obvodu FPGA

POKYNY PRO VYPRACOVÁNÍ:

Seznamte se s problematikou číslicových filtrů, jejich popisem pomocí přenosových funkcí a návrhem číslicových filtrů v prostředí Matlab s nástrojem Filter Designer. Proveďte návrh několika vzorových IIR filtrů dolní propusti a porovnejte jejich struktury. Na základě porovnání struktur zvolte vhodnou architekturu pro implementaci laditelného číslicového filtru IIR, popište v jazyce VHDL a ověřte simulací. Dosažené výsledky porovnejte se vzorovou implementací generovanou nástrojem Filter Designer v Matlabu.

DOPORUČENÁ LITERATURA:

Dle pokynů vedoucího práce.

Termín zadání: 6.2.2023 ***Termín odevzdání:*** 1.6.2023

Vedoucí práce: Ing. Vojtěch Dvořák, Ph.D.

doc. Ing. Pavel Šteffan, Ph.D.
předseda rady studijního programu

Abstrakt

Práce se zabývá problematikou návrhu číslicových IIR filtrů. V práci jsou představeny základní struktury IIR filtrů včetně kaskádové struktury SOS. V teoretické části je posouzena vhodnost koeficientů přenosové funkce, získaných pomocí nástroje *Filter Designer*, pro laditelný IIR filtr implementovaný do obvodu FPGA. V praktické části je popsán navržený laditelný IIR filtr v jazyce VHDL.

Klíčová slova

IIR, číslicový filtr, FPGA, SOS, laditelný, Filter Designer, MATLAB, DSP, RAM, VHDL

Abstract

Proposed bachelor thesis is focused on the design of an IIR filter. This work presents common structures of IIR filters including cascaded SOS structure. The outcome of this work is a summary of theory, assessment of the transfer function coefficients generated by the Filter Designer tool for a tunable IIR filter implemented on an FPGA and finally an implementation of a filter described in VDHL.

Keywords

IIR, digital filter, FPGA, SOS, tunable, Filter Designer, MATLAB, DSP, RAM, VHDL

Bibliografická citace

ŠTĚPÁN, Matěj. *Implementace laditelného číslicového filtru do obvodu FPGA*. Brno, 2023. Dostupné také z: https://www.vut.cz/studenti/zav-prace/detail/152269. Bakalářská práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav mikroelektroniky. Vedoucí práce Vojtěch Dvořák.

Prohlášení autora o původnosti díla

Jméno a příjmení studenta:	Matěj Štěpán
VUT ID studenta:	220843
Typ práce:	Bakalářská práce
Akademický rok:	2022/23
Téma závěrečné práce:	Implementace laditelného číslicového filtru do obvodu FPGA

Prohlašuji, že svou závěrečnou práci jsem vypracoval samostatně pod vedením vedoucí/ho závěrečné práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené závěrečné práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

.

V Brně dne 1. června 2023 ----------

podpis autora

Poděkování

Děkuji vedoucímu práce Ing. Vojtěchu Dvořákovi Ph.D. za odbornou, pedagogickou a metodickou podporu při zpracování práce.

V Brně dne: 1. června 2023 ----------

podpis autora

Obsah

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

Úvod

Tato práce se zabývá problematikou návrhu číslicových filtrů. Motivací je vytvoření HDL popis laditelného číslicového filtru, společně s verifikačním prostředím. Práce je strukturována tak, aby poskytla případnému čtenáři snazší úvod do problematiky a seznámila ho s nástrojem *Filter Designer*, který je součástí knihoven programu MATLAB.

V první kapitole je krátce shrnuta teorie číslicových filtrů. Je zmíněn jejich popis pomocí přenosových funkcí a jak je z těchto funkcí odvozena nejjednodušší fyzická struktura filtru. Dále se kapitola věnuje problematice návrhu filtrů. Shrnuje základní typy struktur včetně kaskádové struktury SOS (*Second-Order Sections*) a jejich vliv na návrh a kvantizační chyby. Na konec vytýká výhody použití programů specificky určených pro návrh filtrů.

Druhá kapitola obsahuje návrhy filtrů typu dolní, horní a pásmová propust, které byly vytvořeny pomocí nástroje *Filter Designer* z knihoven prostředí MATLAB. Prozkoumává použitelnost nástrojem generovaných koeficientů v navazující bakalářské práci. Mezi filtry je porovnána závislost koeficientů přenosové funkce na typu filtru a mezním kmitočtu. Je shrnuto, jakým způsobem nástroj filtry implementuje, a které další užitečné funkce nabízí.

Třetí kapitola líčí funkční princip návrhu – využití RAM paměti pro uložení koeficientů, díky čemuž je lze přepisovat a docílit tak laditelnosti filtru. Je zde k nalezení blokové schéma architektury, bližší popis jednotlivých bloků, popis komunikačního rozhraní a struktura verifikačního prostředí.

Čtvrtá kapitola krátce seznamuje s pomocnými m-skripty, které umožňují tvorbu inicializačního souboru a zrychlují verifikační proces. Následně prezentuje výsledky funkční verifikace vlastního návrhu filtru, tedy porovnání s HDL popisem vygenerovaným pomocí programu *Filter Designer*.

Závěr shrnuje zjištění a výsledky práce, hodnotí kvalitu vytvořeného popisu a pojednává o možných zlepšeních.

1. ČÍSLICOVÉ FILTRY

Číslicové, nebo také digitální filtry, jsou obdobou analogových filtrů. Mezi hlavní důvody, proč se využívají, patří jejich nezávislost na výrobní toleranci součástek, v porovnání snazší proces návrhu a případné úpravy přenosové funkce jen pomocí změny softwaru. Také umožňují zpracování pomocí nekauzálních systémů. Jejich využití ale limituje několik vlastností plynoucích z jejich podstaty.

Číslicové signály jsou signály disktrétní, a tudíž je potřeba dodržet Nyquistův teorém, aby bylo zabráněno aliasingu. To znamená, že vzorkovací frekvence musí být alespoň dvojnásobek nejvyšší frekvenční složky spektra vzorkovaného signálu. Vlivem vzorkování se spektrální obálka signálu nekonečně zrcadlí okolo vzorkovací frekvence. Pokud není teorém dodržen, konce spekter se překrývají a dochází ke zkreslení signálu – aliasingu. Z toho plyne jedno z omezení digitálních filtrů a digitálního zpracování signálů obecně – pokud chceme předejít aliasingu, je jejich spektrum kmitočtově omezené. To ale třeba v případě audio signálů není takový problém. Lidské ucho je schopné v ideálním případě zachytit maximální frekvenci okolo 20 kHz. Při vzorkování se tedy volí vzorkovací frekvence o něco vyšší než dvojnásobek (běžně 48 kHz). Tento prostor navíc (4 kHz) snižuje zkreslení ve slyšitelné části spektra způsobené nelinearitou přenosové charakteristiky analogového anti-aliasingového filtru (dolní propust). Ten je nutný, protože spektrum spojitých signálů je nekonečné. [1]

Digitální filtry se dělí do dvou kategorií, podle jejich impulsní odezvy:

- IIR – *Infinite Impulse Response* – filtry s nekonečnou impulzní odezvou
- FIR – *Finite Impulse Response* – filtry s konečnou impulsní odezvou

Tato práce se bude zabývat pouze prvním typem, tedy IIR filtry s nekonečnou impulsní odezvou. V porovnání s FIR filtry nabízí menší implementační náročnost (tzn. pro dosažení podobných přenosových parametrů je potřeba filtr nižšího řádu), za cenu komplikovanější matematické analýzy a nelinearity fázové charakteristiky. Zároveň je potřeba zmínit, že se tato práce bude zabývat výhradně návrhem pro hradlová pole neboli FPGA.

1.1 Popis filtrů pomocí přenosové funkce

Filtry, ať už analogové nebo číslicové, se popisují pomocí přenosové funkce

$$H(\omega) = \frac{Y(\omega)}{X(\omega)}; \tag{1.1}$$

kde X(ω) je obraz signálu vstupního a Y(ω) je obraz výstupního. V tomto příkladu se jedná o přenosovou funkci lineárního systému se spojitým časem. [1] Pro návrh a analýzu číslicových filtrů se v praxi využívá transformace Z, která slouží pro analýzu diskrétních signálů. Transformovaná přenosová funkce pak vypadá takto [2]:

$$H(z) = \frac{Y(z)}{X(z)} = \sum_{n=-\infty}^{\infty} h[n] z^{-n}; \tag{1.2}$$

kde h[n] je signál s diskrétním časem a z^{-n} je obecné řešení diferenční rovnice. Obecný tvar přenosové charakteristiky pro IIR filtry je potom:

$$H(z) = \frac{Y(z)}{X(z)} = \frac{\sum_{k=0}^{N} b(k)\, z^{-k}}{\sum_{k=1}^{M} a(k)\, z^{-k}}; \tag{1.3}$$

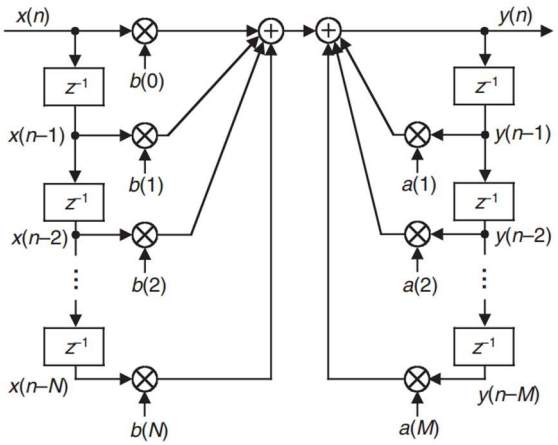
Kde nejvyšší hodnota k určuje stupeň filtru. Funkci lze upravit do tvaru vyjadřující závislost Y(z):

$$Y(z) = X(z) \sum_{k=0}^{N} b(k)\, z^{-k} + Y(z) \sum_{k=1}^{M} a(k)\, z^{-k}. \tag{1.4}$$

Pokud převedeme funkci zpět do domény diskrétního času, získáme rovnici ve tvaru:

$$y[n] = \sum_{k=0}^{N} b(k)\, x[n-k] + \sum_{k=1}^{M} a(k) y[n-k]. \tag{1.5}$$

Z této rovnice potom lze odvodit obecnou strukturu IIR filtru typu *Direct Form I* na obrázku 1.1.

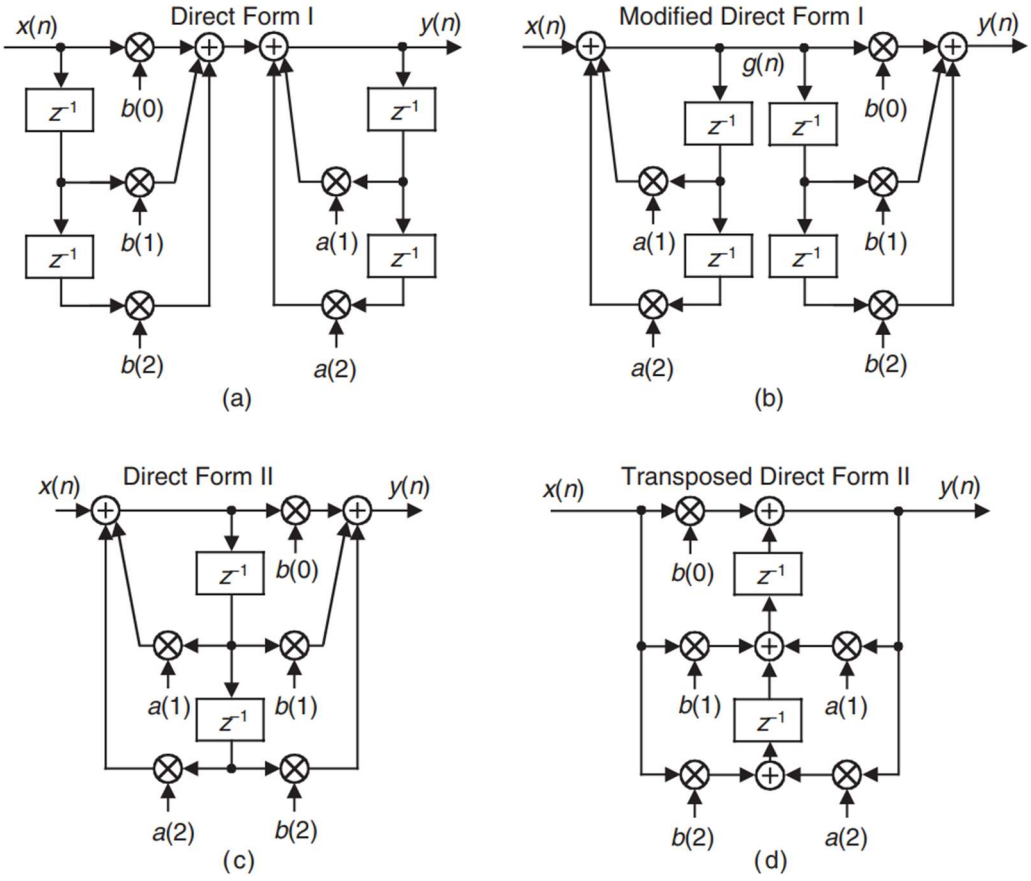


Obrázek 1.1 IIR filtr typu *Direct Form I* [2]

Porovnáním obrázku 1.1 a rovnic (1.4 a (1.5, lze získat představu jak prvek z^{-n} (z^{-k} v rovnici(1.4) reprezentuje zpoždění o k vzorků. V reálné struktuře je zpoždění uskutečněno pomocí paměťového prvku. Sekce filtru s koeficienty b(k) se nazývá *feedforward* a zpětnovazební sekce s koeficienty a(k) se nazývá *feedback*.

1.2 Problematika návrhu číslicových filtrů

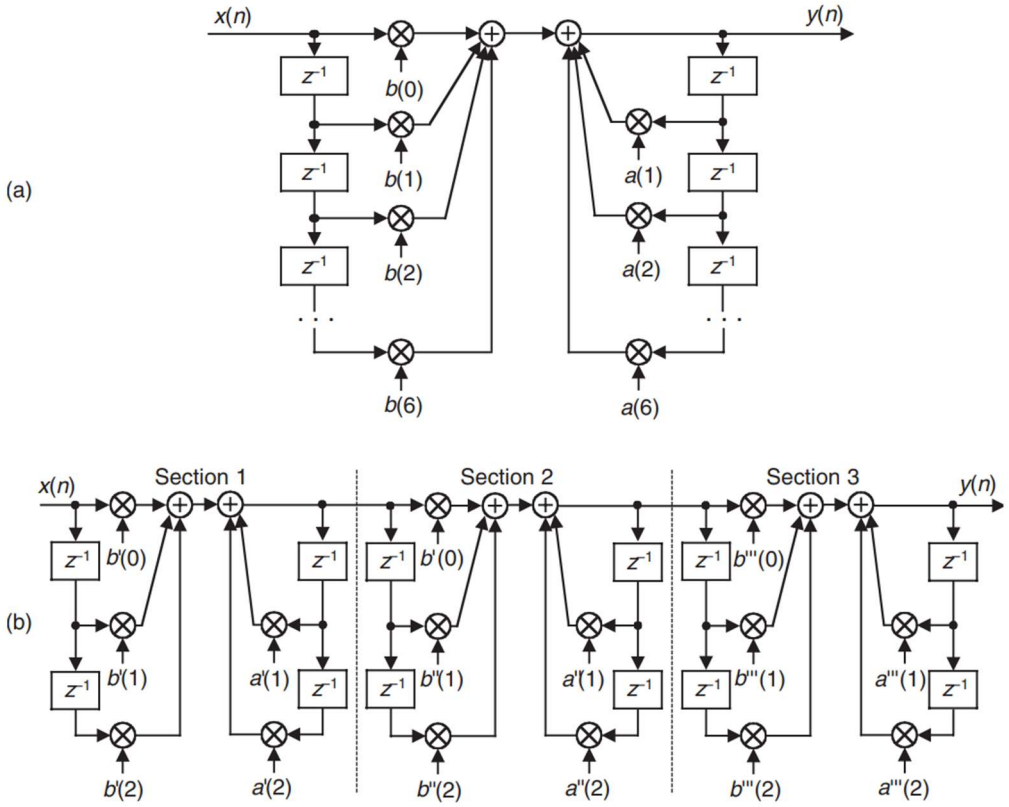
Struktura *Direct Form I*, zobrazená na obrázku 1.1, je nejjednodušší způsob realizace matematické funkce popisující IIR filtr. Jelikož se jedná se o lineární časově invariantní systém, můžeme pořadí sekcí filtru změnit a jeho výstup zůstane stejný. Schéma *b* na obrázku 1.2 ilustruje redundanci paměťových prvků. Jejich odstraněním a spojením datového toku získáme strukturu *Direct Form II* (obr. 1.2, schéma *c*). K implementaci nové struktury je potřeba jen polovina paměťových prvků. Ale jelikož paměťové buňku musí obsáhnout výsledky výpočtů obou sekcí filtru, pokud bude filtr využívat výpočty s pevnou řádovou čárkou, mohou nastat problémy s přetečením (saturací). [2]



Obrázek 1.2 Odvození *Direct Form II* a *Transposed Direct Form II* [2]

Dalším způsobem, jak modifikovat vlastnosti implementovaného filtru pomocí změny struktury, je transpozice. Transponovanou strukturu získáme otočením směru všech signálových toků a záměnou sčítaček za uzly a uzlů za sčítačky. Na schématu *d* na obrázku 1.2 je zobrazena transponovaná struktura *Direct Form II*, tedy *Transposed Direct From II*. Velkou výhodou transponovaných struktur implementovaných na hradlových polích (FPGA) je kratší délka kritické cesty, tedy čas potřebný k ustálení stavů na logických hradlech mezi vstupem a výstupem paměťových prvků. Tím se zvyšuje maximální dosažitelná frekvence hodinového signálu, který obvod využívá.

To lze ověřit na schématech *c* a *d* (obr. 1.2). Na schématu *c* je nejdelší kritická cesta od výstupu z prvního paměťového prvku přes násobičku a(1), první sčítačku vlevo, násobičku b(0) a na výstup přes sčítačku vpravo. Celkem signál bude dvakrát násoben a dvakrát sčítán tří vstupovou sčítačkou. Na schématu *d* vede kritická cesta přes stejné prvky, jen naopak. Na první pohled se může zdát, že jsou cesty ekvivalentní, ale není tomu tak. Horní a spodní sčítačky na schématu *d* jsou pouze dvou vstupové. To znamená, že na kritické cestě schématu *d* je potřeba čekat o dobu sečtení jednoho vstupu méně. Cenou za zrychlení je plocha (množství logiky) – celkem je sčítáno 7 vstupů, tedy o jeden víc než na schématu *c*.



Obrázek 1.3 Kaskádová struktura *Second-Order Sections* [2]

Digitální technika využívá k reprezentaci čísel binární soustavy. Vlivem technických omezení je možné reprezentovat čísla jen v určitém rozsahu. Tomuto rozsahu se jinak říká slovo. Pokud chceme vyjádřit reálné koeficienty IIR filtru, je zapotřebí využít binární reprezentace s frakční částí. Tím, že jsme limitování délkou slova a vlivem nedokonalosti binární reprezentace reálných čísel, budou vznikat nevyhnutelné kvantizační chyby. Každá ze struktur na obrázku 1.2 bude méně či více náchylná ke kvantizačním chybám a přetečení. Při návrhu je tedy kritické zvážit vlastnosti jednotlivých struktur a jejich vliv na výsledný filtr. Obecně platí, že číselný formát *fixed-point*, tedy formát s pevnou řádovou čárkou je jednodušší pro návrh, ale zároveň nejvíce náchylný k chybám způsobeným přetečením. Zvláště pak u filtrů vyšších řádů jde o nemalý problém.

S rostoucím řádem roste i míra obtížnosti analýzy systému a pravděpodobnost vzniku chyb vlivem vnitřního fungování filtru, tj. velikosti koeficientů a bitové rozměry aritmetických a paměťových prvků. Hardwaroví návrháři v praxi těmto jevům předcházejí implementací filtrů ve formě kaskádové struktury – tedy zapojením několika filtrů nižšího (standardně 2. řádu) za sebe. Ilustraci lze vidět na obrázku 1.3. V anglické literatuře tuto metodu najdeme pod pojmem SOS (*Second-Order Sections*). Struktury SOS nejenže minimalizují chyby spojené s přetečením, ale jsou i jednodušší pro návrh. Navrhne se struktura jedné vzorové sekce 2. řádu, kterou lze snadněji analyzovat. Zapojením několika sekcí za sebe a úpravou jejich koeficientů dosáhneme požadované přenosové charakteristiky ekvivalentního filtru stejného řádu s celistvou strukturou. [2]

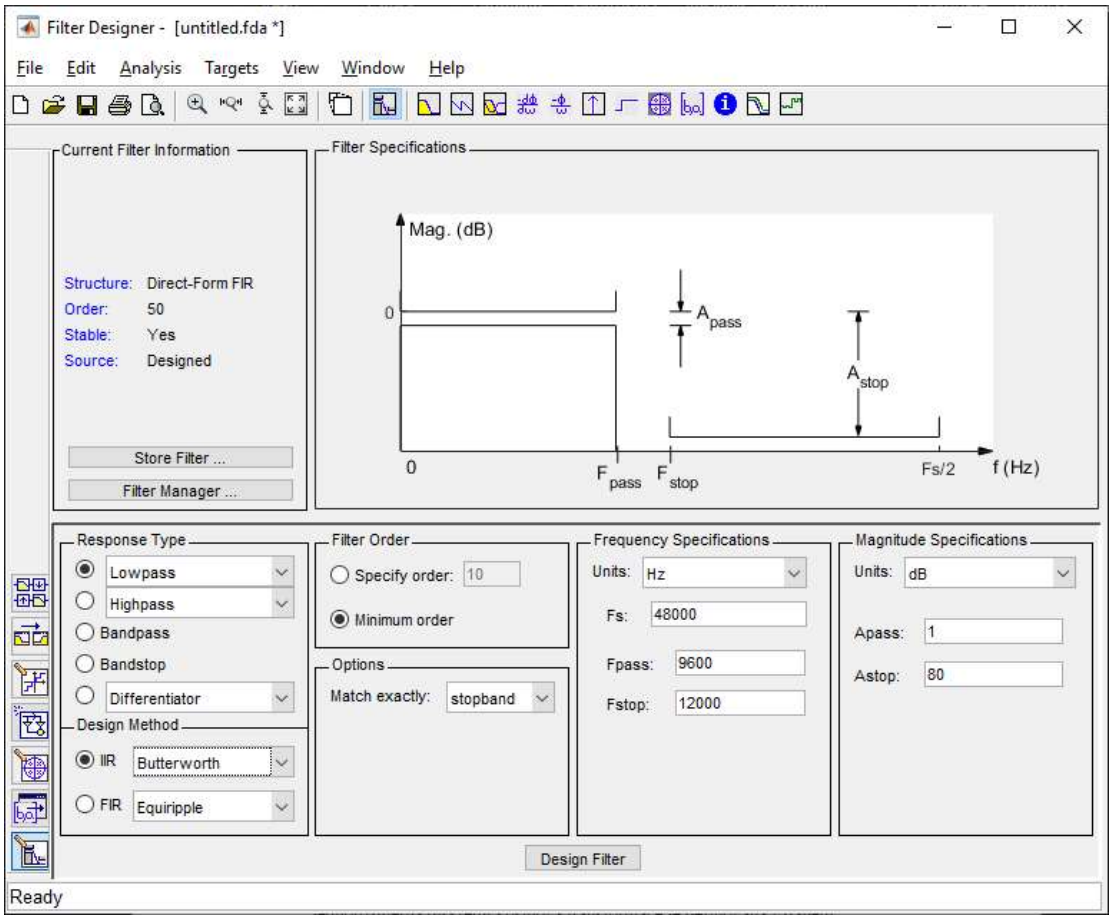
Jak si lze ale představit, matematická analýza filtrů vyšších řádů začne být komplikovaná i přes použití kaskádové struktury 2. druhého řádu. O to víc, pokud chceme brát v potaz analýzu kvantizačních chyb. Pokoušet se o tyto výpočty ručně nebo s použitím tabulkových výpočtů by bylo přinejmenším velmi zdlouhavé a neintuitivní. Proto se dnes využívá numerických nástrojů, speciálně vytvořených pro návrh a analýzu číslicových filtrů. Ty umožnují velmi rychlý matematický i automatizovaný HW návrh na základě jen několika vstupních údajů, např. mezního kmitočtu a útlumu v nepropustném pásmu. Jednomu z těchto nástrojů se věnuje následující kapitola**.** [2]

2. MODELOVÁNÍ FILTRŮ V PROSTŘEDÍ MATLAB

Programovací prostředí MATLAB je hojně využíváno v rozličných oblastech průmyslu. Zjednodušeně řečeno se jedná o velmi chytrou kalkulačku, která v rukou zkušeného uživatele umožňuje výpočetně náročné procesy potřebné pro návrh a analýzu systémů v moderních technologiích. Praktickými příklady jsou kontrolní systémy a analýza digitálních signálu a komunikací. Kromě programovacího prostředí nabízí firma MathWorks v prostředí MATLAB již hotové nástroje pro specifické účely. [5]

2.1 Nástroj Filter Designer

Pro návrh filtru se specifickým cílem získání koeficientů, byl vybrán nástroj *Filter Designer* z knihovny nástrojů prostředí MATLAB. Nástroj umožňuje vysoce intuitivní a efektivní návrh číslicových filtrů a aproximuje některé jejich vlastnosti po implementaci.

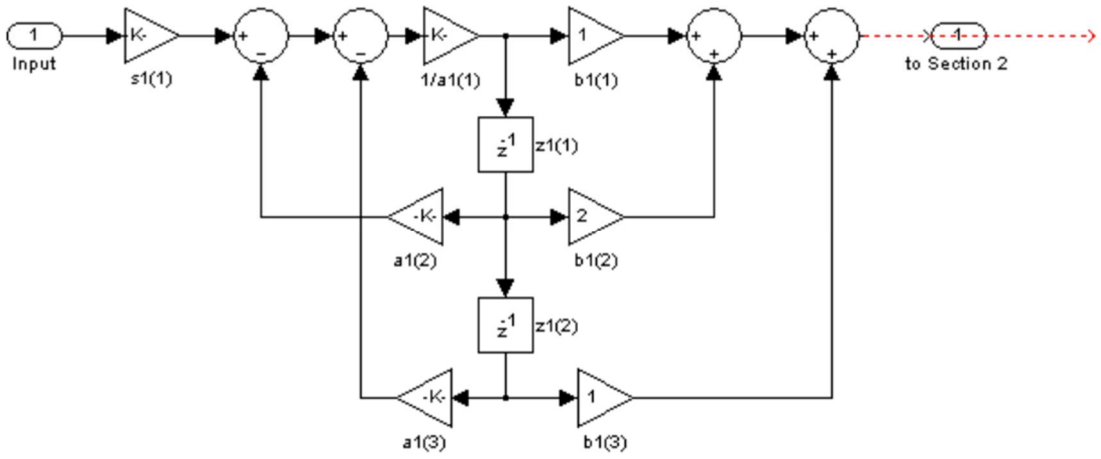


Obrázek 2.1 Grafické prostředí aplikace *Filter Designer* [5]

Grafické prostředí na obrázku 2.1 umožňuje volbu typu odezvy, mezního a vzorkovacího kmitočtu (nebo normovaného mezního kmitočtu), volbu řádu nebo možnost výpočtu nejmenšího možného. Jelikož je zvolený režim minimálního řádu tak je k dispozici i možnost specifikace útlumu v přenosovém a zádržném pásmu. Při návrhu filtru typu IIR lze zvolit možnost návrhu kaskádovou strukturou a strukturu samotné kaskády. V levé části rozhraní jsou k dispozici panely pro úpravu vlastností návrhu jako editor pólů a nul a kvantizační panel. Panel pro aproximaci kvantizované přenosové charakteristiky umožňuje rychlý odhad vlastností automaticky generovaného návrhu (například v jazyce VHDL).

2.2 Návrh filtrů v nástroji Filter Designer

Vzhledem k cíli práce, tj. návrh a implementace laditelného číslicového filtru do FPGA, budou při návrhu vzorových filtrů brány v potaz zjištění zmíněná v kapitole 1.2, konkrétně pozitivní vliv struktury SOS na jednoduchost návrhu. Všechny filtry tedy budou v této struktuře navrhovány.



Obrázek 2.2 SOS sekce struktury *Direct Form II* [7]

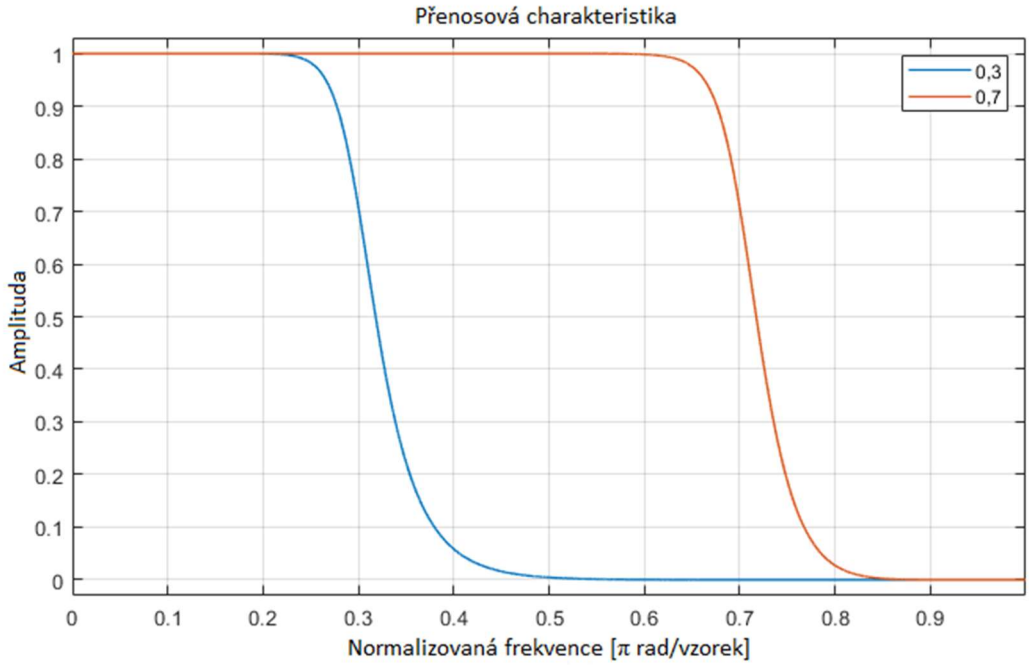
Jako struktura jednotlivých sekcí byla zvolena *Direct Form II*. Způsob implementace programem *Filter Designer* lze vidět na obrázku 2.2. Pro každý typ přenosové charakteristiky budou navrženy dva filtry 8. řádu typu Butterworth s rozdílnými mezními kmitočty a budou porovnány jejich koeficienty. Koeficienty budou generovány v 16bitovém formátu s pevnou řádovou čárkou. Volba šířky řádové části a dalších parametrů v kvantizačním panelu bude ponechána na nástroji.

2.2.1 Návrh filtru typu dolní propust

Na obrázku 2.3 lze vidět amplitudovou přenosovou charakteristiku navržených dolních propustí. Jako mezní kmitočty byly zvoleny hodnoty ω_0 = 0,3 a 0,7, kde ω značí normalizovaný úhlový kmitočet v π radiánech/vzorek. Tato jednotka vztahuje mezní kmitočet k vzorkovací frekvenci podle vztahu

$$\omega = \frac{2\pi f}{f_{vz.}}; \tag{2.1}$$

Kde f je proměnná frekvence a $f_{vz.}$ Je vzorkovací frekvence. Normalizovaná frekvence se pohybuje v rozmezí <-π, π>. Například frekvence ω = 0,5 udává kmitočet rovný jedné čtvrtině vzorkovací frekvence. [2]



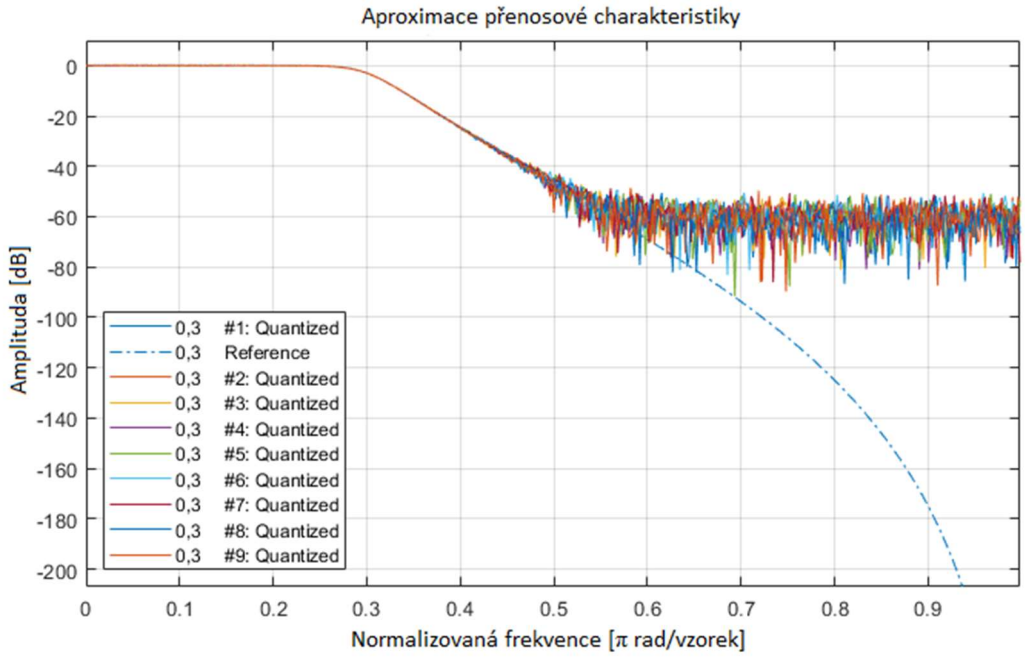
Obrázek 2.3 Přenosová charakteristika dolních propustí

V tabulce 2.1 jsou k vidění koeficienty jednotlivých sekcí struktury SOS. Hodnoty mezního kmitočtu byli zvolené kvůli jejich symetrii okolo čtvrtiny vzorkovací frekvence, tedy ω = 0,5. Koeficienty čitatele přenosové funkce všech sekcí pro oba filtry jsou nezávislé na hodnotě mezního kmitočtu. Mění se pouze koeficienty jmenovatele, tedy pozice pólů a zisk jednotlivých sekcí. Zároveň díky symetrii zvolených kmitočtů okolo ω = 0,5. Je absolutní hodnota koeficientů korespondujících sekcí stejná.

Tabulka 2.1 Vygenerované koeficienty dolních propustí

1. Filtr (ω_0 = 0,3)							
Sekce	**Čitatel**			**Jmenovatel**			**Zisk**
1.	1	2	1	1	-1,01532	0,72737	0,17801
2.	1	2	1	1	-0,81104	0,37982	0,14220
3.	1	2	1	1	-0,70281	0,19569	0,12322
4.	1	2	1	1	-0,65547	0,11516	0,11492
2. Filtr (ω_0 = 0,7)							
1.	1	2	1	1	1,01532	0,72737	0,68567
2.	1	2	1	1	0,81104	0,37982	0,54771
3.	1	2	1	1	0,70281	0,19569	0,47463
4.	1	2	1	1	0,65547	0,11516	0,44266

Na obrázku 2.4 je několik iterací předpokládané reálné přenosové charakteristiky filtru s mezním kmitočtem ω_0 = 0,3, kterou program vypočítává na základě údajů v kvantizačním panelu a zvolené struktury. Pro stejné parametry vychází charakteristika po každém výpočtu jinak. Na webu Mathworks [8] lze dohledat, že nástroj *Filter Designer* používá k získání této charakteristiky metodu Monte Carlo (blíže nespecifikováno). Nejedná se tedy o konkrétní výpočet chyby způsobené kvantizací, nýbrž aproximaci. Zobrazení několika iterací na sobě bylo zvoleno pro lepší ilustraci výsledků této metody. Modrá přerušovaná křivka je referenční přenosová charakteristika.



Obrázek 2.4 Aproximace přenosové charakteristiky

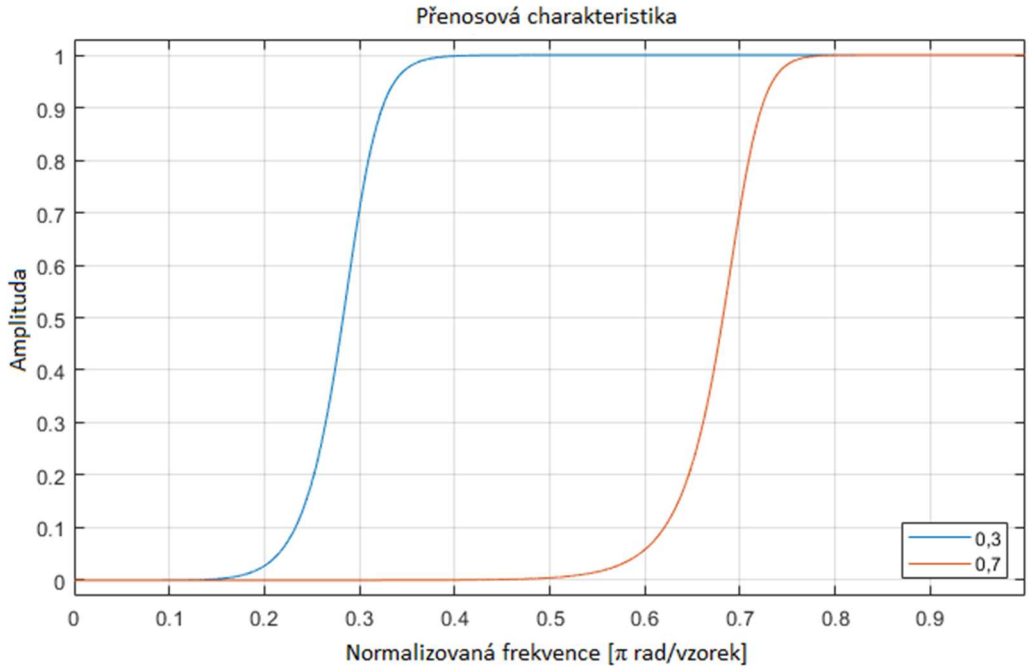
Nejhustší oblast šumu v nepropustném pásmu okolo -60 dB koreluje s chybou způsobenou kvantizací. Délka řádové části výstupního slova byla programem nastavena na 10 bitů. Pokud dosadíme váhu LSB do rovnice (2.2) [1]:

$$|H(\omega)|_{dB} = 20\,log_{10}|H(\omega)| = 20\,log_{10}\,2^{-10} \approx -60{,}206\,dB; \qquad (2.2)$$

dostaneme hodnotu přibližně -60 dB.

2.2.2 Návrh filtru typu horní propust

Obrázek 2.5 ukazuje charakteristiku horních propustí s identickými vstupními parametry jako v kapitole 0 na obrázku 2.3.



Obrázek 2.5 Přenosová charakteristika horních propustí

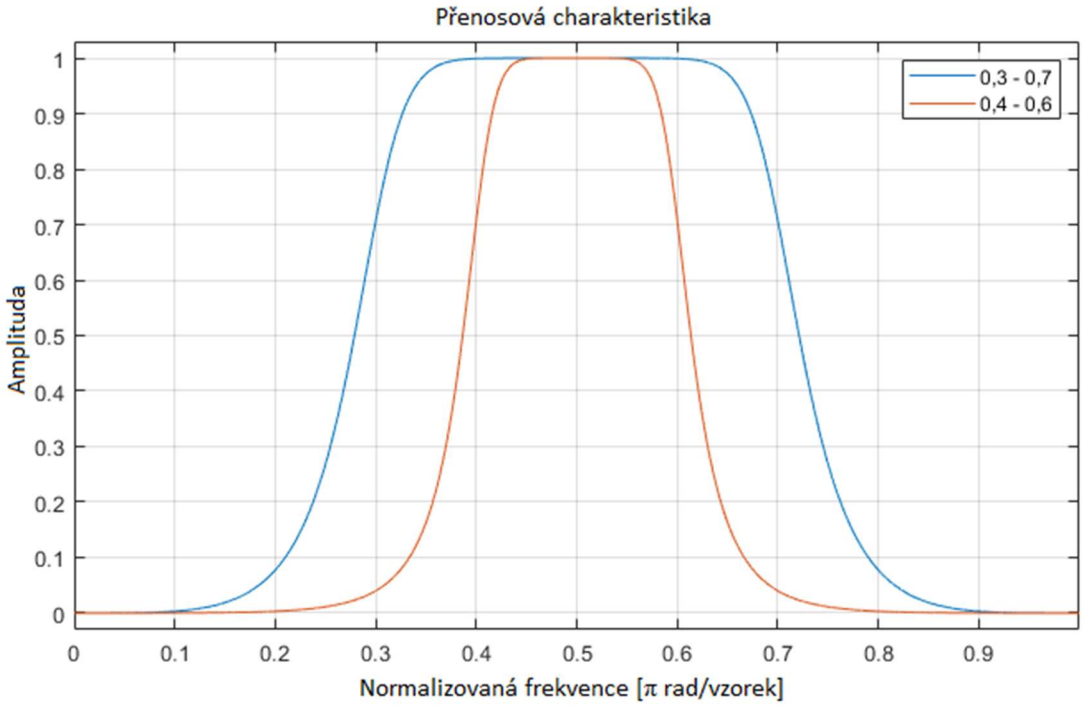
V tabulce 2.2 se objevují stejné hodnoty jako tabulce 2.1. Oproti dolním propustím se u čitatele horních propustí obrátilo jedno znaménko a zisk sekcí prvního filtru byl zaměněn za zisk sekcí druhého. Koeficienty jmenovatele zůstávají stejné.

Tabulka 2.2 Vygenerované koeficienty horních propustí

1. Filtr (ω_0 = 0,3)							
Sekce	**Čitatel**			**Jmenovatel**			**Zisk**
1.	1	-2	1	1	-1,01532	0,72737	0,68567
2.	1	-2	1	1	-0,81104	0,37982	0,54771
3.	1	-2	1	1	-0,70281	0,19569	0,47463
4.	1	-2	1	1	-0,65547	0,11516	0,44266
2. Filtr (ω_0 = 0,7)							
1.	1	-2	1	1	1,01532	0,72737	0,17801
2.	1	-2	1	1	0,81104	0,37982	0,14220
3.	1	-2	1	1	0,70281	0,19569	0,12322
4.	1	-2	1	1	0,65547	0,11516	0,11492

2.2.3 Návrh filtru typu pásmová propust

Při návrhu pásmových propustí, s přenosovou charakteristikou na obrázku 2.6, byly zvoleny mezní kmitočty ω_0 = 0,3 – 0,7 a 0,4 – 0,6.



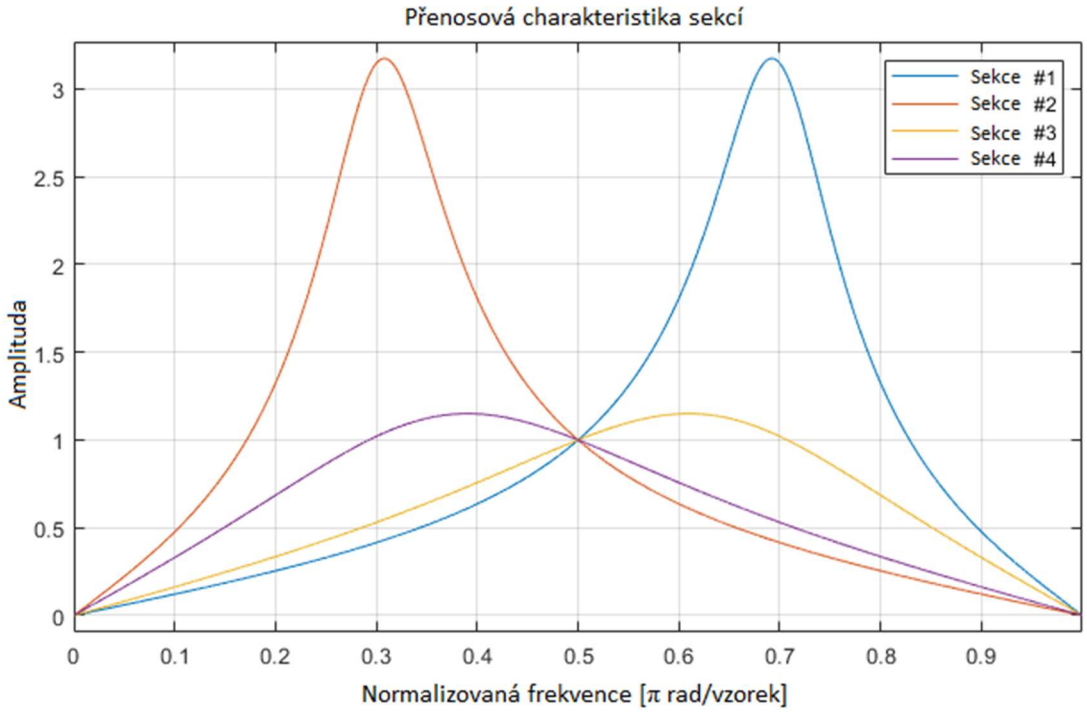
Obrázek 2.6 Přenosová charakteristika pásmových propustí

Čitatele obou filtrů v tabulce 2.3 jsou stejné. V závislosti na mezním kmitočtu se mění pouze jmenovatel a zisk sekcí.

Tabulka 2.3 Vygenerované koeficienty pásmových propustí

1. Filtr (ω_0 = 0,3 – 0,7)							
Sekce	**Čitatel**			**Jmenovatel**			**Zisk**
1.	1	0	-1	1	0,95532	0,68288	0,50329
2.	1	0	-1	1	-0,95532	0,68288	0,50329
3.	1	0	-1	1	0,42345	0,25414	0,42884
4.	1	0	-1	1	-0,42345	0,25414	0,42884
2. Filtr (ω_0 = 0,4 - 0,6)							
1.	1	0	-1	1	0,51960	0,79545	0,27921
2.	1	0	-1	1	-0,51960	0,79545	0,27921
3.	1	0	-1	1	0,19944	0,54419	0,24877
4.	1	0	-1	1	-0,19944	0,54419	0,24877

Obrázek 2.7 ukazuje přenosové charakteristiky jednotlivých sekcí první PP na obrázku 2.6 s ω_0 = 0,3 – 0,7. Stejné závislosti pro DP a HP jsou k dispozici v příloze A (A.1, A.2).



Obrázek 2.7 Přenosová charakteristika sekcí pásmové propusti

2.3 Shrnutí

Filter Designer navrhuje všechny typy filtrů stejným způsobem. Zapojení SOS zůstává stejné a struktura sekcí také. Jediné, co se mezi všemi ukázkovými filtry mění, jsou koeficienty. Koeficienty čitatele se mění na základě typu filtru a koeficienty jmenovatele se mění s mezním kmitočtem. Lze tedy konstatovat, že laditelnosti filtru lze dosáhnout pouze změnou koeficientů bez změny struktury. Při volbě velikosti slov pro výsledky aritmetických operací uvnitř filtru bude třeba brát v potaz maximální zesílení jednotlivých sekcí, jinak bude docházet k saturaci a zkreslení přenosové charakteristiky.

3. NÁVRH LADITELNÉHO FILTRU

V teoretické části bylo ověřeno, že struktura zapojení filtru zůstává stejná nehledě na přenosovou funkci. To umožňuje do FPGA implementovat jedno zapojení a změny přenosové funkce docílit pouze přepisem koeficientů. Ty je tedy nutné někde uložit. Struktura filtru vyžaduje ještě další paměťové prvky, a to pro uložení výsledku první fáze filtru. K uložení výsledků i koeficientů byla zvolena paměť RAM.

Sekvence výpočtů pro jednu sekci SOS struktury je následující:

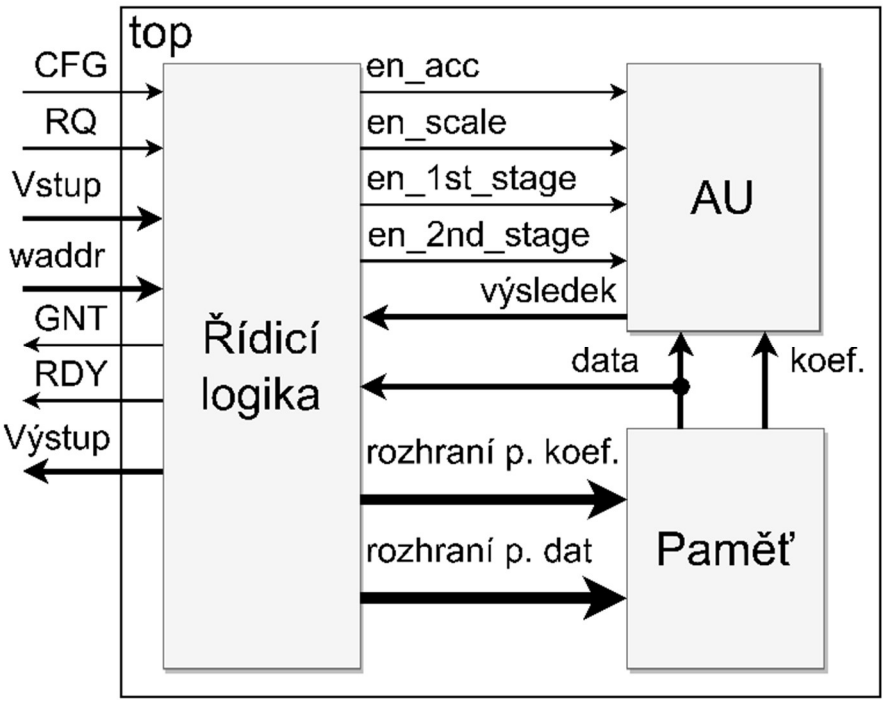
$$\begin{aligned} a_0 \cdot a_1 \cdot x[n] - a_2 \cdot z.v.[n-1] - a_3 \cdot z.v.[n-2] = z.v.; \\ b_1 \cdot z.v. + b_2 \cdot z.v.[n-1] + b_3 \cdot z.v.[n-2] = y[n]; \end{aligned} \tag{3.1}$$

kde a_0 je normovací koeficient (s1 na obrázku 2.2), a_x jsou koeficienty zpětné vazby, b_x jsou koeficienty dopředné vazby, *z.v.[n-i]* reprezentuje výsledky zpětné vazby zpožděné o *i, x[n]* jsou data na vstupu, *z.v.* je výsledek zpětnovazební fáze a *y[n]* je konečný výsledek. Tyto výpočty lze provádět paralelně tak, jak ukazuje zapojení na obrázku 2.2, nebo sériově. Sériový výpočet trvá déle, ale využívá méně zdrojů podle stupně serializace. Při realizaci filtru bylo zvoleno právě zapojení pro sériový výpočet. To, společně s využitím struktury SOS, umožňuje měnit řád přenosové funkce filtru. Kombinací přepisu koeficientů a sériového zapojení je možné měnit řád, typ i mezní kmitočty přenosové funkce při stejném využití zdrojů. Samozřejmě s rostoucím řádem bude lineárně stoupat výpočetní doba.

Funkční princip navrženého filtru spočívá ve vyčtení korespondujících dat – koeficientu a vzorku – z paměti, jejich vynásobení a uložení do akumulátoru. Zapojení lze nalézt na obrázku 3.5. Při návrhu bylo zjištěno, že operace s koeficienty a_1 a b_1 není nutné provádět, jelikož jde ve všech případech o násobení jedničkou, tím se zkrátila výpočetní doba o dva cykly.

Pro jednoduchost návrhu byl zvolen číselný formát s pevnou řádovou čárkou ve dvojkovém doplňku s šířkou slova 16 bitů a 13bitovou frakční částí. Hodnoty koeficientů a výsledků se tedy pohybují v rozsahu <-4; 3.9998779296875>. Vstupní data mají stejný formát, ale nabývají pouze hodnot v intervalu <-1;1>. Tím by mělo být alespoň částečně kompenzováno několikanásobné zesílení frekvencí okolo mezního kmitočtu prvních sekcí SOS struktury, viz. obrázek 2.7 a příloha A (A.1, A.2). Šířka akumulátoru byla ponechána na programu *Filter Designer* a je 34 bitů.

Architektura filtru je rozdělena do tří bloků: Řídicí jednotka, *AU* a Paměť (viz. obrázek 3.1). Blok řídicí jednotky zajišťuje řízení výpočtu, přičemž přistupuje do paměti s daty a koeficienty a odesílá je do aritmetické jednotky. Blok *AU* obsahuje aritmetickou jednotku v pevné řádové čárce, v níž jsou realizovány všechny aritmetické operace s daty a poslední, paměťový blok obsahuje dvě RAM paměti, jednu pro koeficienty a druhou pro data.



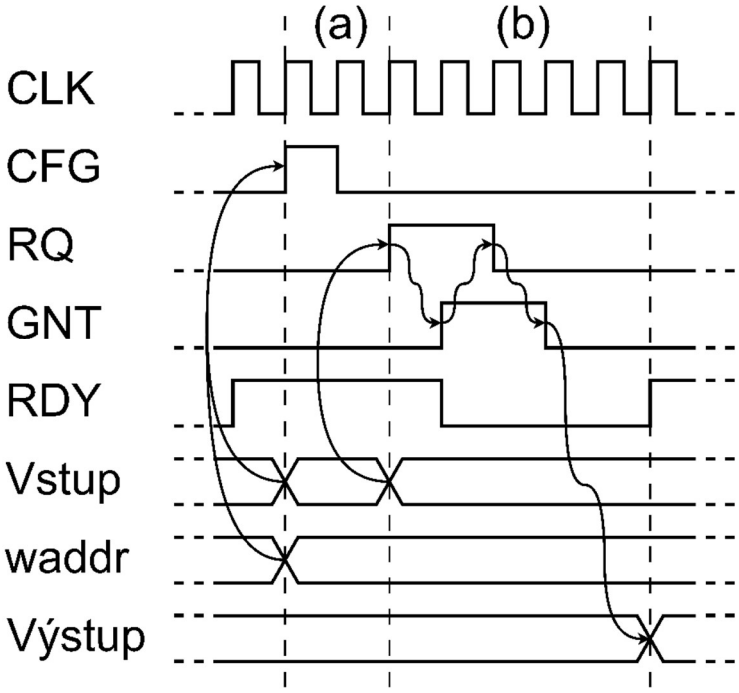
Obrázek 3.1 Blokové schéma architektury

Komunikace se zařízením je umožněna proprietárním rozhraním. Po resetu je zvednut příznak resetu a dochází k inicializaci pamětí. Uživatelský přístup je odepřen. Při inicializaci jsou vstupní data pamětí nastaveny na nulu a spustí se čítače, které postupně projdou všemi adresami. Inicializace končí, když čítače dojdou na konec.

Po inicializaci je nastaven příznak do nuly, signál *RDY* je nastaven do log. 1 a uživateli je umožněn přístup k zápisu do paměti koeficientů a spuštění výpočtu. Zápis koeficientů a spuštění výpočtu je možné jen když je signál *RDY* v log. 1.

Pro zápis do paměti koeficientů je potřeba nastavit data (koeficient) a adresu na korespondující vstupy a zvednout *CFG* do log. 1 alespoň na jeden hodinový cyklus. Zařízení zápis nijak nepotvrzuje. Před výpočtem je potřeba nahrát všechny koeficienty.

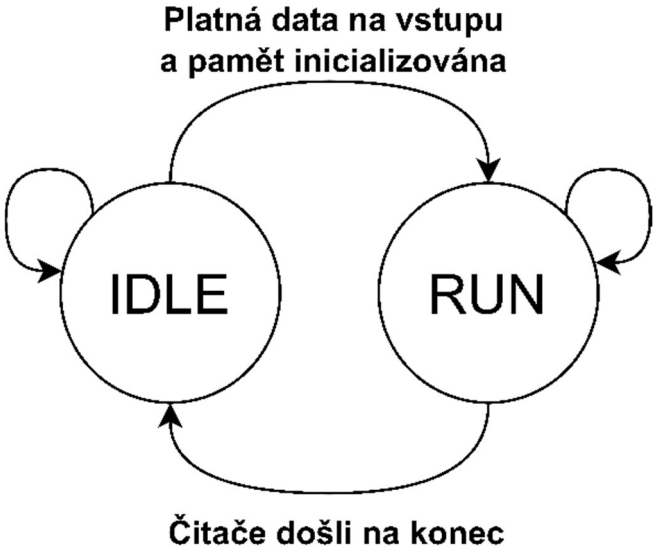
Pro spuštění výpočtu musí uživatel nastavit data na vstup a poté zvednout *RQ* do log. 1. Signál *RDY* padá do log. 0 a signál *GNT* je nastaven do log. 1 indukujíc, že byla data přijata a nastaven zpět do log. 0 poté, co uživatel nastaví *RQ* do log. 0. Jsou spuštěny čítače. Jakmile dojdou čítače na konec, je výsledek poslední sekce přiveden do výstupního registru a signál *RDY* je nastaven do log. 1 indikujíc platná data na výstupu. Signálový tok komunikace lze vidět na obrázku 3.2.



Obrázek 3.2 Signálový diagram zápisu koeficientu (a) a spuštění výpočtu (b)

3.1 Řídicí jednotka

Řídicí jednotka obsahuje stavový automat a dva čítače. Stavový automat (obrázek 3.3) má dva stavy: *idle* (nečinný) a *run* (výpočetní).

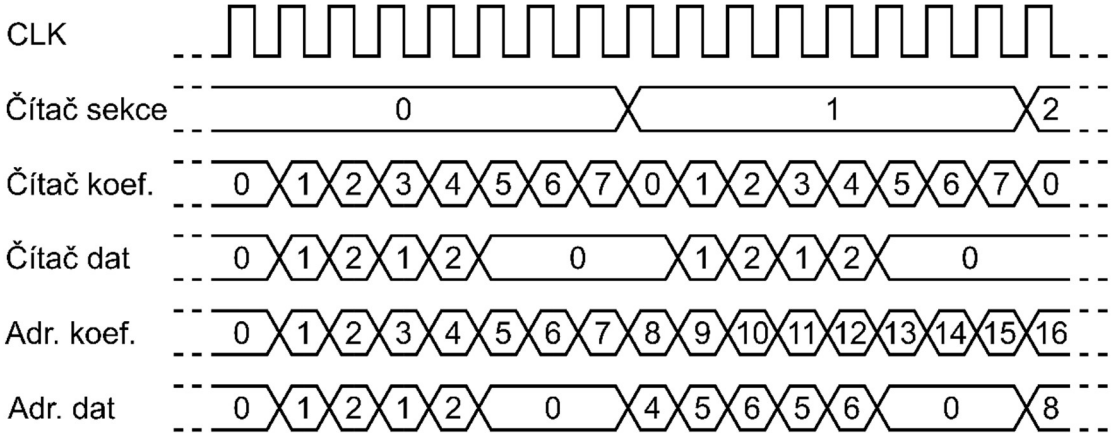


Obrázek 3.3 Stavový diagram řídicí jednotky

Čítače slouží ke generování adres pro přístup do paměti. Aby čítače nevyžadovali logiku navíc, je pro paměť dat a koeficientů každé sekce filtru alokovaný počet adres rovný nejbližšímu vyššímu exponentu dvou (více v kapitole 3.3). Čítač sekcí ovládá horní část adresy, tedy volí adresní rozsah alokovaný pro jednotlivé sekce. Čítač kroků slouží ke krokování operací a ovládání spodní část adresy paměti koeficientů. Pro pročítání paměti dat (spodní části adresy) je dekódována hodnota čítače kroků tak, aby z paměti byla vyčtena korespondující data pro násobení s koeficientem.

Čítač koeficientů zároveň ovládá několik *enable* signálů, které slouží ovládání *AU* a zápisu do pracovních registrů (v závorce je hodnota čítače kroků, kdy je signál aktivní):

a) *en_acc* (<2;6>) – povoluje zápis do akumulátoru, funguje jako reset registru
b) *en_scale* (2) – zaokrouhlení po normování
c) *en_1st_stage* (<3;4>) – odečítání součinů zpětnovazební fáze
d) *en_2nd_stage* (4) – zaokrouhlení po zpětnovazební fázi
e) *en_old_delay* (2) – uloží *z.v.[n-1]* do 1. pracovního registru pro zapsaní do paměti
f) *en_new_delay* (4) – uloží výsledek zpětnovazební fáze do 2. prac. reg. pro zapsání do paměti
g) *en_result* (6) – uloží výsledek sekce do 2. pracovního registru pro zápis do paměti (na vstup další sekce) nebo na výstup

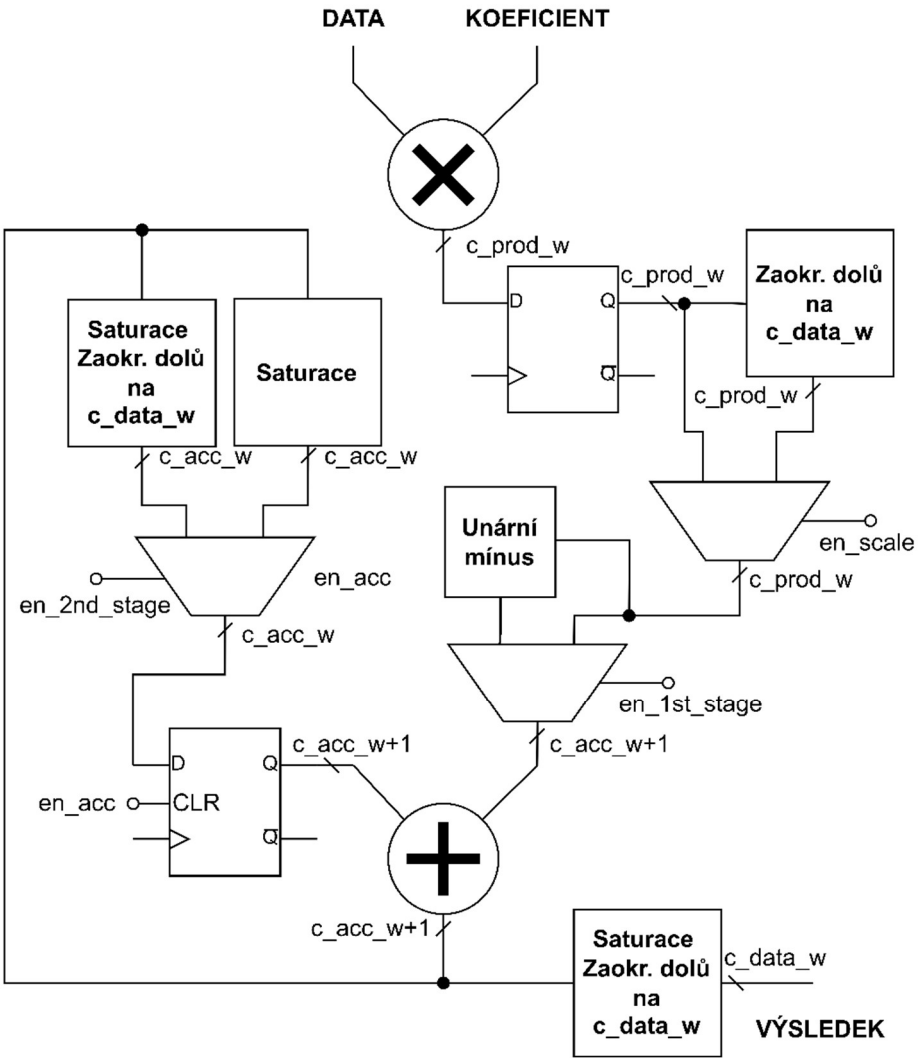


Obrázek 3.4 Signálový diagram hodnot čítačů a adres

Při krocích 5, 6 a 7 jsou postupně do paměti (nebo na výstup) zapisována data v pracovních registrech. Principiálně jde o posunutí zpožděných dat a zápis výsledku na vstup další sekce (nebo na výstup).

3.2 Aritmetická jednotka

Pro serializovaný výpočet je potřeba provádět *Multiply-Accumulate* (MAC) operace, tedy násobení vzorku a koeficientu a jejich následné přičtení k předešlému výsledku. Navrhnutá aritmetická jednotka využívá jedné násobičky a jedné sčítačky (plná serializace). Pro výpočet jedné sekce je nutno provést pět MAC operací. Výsledky normování a zpětnovazební fáze jsou zaokrouhleny na šířku vstupního slova se zachováním řádové čárky. Výsledky zpětnovazební fáze je nutno odčítat. Toho bylo docíleno změnou znaménka (unární mínus). Pokud by mělo dojít k přetečení výsledku sčítání, je hodnota saturována na šířku akumulátoru. Pro snížení kritické cesty byl využit *pipeline* registr za násobičkou. Schéma *AU* lze vidět na obrázku 3.5 . Šířka vstupního slova, násobičky, akumulátoru a frakční části je nastavitelná v knihovně *tuneFilter_pkg.vhd*.

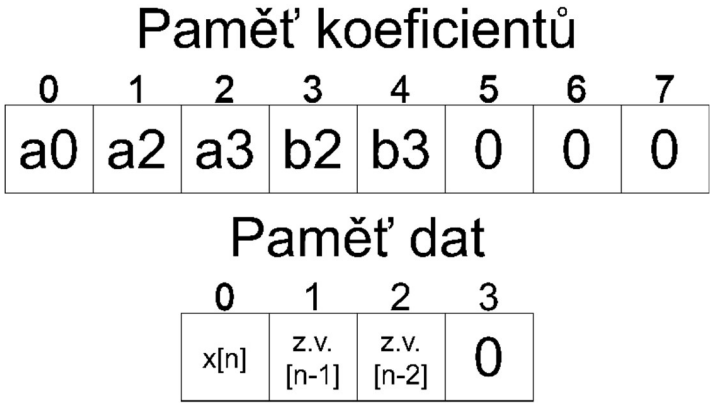


Obrázek 3.5 Schéma aritmetické jednotky

3.3 Paměť

Architektura využívá k ukládání dat a koeficientů blokové paměti SRAM. Pro každou sekci filtru je potřeba uložit 5 koeficientů, 2 vzorky a vstupní data. Jelikož je rychlejší vyčítat koeficient a vzorek naráz, jsou uloženy ve dvou samostatných blocích. Aby byla ušetřena logika čítačů, je pro každou sekci filtru alokován počet buněk rovný nejbližšímu vyššímu exponentu dvojky. Pro koeficienty 8, pro vzorky a vstupní data 4.

Koeficienty jsou uloženy v pořadí: a_0, a_2, a_3, b_2, b_3 a vzorky v pořadí: *x[n], z. v.[n-1], z. v.[n-2]*.



Obrázek 3.6 Pořadí uložených dat

4. POROVNÁNÍ S REFERENČNÍM MODELEM

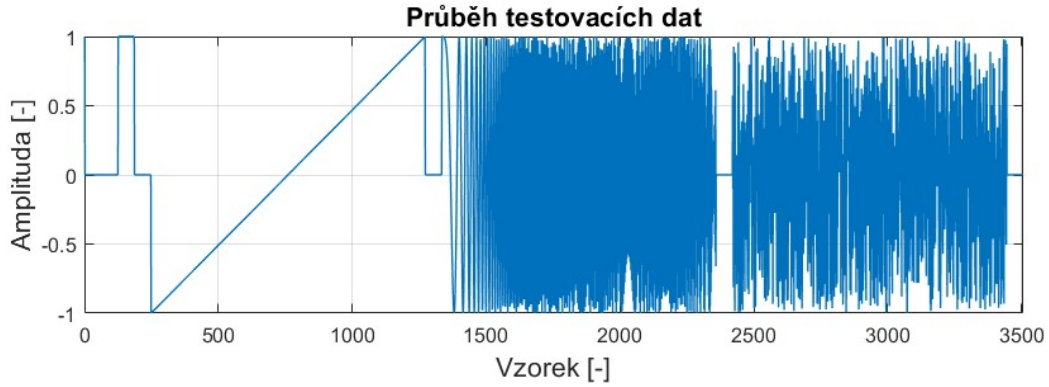
Tato kapitola porovnává realizovaný laditelný filtr s automaticky vygenerovaným HDL popisem programu *Filter Designer* pro případy v kapitole 2.2. Navržené filtry (vlastní i generované) mají následující nastavení číselného formátu, volitelné v kvantizačním panelu nástroje *Filter Designer* a knihovně *tuneFilter_pkg.vhd*:

Tabulka 4.1 Nastavení kvantizačního panelu ve formátu [šířka slova; frakční část] a názvy korespondující názvy proměnných

Číselný formát koeficientů	**Vstup a výstup sekcí**	**Součin**	**Akumulátor**	**Zaokrouhlení**	**Přetečení**
16;13	16;13	32;26	34;26	*Floor*	*Saturate*
c_data_w; c_frac_w	c_data_w; c_frac_w	c_prod_w; c_frac_w*2	c_acc_w; c_frac_w*2		

Pro simulaci byly využity testovací data generována v prostředí *Filter Designer*, na obrázku 4.1 je vidět jejich průběh. Jedná se o sérii standartních signálů v pořadí (od – do):

1. Jednotkový impuls δ[n] (vzorky 1–125)
2. Jednotkový skok u[n] (vzorky 126–148)
3. Pila (vzorky 249–1272)
4. *Chirp*, harmonický průběh s rostoucí frekvencí (vzorky 1335–2420)
5. Bílý šum (vzorky 2421–3506)



Obrázek 4.1 Průběh testovacích dat – δ[n], u[n], pila, *chirp*, bílý šum

4.1 Pomocné m-skripty

Pro vytvoření inicializačního souboru, testovacího vektoru a zpracování výstupních dat filtru byli vytvořeny krátké m-skripty.

4.1.1 Generátor inicializačního souboru

Tento skript slouží k vytvoření inicializačního souboru paměti koeficientů. Uživatel navrhne filtr pomocí programu *Filter Designer*, vyexportuje matici SOS a vektor normovacích koeficientů G do pracovního prostředí MATLAB (*File>Export…>To Workspace, As Coefficients*), uvnitř skriptu nastaví jméno souboru a spustí jej.

4.1.2 Generátor testovacího vektoru

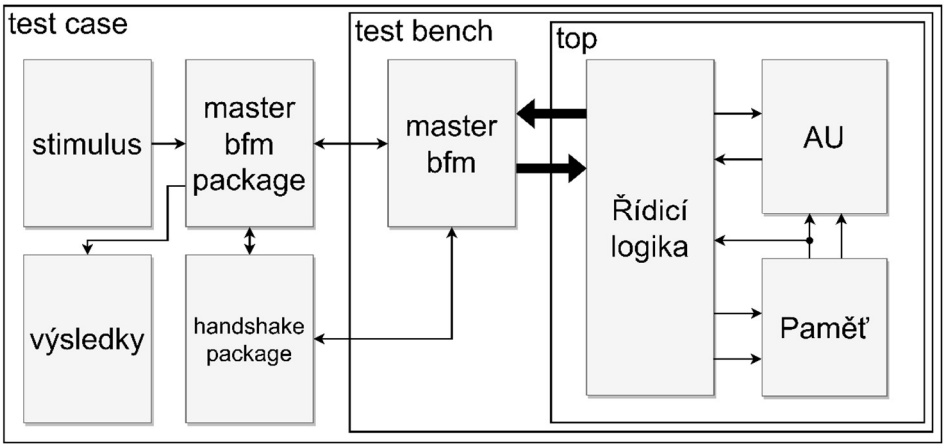
Testovací vektor pro formální verifikaci je získán z knihovny, kterou je možné generovat zároveň s HDL popisem filtru. Hrubá data stimulu, předpokládaných výstupů a jejich délka jsou vyčteny pomocí m-skriptu a uloženy do textového souboru. Uživatel při generování HDL popisu zvolí *Multi-file test bench*, přesune vygenerovanou knihovnu (*filter_tb_data.vhd*) do složky s m-skriptem, nastaví jména souborů a spustí skript.

4.1.3 Skript pro vyhodnocení výsledků

Pro vizuální porovnání výsledků byl vytvořen m-skript, který přečte soubor s výstupními daty a vykreslí graf jejich průběhu společně s referenčními daty a graf jejich rozdílu.

4.2 Verifikační prostředí

Pro funkční verifikaci návrhu byl vytvořen jednoduchý *BFM* (*Bus functional model*). Entita *master bfm* funguje jako virtuální řídící zařízení – ovládá komunikační rozhraní s blokem *top*. Testovací vektor (stimulus) vygenerovaný pomocí programu *Filter Designer* a m-skriptu je čtený knihovnou *master_bfm_pkg.vhd*, která ovládá rozhraní *master bfm*. Funkce z knihovny zároveň čte data, která *master bfm* přijímá a zapisuje je do textového souboru pro pozdější zpracování. Blokové schéma celého prostředí lze vidět na obrázku 4.27.



Obrázek 4.2 Blokové verifikačního prostředí

Funkční verifikace byla provedena formou přímého testu – porovnáním výstupu s HDL popisem programu *Filter Designer*.

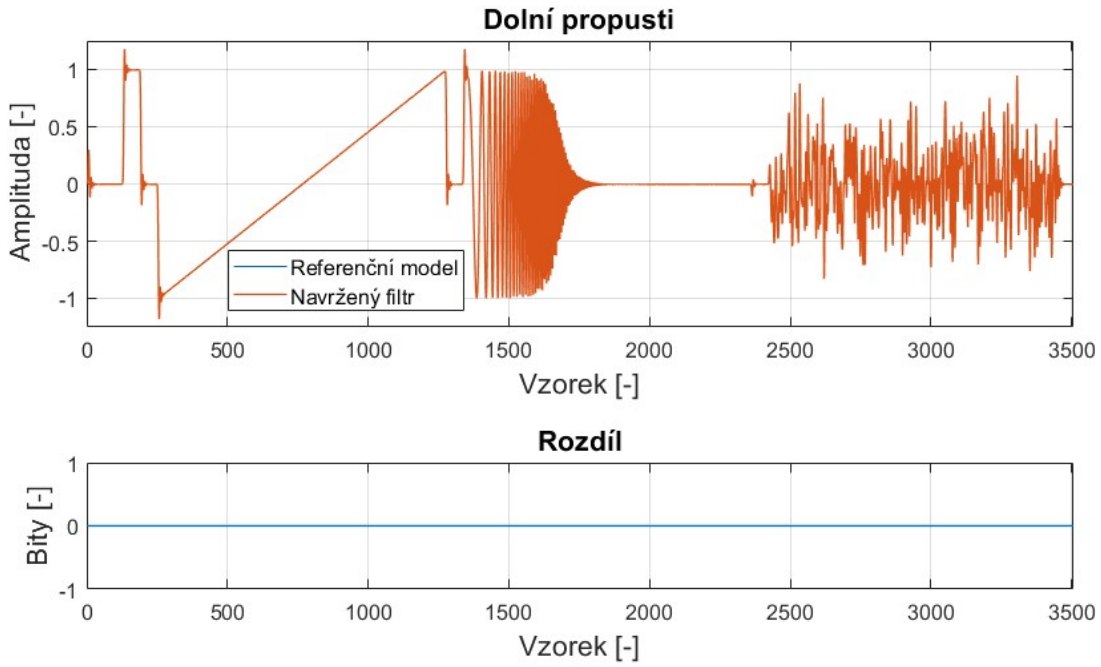
4.3 Výsledky simulace

Simulace byla provedena pro 4 typy filtrů:

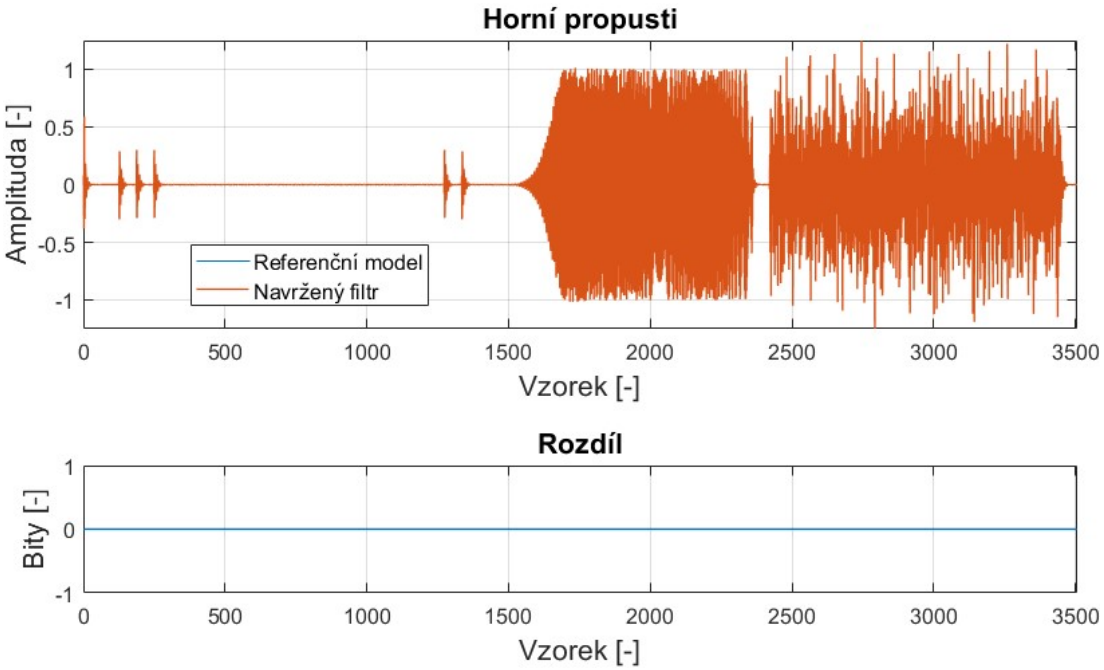
1. Dolní propust, $\omega_0 = 0{,}3$
2. Horní propust, $\omega_0 = 0{,}3$
3. Pásmová propust, $\omega_0 = 0{,}3 – 0{,}7$
4. Pásmová zádrž, $\omega_0 = 0{,}3 – 0{,}7$

Průběhy výsledků navrženého i referenčního filtru 8. řádu lze vidět na grafech 4.32, 4.43, 4.54 a 4.65, společně s rozdílem průběhů v bitech.

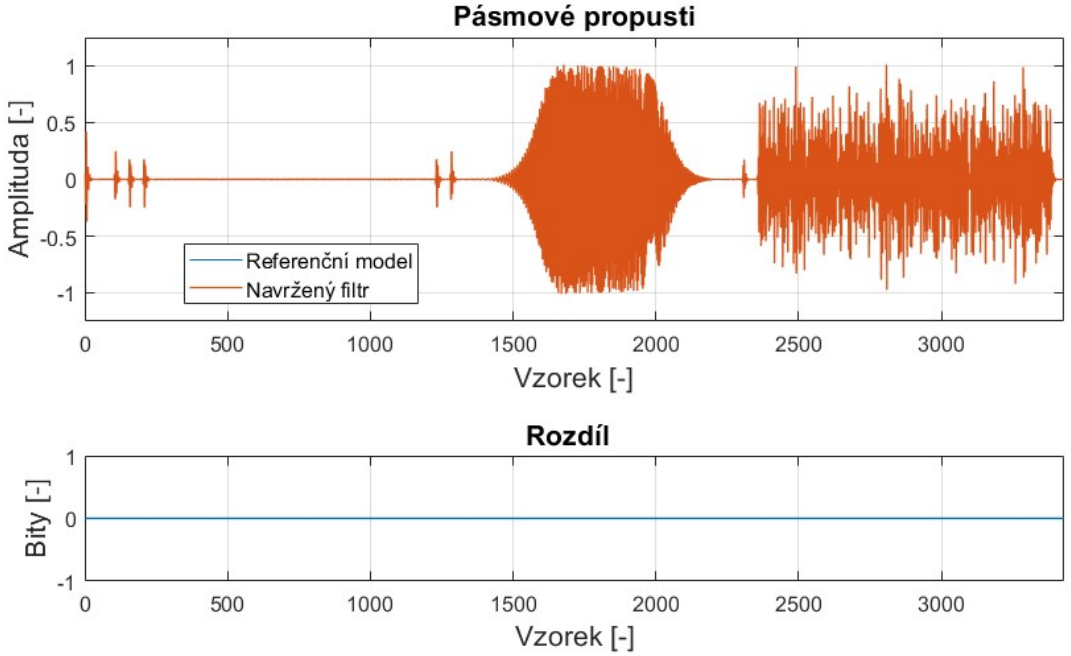
Jak lze vidět, navržený filtr má identické výsledky s referenčním ve všech případech. Na průběhu signálu lze pozorovat chování filtrů. Filtry, které propouští nejnižší frekvence neutlumily průběh pily (který se frekvencí prakticky blíží nule), tlumí vyšší frekvence průběhu signálu *chirp* a snížili amplitudu bílého šumu. Naopak filtry, které propouští nejvyšší frekvence pilovitý průběh odstranily, utlumily nízké frekvence v průběhu signálu *chirp* a amplitudu bílého šumu tolik nesnížily.



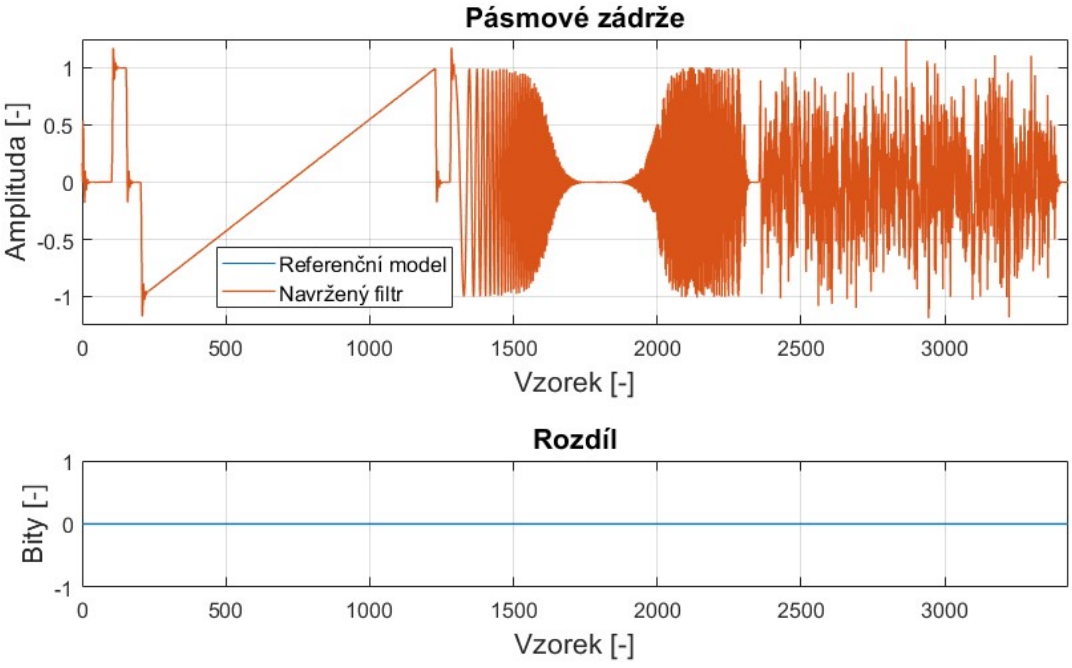
Obrázek 4.3 Grafy výstupů dolních propustí a jejich rozdíl



Obrázek 4.4 Grafy výstupů horních propustí a jejich rozdíl



Obrázek 4.5 Grafy výstupů pásmových propustí a jejich rozdíl



Obrázek 4.6 Grafy výstupů pásmových zádrží a jejich rozdíl

5. IMPLEMENTACE NAVRŽENÉHO FILTRU

Jako cílové zařízení pro implementaci bylo zvoleno FPGA Artix-7 (xc7a35tcpg236-1) od firmy Xilinx, vzhledem k dostupnosti vývojové desky Basys 3, která jej využívá. Pole nabízí 20800 náhledových tabulek (LUT), 41600 klopných obvodů (FF), 1800 kbit blokové paměti RAM a 90 signálových procesorů DSP48E1. K syntéze VHDL kódu pro toto FPGA byl využitý program Vivado (verze 2022.1). [9]

Architektura využívá synchronní reset, aby se předešlo poškození dat v paměti RAM, a protože RAM paměti a DSP48E1 v použitém FPGA nevyužívají asynchronní reset. Všechny signály, kromě příznaku resetu, jsou po resetu nastaveny do log. 0 [10]

Vlastní navržený filtr byl porovnán s referenčním modelem v rámci využitých zdrojů po implementaci pro 8. a 20. řád. U obou řádů byla provedena implementace se snahou docílit nejnižší možné periody hodinového taktu, tedy nejvyšší frekvence. Vzhledem k tomu, že bylo využito základní strategie pro syntézu a implementaci a hodnota periody byla nastavována ručně, jsou hodnoty pouze orientační.

Tabulka 5.1 Tabulka využitých zdrojů po implementaci a maximální frekvence

Filtr	**Navržený filtr**		**Referenční model**	
Řád	**8.**	**20.**	**8.**	**20.**
LUT	181	183	292	651
Registry	93	95	311	704
Bloková RAM	1	1	0	0
DSP	1	1	1	1
F7 Mux	0	0	64	143
F8 Mux	0	0	16	66
T_{MIN} [ns]	7,250	7,250	18,547	20,003
f_{MAX} [MHz]	137,9	137,9	53,9	49,9

Navržený filtr využívá výrazně méně zdrojů a s rostoucím řádem se mění pouze počet využitých registrů. Nárůst počtu využitých registrů u instance 20. řádu je způsobený větší pamětí pro ukládání koeficientů a dat sekcí. Využité registry slouží k rozšíření rozsahu čítače sekcí tak, aby obsáhl všechny adresy.

Díky využití *pipeline* registru za násobičkou a optimalizací popisu bylo docíleno nižší kritické cesty než u referenčních modelů. Nejdelší kritická cesta navrženého filtru tak je mezi výstupem registru paměti RAM a výstupem *pipeline* registru násobičky. Bylo tak docíleno výrazně vyšší maximální frekvence hodinového taktu. Navržený filtr vypočte výsledek jedné sekce za 8 hodinových taktů a referenční model za 6. Zpoždění dvou taktů je způsobeno výstupními registry pamětí RAM a *pipeline* registrem násobení. Navržený filtr tedy provede výpočet jedné sekce rychleji i přes 2taktové zpoždění.

6. ZÁVĚR

V teoretické části bylo uvedeno seznámení s tématikou číslicových filtrů a problematikou jejich návrhu. Při návrhu filtrů může být překážkou náročnější matematická analýza, proto je vhodné v rámci zrychlení procesu návrhu použít nástroje k tomu uzpůsobené. Čtenář byl seznámen se základními typy struktur IIR filtrů a kaskádovou strukturou *Second-Order Sections*. Struktura SOS je vhodná pro hardwarovou implementaci díky lepším vlastnostem v rámci stability a kvantizačních chyb. Číselný formát s pevnou řádovou čárkou umožňuje snadnější implementaci za cenu kvantizačních chyb (přesnosti).

Byly navrženy vzorové IIR filtry třech typů, pro každý typ po dvou mezních kmitočtech. Bylo zjištěno, že při změně typu nebo mezního kmitočtu nedochází ke změně zapojení a filtry se liší pouze hodnotou koeficientů. Jejich změnou, je tedy možné docílit laditelnosti beze změny zapojení.

Cílem této práce byl návrh a verifikace číslicového filtru v jazyce VHDL. V rámci této práce byl navržený koncept laditelného IIR filtru, který využívá RAM paměti pro ukládání koeficientů a dat. Přepsáním koeficientů lze změnit přenosovou funkci filtru jak v rámci typu, tak mezního kmitočtu. Volbou plně serializovaného výpočtu bylo umožněno měnit i maximální řád filtru s minimální změnou využitých zdrojů (tabulka 5.1 Tabulka).

Architektura zařízení byla rozdělena do tří bloků: řídicí logika, *AU* a paměť. Blok řídicí logiky obsahuje stavový automat a dva čítače. Stavový automat má dva stavy: *idle* (nečinný) a *run* (výpočetní). Čítače slouží k přístupu do paměti a generování řídicích signálu aritmetické jednotky. Aritmetická jednotka obsahuje jednu násobičku, jeden akumulátor a logiku pro saturaci a zaokrouhlení.

Pro verifikaci byl vytvořen BFM společně s jednoduchým verifikačním prostředím. Ověření návrhu bylo provedeno funkční verifikací testovacími daty vygenerovanými pomoci nástroje *Filter Designer* a m-skriptů. Testovací data obsahovala standartní signály δ[n], u[n], pila, *chirp* a bílý šum. Testovaná instance filtru byla 8. řádu a její výstupní data byla porovnána referenčními modely vygenerovanými programem *Filter Designer*. Výstupy navrženého filtru a referenčního modelu byly identické.

Návrh byl implementován v instanci 8. a 20. řádu pro FPGA Artix-7 a porovnán s referenčními modely stejných řádů. Navržený filtr využívá už při 8. řádu výrazně méně zdrojů – méně než dvě třetiny LUT a třetinu registrů. Zároveň dosahuje více než dvojnásobné maximální frekvence hodinového taktu.

Po implementaci lze řád filtru pouze snižovat vypnutím sekce nahráním jednotkového normovacího koeficientu a ostatních nulových. Tímto se samozřejmě nezmění výpočetní čas a budou prováděny prázdně operace. Změny řádu po implementaci by šlo docílit nahráním informace o řádu filtru. Nejmenší instance blokové RAM je 18kbit, což je více než dostačující pro ukládání dat. Čítač sekci by tedy doplňoval šířku

adresy pro celý rozsah bloku paměti a jeho maximální hodnota by se řídila hodnotou nahranou v registru. Tímto by tedy šlo řád filtru snižovat i zvyšovat za běhu bez provádění prázdných operací.

LITERATURA

[1] SMÉKAL, Zdeněk. *Analýza signálů a soustav - BASS*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav komunikací, 2012. ISBN 978-80-214-4453-9.

[2] LYONS, Richard G. *Understanding digital signal processing*. 3rd ed. Upper Saddle River: Prentice Hall, 2011. ISBN 978-0-13-702741-5.

[3] KONÍČEK, C. Zpracování signálu z digitálních mikrofonů typu MEMS. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2012. 37 s. Vedoucí bakalářské práce doc. Ing. Lukáš Fujcik, Ph.D..

[4] SMITH, Julius O. *Introduction to Digital Filters with Audio Applications* [online book]. Stanford, CA, 2007 [cit. 2022-11-25]. Dostupné z: http://ccrma.stanford.edu/~jos/filters/

[5] *What is MATLAB?* [online]. MathWorks [cit. 2022-11-27]. Dostupné z: https://www.mathworks.com/discovery/what-is-matlab.html#:~:text=Millions%20of%20engineers%20and%20scientists,computational%20finance%2C%20and%20computational%20biology.

[6] VÍCH, Robert a Zdeněk SMÉKAL. Číslicové filtry. 1. vyd. Praha: Academia, 2000, 218 s. ISBN 80-200-0761-X.

[7] Discrete-time, second-order section, direct-form II filter. *MathWorks* [online]. MathWorks [cit. 2022-12-04]. Dostupné z: https://www.mathworks.com/help/releases/R2022b/signal/ref/dfilt.df2sos.html

[8] *Use Filter Designer with DSP System Toolbox Software* [online]. MathWorks, 2022 [cit. 2022-12-04]. Dostupné z: https://www.mathworks.com/help/dsp/ug/use-fdatool-with-dsp-system-toolbox-software.html

[9] *Basys 3™ FPGA Board Reference Manual.* Rev. C. 1300 Henley Court, Pullman, WA 99163, 2016. Dostupné také z: https://digilent.com/reference/_media/basys3:basys3_rm.pdf

[10] *Vivado Design Suite User Guide: UG901*. V2022.2. San Jose, 2100 Logic Dr, United States, 2022. Dostupné také z: https://www.xilinx.com/content/dam/xilinx/support/documents/sw_manuals/xilinx2022_2/ug901-vivado-synthesis.pdf

[11] *Artix-7 FPGAs Data Sheet: DC and AC Switching Characteristics: DS181*. V1.27. San Jose, 2100 Logic Dr, United States, 2022. Dostupné také z: https://docs.xilinx.com/v/u/en-US/ds181_Artix_7_Data_Sheet

[12] *7 Series DSP48E1 Slice: UG479*. V1.10. San Jose, 2100 Logic Dr, United States, 2018. Dostupné také z: https://docs.xilinx.com/v/u/en-US/ug479_7Series_DSP48E1

[13] *VHDLwhiz* [online]. c/o Liv Anne Gunvaldsen, Kvernhusheia 46B, 4634 KRISTIANSAND S: JENSEN TECH, 2023 [cit. 2023-05-24]. Dostupné z: https://vhdlwhiz.com/

SEZNAM SYMBOLŮ A ZKRATEK

Zkratky:

AU	Arithmetic Unit
DP	Dolní propust
DSP	Digital Signal Processing
FF	Flip-Flop
FIR	Finite Impulse Response
FPGA	Field-Programmable Gate Array
HP	Horní propust
IIR	Infinite Impulse Response
LSB	Least significant bit
LUT	Look-Up Table
MAC	Multiply-Accumulate
PP	Pásmová propust
RAM	Random Access Memory
SOS	Second-Order Sections

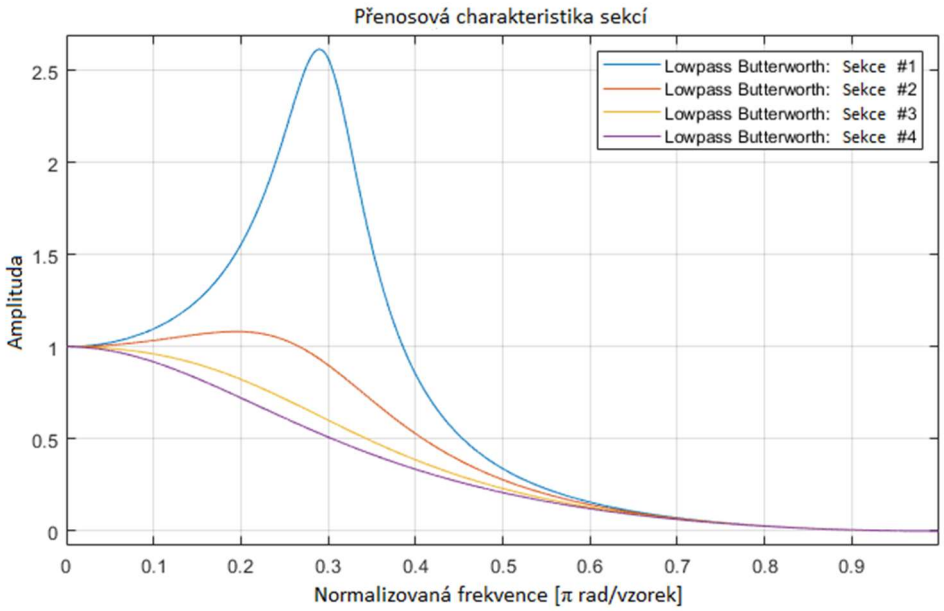
Symboly:

ω	normalizovaný úhlový kmitočet	[πrad/vz.]

SEZNAM PŘÍLOH

Příloha A - Přenosové charakteristiky sekcí DP a HP

A.1 Přenosová charakteristika sekcí DP ($\omega_0 = 0{,}3\pi$)



A.2 Přenosová charakteristika sekcí HP ($\omega_0 = 0{,}3\pi$)