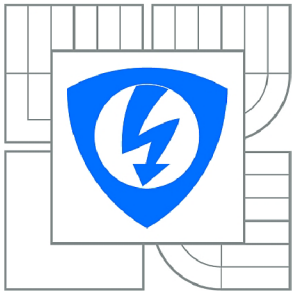
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV MIKROELEKTRONIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF MICROELECTRONICS

NÁVRH DIGITÁLNĚ-ANALOGOVÉHO PŘEVODNÍKU TYPU SIGMA-DELTA V TECHNOLOGII CMOS

DESIGN OF SIGMA-DELTA DIGITAL-TO-ANALOG CONVERTER IN CMOS TECHNOLOGY

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

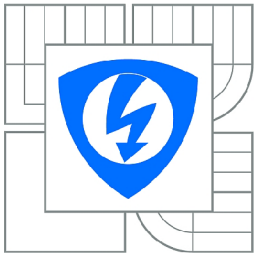
AUTOR PRÁCE — Bc. LUDĚK SOUKUP

AUTHOR

VEDOUCÍ PRÁCE — doc. Ing. LUKÁŠ FUJCIK, Ph.D.

SUPERVISOR

BRNO 2012



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav mikroelektroniky

Diplomová práce

magisterský navazující studijní obor
Mikroelektronika

Student:	Bc. Luděk Soukup	***ID:***	72888
Ročník:	2	***Akademický rok:***	2011/2012

NÁZEV TÉMATU:

Návrh digitálně-analogového převodníku typu sigma-delta v technologii CMOS

POKYNY PRO VYPRACOVÁNÍ:

Cílem diplomové práce je navrhnout digitálně-analogový (DA) převodník sigma-delta v technologii CMOS. V rámci semestrálního projektu se student zaměří na problematiku návrhu interpolačních filtrů a jejich návrh a modelování v prostředí Matlab. Student následně vytvoří model celého řetězce s ohledem na dosažení požadovaných parametrů zadaných vedoucím. V diplomové práci student popíše veškeré dílčí bloky převodníku DA v jazyce HDL, provede syntézu a analýzu navrženého převodníku v prostředí Cadence RTL Compiler. Výsledky vyhodnotí a porovná s navrženým modelem.

DOPORUČENÁ LITERATURA:

Dle pokynů vedoucího práce

Termín zadání:	6.2.2012	***Termín odevzdání:***	24.5.2012

Vedoucí práce: doc. Ing. Lukáš Fujcik, Ph.D.
Konzultanti diplomové práce:

prof. Ing. Vladislav Musil, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

Abstrakt

Tato diplomová práce se zabývá problematikou digitálně-analogového převodu a možnostmi jeho realizace prostřednictvím digitálních obvodových struktur. Cílem je navrhnout digitálně-analogový převodník typu sigma-delta s rozlišením 14 bitů a pracovním kmitočtovým pásmem (0 ÷ 20) kHz, jehož hlavní funkční bloky, tedy interpolátor a modulátor sigma-delta, budou realizovány jako digitální obvodové struktury. Rekonstrukční filtr bude realizován jako analogová obvodová struktura. Pro návrh a následné ověření vlastností navrženého převodníku budou použity programy MATLAB a Simulink. Navržené digitální struktury budou popsány v jazyce VHDL.

Klíčová slova

Číslicové zpracování signálu, interpolace, integrované obvody, aritmetické operace, převodník sigma-delta, digitálně-analogový převodník

Abstract

This master's thesis deals with the issue of digital to analog conversion and possibility of its realization in digital circuits. Goal of this project is to design sigma-delta digital to analog converter with resolution of 14 bits and frequency band (0 ÷ 20) kHz. Main functional blocks: interpolator and modulator sigma-delta will be realized like digital structures. Reconstruction filter will be realized like an analog structure. For design a check of parameters of designed converter programs MATLAB and Simulink are used. Designed digital structures will be described by VHDL language.

Key words

Digital signal processing, interpolation, integrated circuits, arithmetic operations, sigma-delta converter, digital to analog converter

Bibliografická citace

SOUKUP, L. Návrh digitálně-analogového převodníku typu sigma-delta v technologii CMOS. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2012. 51 s. Vedoucí semestrální práce doc. Ing. Lukáš Fujcik, Ph.D..

Prohlášení autora o původnosti díla:

Prohlašuji, že svoji diplomovou práci na téma Návrh digitálně-analogového převodníku typu sigma-delta v technologii CMOS jsem vypracoval samostatně pod vedením vedoucího diplomové 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 diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové 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 § 152 trestního zákona č. 140/1961 Sb.

V Brně dne 24. května 2012 .....

Autor

Poděkování

Děkuji vedoucímu diplomové práce doc. Ing. Lukáši Fujcikovi, Ph.D. za cenné odborné rady a účinnou pedagogickou pomoc při zpracování této diplomové práce.

V Brně dne 24. května 2012 .....

Autor

Obsah

ÚVOD ... 1

1 NÁVRH A REALIZACE DIGITÁLNÍCH SYSTÉMŮ ... 2

1.1 NÁVRH DIGITÁLNÍCH SYSTÉMŮ ... 2

1.1.1 Jazyk VHDL ... 2

1.1.2 Proces návrhu digitálních systémů ... 3

1.2 ZPŮSOBY REALIZACE ... 4

1.2.1 Obvody ASSP ... 4

1.2.2 Obvody FPGA ... 5

1.2.3 Obvody ASIC ... 5

2 TEORIE PŘEVODU SIGNÁLU ... 6

2.1 SIGNÁLY ANALOGOVÉ A DIGITÁLNÍ ... 6

2.2 TEORIE DIGITÁLNĚ-ANALOGOVÉHO PŘEVODU ... 10

2.3 VLASTNOSTI D/A PŘEVODNÍKŮ ... 11

2.3.1 Statické vlastnosti ... 11

2.3.2 Dynamické vlastnosti ... 14

2.3.3 Další vlastnosti ... 15

2.4 D/A PŘEVODNÍKY TYPU SIGMA-DELTA ... 16

2.4.1 Interpolátor ... 16

2.4.2 Digitální modulátor sigma-delta ... 18

2.4.3 Analogový rekonstrukční filtr ... 19

3 NÁVRH A SIMULACE MODELU PŘEVODNÍKU ... 20

3.1 ANALÝZA ZADÁNÍ ... 20

3.2 INTERPOLÁTOR ... 22

3.3 MODULÁTOR SIGMA-DELTA ... 29

3.4 ANALOGOVÝ REKONSTRUKČNÍ FILTR ... 36

4 POPIS PŘEVODNÍKU V JAZYCE VHDL ... 37

4.1 INTERPOLÁTOR ... 37

4.2 MODULÁTOR SIGMA DELTA ... 40

4.3 SPOJENÍ INTERPOLÁTORU A MODULÁTORU ... 42

ZÁVĚR ... 43

SEZNAM POUŽITÝCH ZKRATEK A SYMBOLŮ ... 44

SEZNAM POUŽITÝCH INFORMAČNÍCH ZDROJŮ ... 45

SEZNAM PŘÍLOH ... 46

Úvod

Digitální systémy dnes nacházejí stále širší uplatnění napříč celým aplikačním spektrem moderní elektroniky. Oproti analogovým systémům vykazují nižší citlivost vůči šumu a rušení, obsaženém ve zpracovávaných signálech, a také nižší citlivost vůči výrobním tolerancím. Dále umožňují realizaci komplexnějších systémů, přičemž při implementaci je možné dosáhnout vyššího stupně integrace. Nezanedbatelnou výhodou je také možnost vytvářet univerzální systémy řízené programem. Veškeré přednosti digitálních systémů by však nebylo možné využívat, pokud bychom nebyli schopni zajistit jejich propojení a kompatibilitu s okolím, především s analogovými systémy a subsystémy.

K zajištění kompatibility mezi analogovými a digitálními systémy slouží zvláštní třída elektronických obvodů, označovaná jako převodníky signálu. Platí přitom, že vlastnosti celého systému jsou do značné míry determinovány právě jejich vlastnostmi. V závislosti na typu převodu rozlišujeme analogově-digitální (A/D) a digitálně-analogové (D/A) převodníky. Tato práce je zaměřena především na problematiku převodu D/A, ačkoli jsou zde rozebírány i některé aspekty převodu A/D. To vychází z úzkého provázání těchto problematik.

Hlavním cílem této práce je navrhnout a ověřit digitální obvodovou strukturu realizující funkci digitálně-analogového převodníku typu sigma-delta s pracovní frekvencí 0÷20 kHz a rozlišením 14 bitů. Tato struktura bude následně popsána v jazyce VHDL. Výstupní filtr, který je nedílnou součástí převodníku, bude z fyzikálních důvodů navržen separátně jako analogová obvodová struktura.

První kapitola je věnována stručnému nástinu návrhového procesu digitálních systémů a jejich realizačních možností. Druhá kapitola je zaměřena na problematiku převodu signálu a v jejím rámci dochází ke stručnému shrnutí nejdůležitějších faktů, pojmů a principů, které jsou s tímto tématem spojeny. Třetí kapitola je věnována vlastnímu návrhu a ověření digitálního systému v prostředí MATLAB. Čtvrtá kapitola se zabývá popisem navržených digitálních struktur v jazyce VHDL.

1 Návrh a realizace digitálních systémů

1.1 Návrh digitálních systémů

Proces návrhu digitálního systému je poměrně složitou záležitostí. Nezastupitelnou roli přitom hraje počítačová podpora návrhu - systémy třídy CAD, které díky schopnosti automatizovat dílčí části návrhových procedur, celý proces návrhu výrazně urychlují. Volba vývojového nástroje závisí na zvolené obvodové platformě, obecně je však k dispozici velké množství vývojových nástrojů jak od výrobců cílového hardwaru (např. Xilinx, nebo Altera) tak od renomovaných firem zaměřených na vývoj softwaru pro elektrotechniku (např. Cadence, či Mentor Graphics). Ve spojitosti s těmito vývojovými systémy je třeba zmínit také jazyk VHDL, který společně s Verilogem patří mezi nejrozšířenější jazyky pro popis digitálních systémů.

1.1.1 Jazyk VHDL

VHDL je programovací jazyk vysoké úrovně abstrakce, který byl vytvořen pro popis a simulaci rozsáhlých digitálních systémů. Svojí povahou spadá do třídy jazyků určených k popisu obvodových struktur, v anglicky psané literatuře označovaných zkratkou HDL (Hardware Description Language). Jazyk VHDL je charakteristický bohatou vyjadřovací schopností a snadnou přenositelností zdrojových kódu mezi různými cílovými technologiemi. Ačkoli o VHDL hovoříme jako o programovacím jazyku, je třeba si uvědomit, že metodika práce s tímto jazykem, je do značné míry odlišná od klasického programování v jazycích, jako jsou Assembler, C, nebo Java. Vytvořený zdrojový kód totiž nepředstavuje posloupnost příkazů prováděnou procesorem, ale obvodovou strukturu, která je následně realizována například v obvodech FPGA. Dále je třeba si uvědomit, že některé elementy jazyka jsou určeny pouze pro simulaci, jejich implementace do cílového obvodu se nejen nepředpokládá, ale dokonce není možná [1].

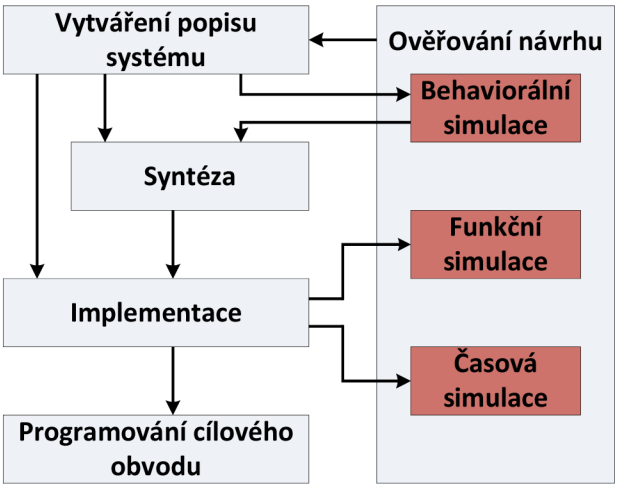
Historie jazyka VHDL je spojena s výzkumným projektem VHSIC (Very High Speed Integrated Circuits) ministerstva obrany Spojených států amerických, který byl zahájen v roce 1981. Vývoj jazyka probíhal v letech 1983 až 1985 ve spolupráci firem IBM, Intermetrics a Texas Instruments. Výsledkem byla základní definice jazyka VHDL (VHSIC Hardware Description Language), která vycházela z jazyka ADA. V roce 1986 byl vývoj předán organizaci IEEE a v roce 1987 byl publikován první standard jazyka pod označením IEEE Std. 1076-1987. Tato verze jazyka bývá zkráceně označována jako VHDL-87 [1].

Do současnosti prošel jazyk VHDL čtyřmi revizemi. První a zároveň nejrozsáhlejší z nich (VHDL-93) byla provedena v roce 1993 a publikována o rok později pod názvem IEEE Std. 1076-1993. Následovaly revize v letech 2000 a 2002, ty však nepřinesly mnoho významných změn a proto nedošlo k jejich širší implementaci do vývojových nástrojů. Poslední revize (VHDL-2008) byla provedena v roce 2008 a publikována pod názvem IEEE Std. 1076-

2008. Ani tato revize nepřináší do standardu jazyka VHDL zásadní změny. Vzhledem ke skutečnosti, že doposud nejrozšířenější revize jazyka VHDL-93, byla vydána již před dlouhými devatenácti lety, lze ovšem její rozšíření předpokládat [1].

1.1.2 Proces návrhu digitálních systémů

Proces návrhu, neboli „Design Flow“, představuje posloupnost kroků, vedoucích ke vzniku digitálního systému. Podoba návrhového procesu není fixní, záleží na cílové technologii, tedy typu obvodů použitých k realizaci navrhovaného systému. V této práci bude věnována pozornost jednak návrhovému procesu programovatelných hradlových polí FPGA, jednak návrhovému procesu zákaznických obvodů ASIC.

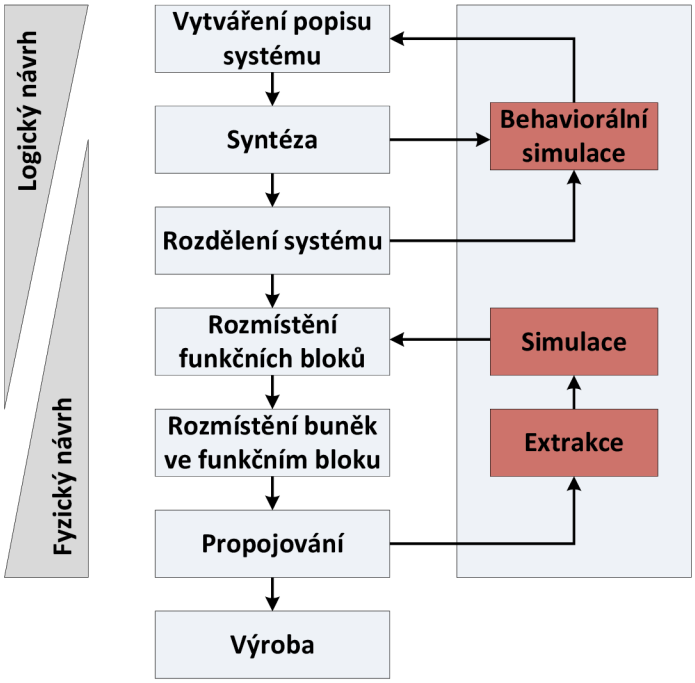


Obrázek 1 - proces návrhu digitálního systému pro FPGA [2]

Diagram mapující návrhový proces pro FPGA zachycuje Obrázek 1. Celý proces začíná vytvářením popisu systému („Design Entry”). K tomuto účelu může být použit schematický editor, nebo lépe, některý z jazyků třídy HDL. Druhým krokem je syntéza („Synthesis”). Jedná se o proces převádění popisu systému na obvodovou strukturu RTL „Register Transfer Logic“, která je nezávislá na cílové technologii. Následuje behaviorální simulace („Behavioral Simulation”). Jejím cílem je ověřit správnost funkce navrhovaného systému. Čtvrtým krokem je implementace („Implementation“), v jejímž průběhu jsou vytvářena konfigurační data pro zvolené FPGA („Bitstream“). Implementace zahrnuje proces mapování („Map“), kdy je systém rozdělen na dílčí sub-bloky odpovídající použitému obvodu FPGA, a proces rozmístění a propojení („Place and Route“), kdy jsou sub-blokům vytvořeným při mapování přiřazeny konkrétní struktury v cílovém obvodu a tyto struktury jsou propojeny. Dále následují funkční a časové simulace. Pokud návrh vyhoví, může být přistoupeno k poslednímu kroku, kterým je programování cílového obvodu [2].

Příklad procesu návrhu zákaznických obvodů ASIC zachycuje Obrázek 2. Obdobně jako u FPGA, celý proces začíná vytvářením popisu systému („Design Entry”) a Syntézou („Synthesis”). Třetím krokem je rozdělení systému do dílčích bloků („System Partitioning”). Následuje behaviorální simulace („Behavioral Simulation”). V případě negativního výsledku

dochází k opakování prvních tří kroků. V případě pozitivního výsledku následuje pátý krok, kterým je rozmístění funkčních bloků na čipu („Floor Planning"). Šestý krok, kterým je rozmístění jednotlivých logických buněk ve funkčním bloku („Placement") a sedmý krok, kterým je propojování logických buněk („Routing"). Následuje extrakce parametrů navrženého obvodu, v jejímž rámci jsou stanoveny parazitní vlivy působící na jednotlivá metalová propojení („Circuit Extraction"). Tyto parametry jsou využívány pro simulace ověřující vlastnosti navrženého obvodu („Post Layout Simulation"). Pokud obvod nevyhoví, je třeba zopakovat kroky pět až devět. Pokud obvod vyhoví, může být zahájena výroba testovací série [3].



Obrázek 2 - proces návrhu zákaznických obvodů ASIC [3]

1.2 Způsoby realizace

Při návrhu digitálního systému hraje velmi významnou roli volba způsobu realizace, potažmo volba cílové technologie. Nejčastěji jsou pro realizaci digitálních systémů využívány tři základní třídy obvodů: standardizované obvody ASSP, programovatelné obvody PLD, respektive podskupina FPGA, a zákaznické obvody ASIC. Rozdíly mezi těmito třídami obvodů jsou jednak ve funkci a výkonu, jednak v ceně a době nutné k vývoji, či výrobě. Obsah této podkapitoly vychází z pramene [3].

1.2.1 Obvody ASSP

Termín ASSP je akronymem anglických slov „Application Specific Standard Product". Jedná se o označení třídy obvodů, které jsou navrženy pro určitou aplikaci, na rozdíl od zákaznických obvodů, jsou však vyráběny pro trh a jsou tedy široce dostupné pro elektrotechnický průmysl. Typickým příkladem obvodů ASSP jsou paměti RAM, mikroprocesory, nebo třeba obvody řad 4000 a 7400.

Obvody ASSP jsou obvykle vyráběny ve velkých sériích, což umožňuje dosažení nízké ceny. Vznikne-li při návrhu digitálního systému potřeba vytvořit strukturu s funkcí odpovídající některému z obvodů ASSP, je třeba zvážit, zda nebude výhodnější použít právě tento obvod. Pro obvody ASSP hovoří nízká cena a nulový čas nutný k návrhu, vlastní návrh naopak umožňuje optimalizaci obvodu vzhledem k výkonu, spotřebě, nebo například komunikačnímu protokolu.

1.2.2 Obvody FPGA

Označení FPGA vychází z anglického pojmenování těchto obvodů „Field Programmable Gate Array", v češtině se ustálil termín programovatelná hradlová pole. Tyto obvody patří do rodiny programovatelných logických obvodů označovaných anglickou zkratkou PLD „Programmable Logic Device". FPGA nacházejí uplatnění jednak během procesu návrhu obvodů ASIC, kdy jsou využívány k testování navržených obvodových struktur, jednak obvody ASIC nahrazují tam, kde by jejich vývoj a výroba byly nerentabilní. Typickým příkladem jsou malosériové výroby specializovaných digitálních systémů. Obvody FPGA jsou také využívány tam, kde je třeba minimalizovat dobu vývoje a výroby digitálního systému (Time to Market), který může být klíčový pro komerční úspěch a získání pozice na trhu.

Nejčastěji jsou obvody FPGA vyráběny technologií SRAM (produkty firem Altera, Xilinx a Lattice Semiconductor). Objevují se však také obvody realizované technologií Anti-Fuse (například produkty firmy Actel).

1.2.3 Obvody ASIC

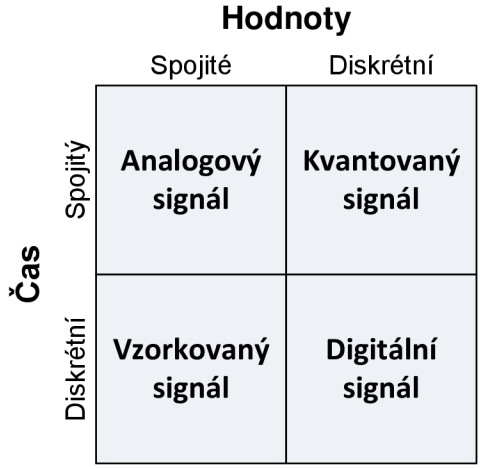
Výraz ASIC je akronymem anglických slov „Application Specific Integrated Circuit", do češtiny se nejčastěji překládá jako zákaznický obvod. Jedná se o označení pro integrované obvody navržené na míru konkrétnímu zákazníkovi. Typickým příkladem mohou být řídící jednotky, nebo systémy pro zpracování signálu.

Z implementačního hlediska dávají zákaznické obvody návrhářům značnou volnost při realizaci vytvářeného systému, zároveň však vyžadují značné investice jak do vývoje, tak do přípravy výroby (např. litografické masky). Z tohoto důvodu jsou plně zákaznické obvody využívány pouze pro realizaci velkých výrobních sérií nebo ve specifických případech (aplikace vyžadující vysokou spolehlivost, nízko příkonové aplikace, …).

2 Teorie převodu signálu

2.1 Signály analogové a digitální

Z hlediska této práce je nejdůležitějším hlediskem pro klasifikaci signálů jejich spojitost v čase a hodnotách. Rozlišovány jsou čtyři základní typy signálů: analogový, kvantovaný, vzorkovaný a digitální. Princip klasifikace velmi názorně ilustruje Obrázek 3. Analogový signál je spojitý v čase i hodnotách, naopak digitální signál je v diskrétní v čase a kvantovaný v hodnotách. Vznik digitálního signálu z analogového je realizován prostřednictvím dvou kroků, kterými jsou vzorkování a kvantování. Pořadí těchto kroků není pro převod důležité [4].



Obrázek 3 - klasifikace signálu na základě spojitosti v čase a hodnotách [4]

Vzorkování („Sampling“), je proces konverze v čase spojitého signálu $f(t)$ na posloupnost diskrétních vzorků $f_n = f(t_n)$ pro určité hodnoty t_n nezávislé proměnné času t. Tyto vzorky jsou obvykle ekvidistantní, a tedy $t_n = n \cdot T_S$, kde T_S je vzorkovací perioda. Vzorkování tedy lze vyjádřit vztahem [5]:

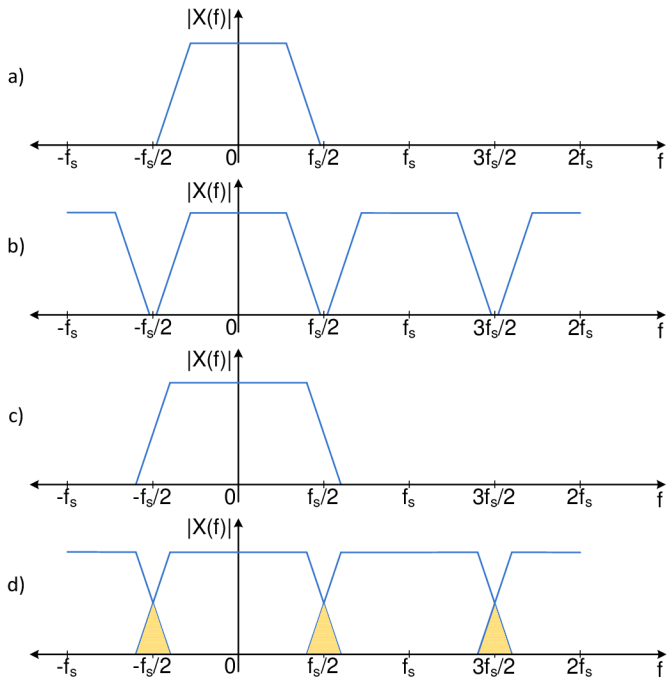
$$f_n = f(n \cdot T_S) \tag{1}$$

Podle [5] lze dále dokázat, že vzorkováním dochází ke změnám ve spektru signálu. Spektrum ideálně vzorkovaného signálu je totiž tvořeno součtem nekonečného počtu replik spektra původního analogového signálu, které jsou vzájemně posunuty o celé násobky vzorkovacího kmitočtu. Nemá-li dojít ke ztrátě přenášené informace, musí každá replika nést úplnou a nezměněnou informaci o původním signálu. Odtud také plyne potřeba zabránit překrytí spekter (aliasingu). K překrytí spekter nedojde při dodržení takzvaného vzorkovacího teorému (někdy také nazývaném podle autorů: Nyquistův, Kotělnikovův, Shannonův) [5]:

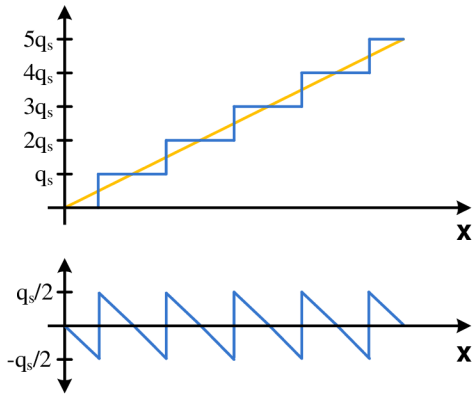
$$f_S \geq 2 \cdot f_B \tag{2}$$

kde f_S je vzorkovací kmitočet a f_B je horní mezní kmitočet pásma zpracovávaného signálu. Příklady spekter původního a navzorkovaného signálu jak při dodržení, tak při nedodržení

vzorkovacího teorému zachycuje Obrázek 4. Oranžovým šrafováním je zvýrazněna oblast nežádoucího překrytí spekter, jehož důsledkem je nevratná ztráta přenášených dat.



Obrázek 4 - příklad modulových frekvenčních spekter při dodržení vzorkovacího teorému: a) vstupní signál b) navzorkovaný signál; a při nedodržení: c) vstupní signál d) navzorkovaný signál [4]



Obrázek 5 - ideální a kvantovaný signál, kvantovací chyba

Kvantování („Amplitude Quantization“), je dle [4] proces diskretizace hodnot zpracovávaného signálu. Princip kvantování spočívá ve vyjádření vstupní spojité hodnoty pomocí nejbližšího celočíselného násobku kvantovacího kroku q_S. Ten je dán podílem velikosti vstupního rozsahu kvantovacího obvodu a počtu možných výstupních hodnot 2^N, kde N je bitová šířka výstupního digitálního slova. Proces kvantování vnáší do zpracovávaného signálu neodstranitelnou kvantovací chybu ε, pro jejíž velikost platí nerovnice (1), princip kvantování a vzniku kvantovací chyby ilustruje Obrázek 5.

$$-\frac{q_S}{2} < \epsilon \leq \frac{q_S}{2} \tag{3}$$

Nyní vyvstává otázka, jak kvantovací chyba ovlivňuje spektru kvantovaného signálu? Odpověď závisí na vzájemné korelovatelnosti kvantovací chyby a vzorkovacího signálu, viz [6]. To ilustrují Obrázek 6, Obrázek 7 a Obrázek 8, na každém z obrázků jsou vždy tři grafy:

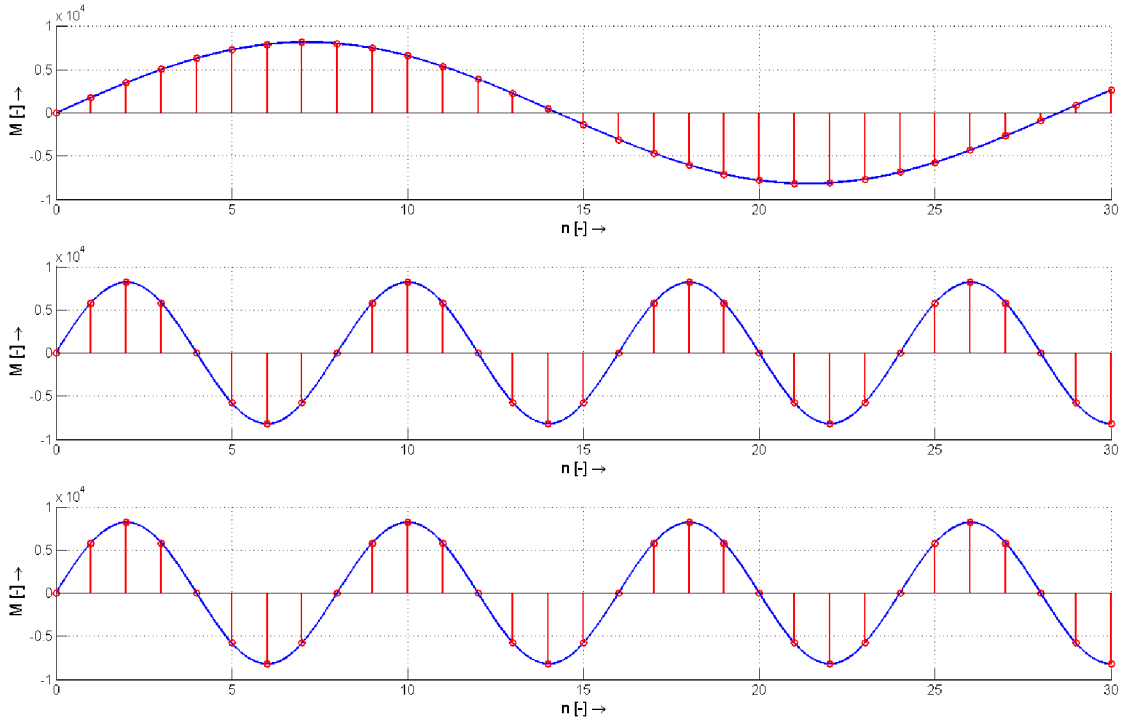
- první odpovídá sinusovému signálu s amplitudou FS/2 kvantovanému 14 bitovým kvantovacím obvodem a vzorkovanému frekvencí $f_s >> 2f$,
- druhý odpovídá sinusovému signálu s amplitudou FS/2 kvantovanému 14 bitovým kvantovacím obvodem a vzorkovanému frekvencí $f_s = 8f$,
- třetí odpovídá sinusovému signálu s amplitudou FS/2 s aditivním bílým šumem nahrazujícím kvantovací šum a vzorkovanému frekvencí $f_s = 8f$.

Obrázek 6 zachycuje časové průběhy vzorků, Obrázek 7 zachycuje průběhu velikosti kvantovacích chyb a Obrázek 8 zachycuje spektrální výkonové hustoty kvantovaných signálů.

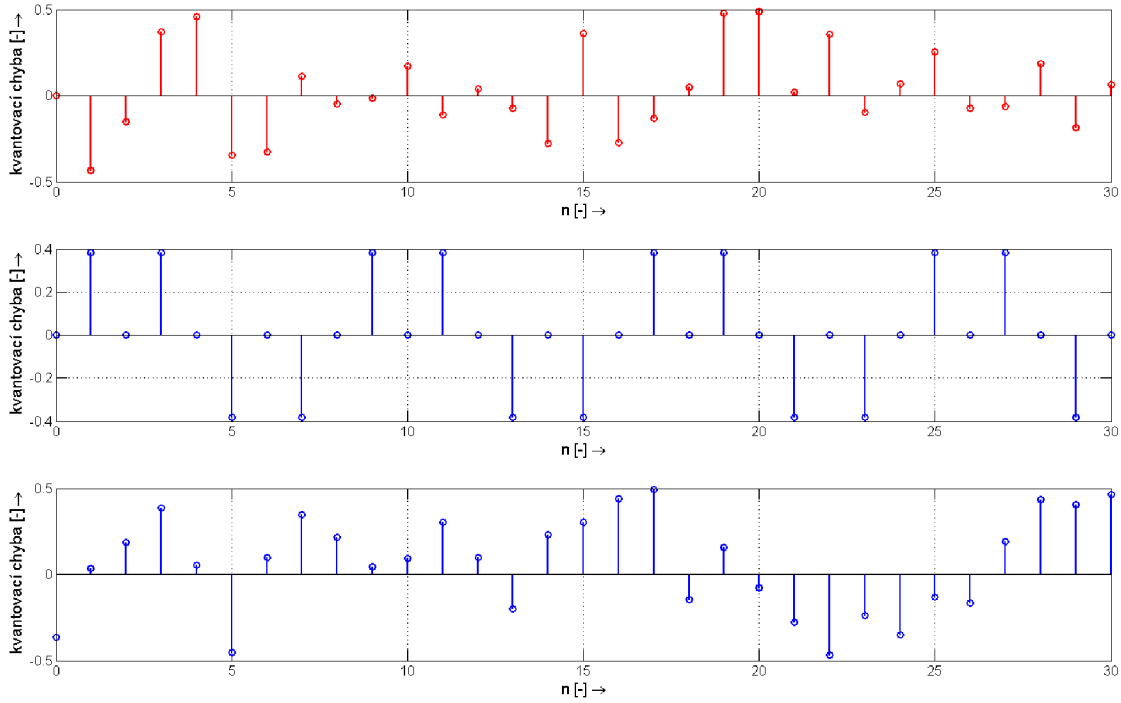
V prvním případě je průběh kvantovací chyby zcela náhodný, neexistuje tedy korelace mezi kvantovací chybou a vzorkovacím signálem. Vzhledem k tomu, že frekvenční spektrum kvantovací chyby není omezeno a nesplňuje tedy podmínku vzorkovacího teorému, dochází ve frekvenční oblasti k překrývání spekter původního signálu a jeho obrazů. Z kvantovací chyby se stává kvantovací šum.

Ve druhém případě je situace jiná. Průběh kvantovací chyby je viditelně periodický, existuje tedy korelace mezi kvantovací chybou a vzorkovacím signálem a vzorkovací chyba nepřechází do kvantovacího šumu. Projevuje jako úzký pík.

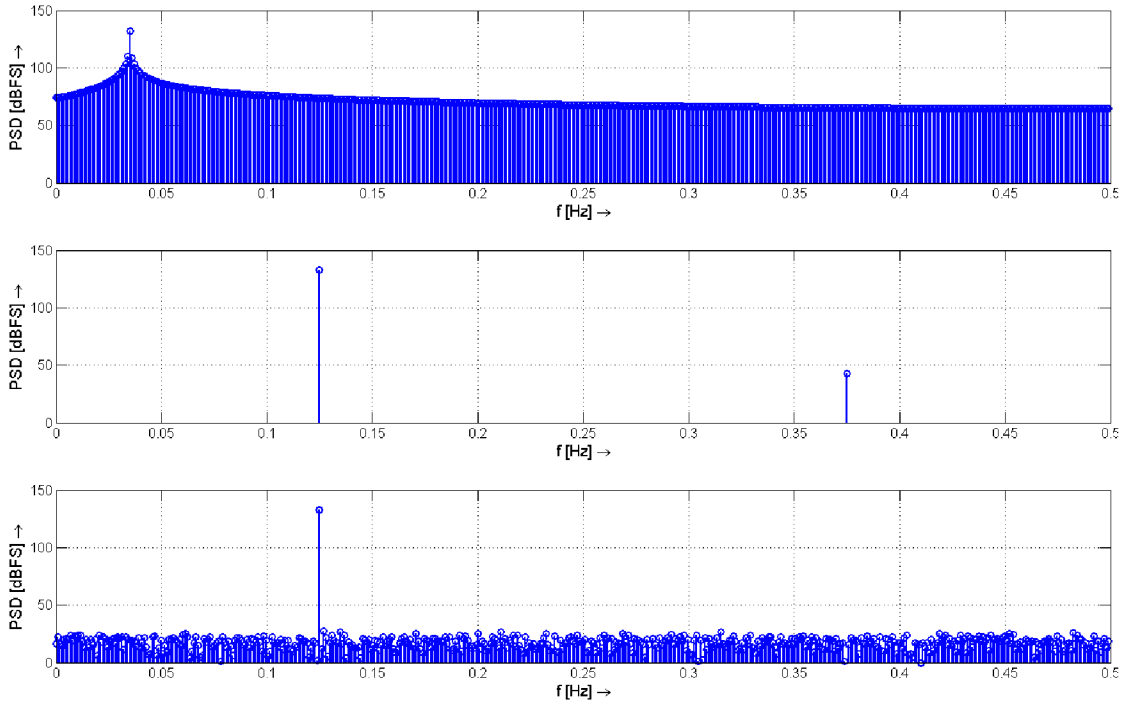
Ve třetím případě je průběh kvantovací chyby opět náhodný, tedy neexistuje korelace mezi kvantovací chybou a vzorkovacím signálem.



Obrázek 6 - časové průběhy zkoumaných signálů



Obrázek 7 - průběhy kvantovací chyby



Obrázek 8 - spektrální výkonová hustota kvantovaného signálu

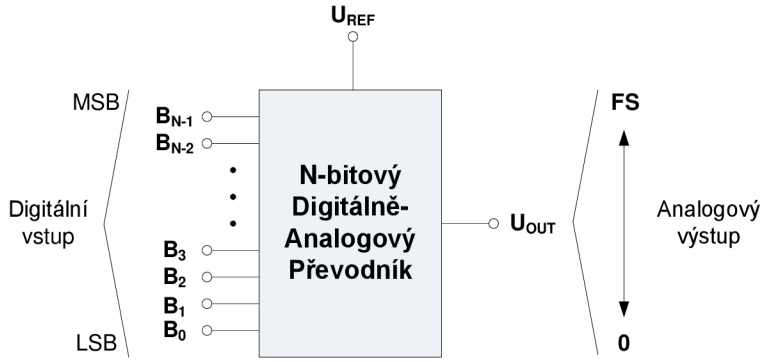
2.2 Teorie digitálně-analogového převodu

Jako digitálně-analogový, nebo zkráceně D/A převod je označován proces, při kterém je na základě vstupního digitálního signálu získáván výstupní analogový signál. Budeme-li uvažovat nejčastější formu D/A převodu, tedy převod binárního čísla na napětí, lze převod popsat vztahem:

$$U_{OUT} = \sum_{i=0}^{N-1} B_i \cdot 2^{-i} \cdot U_{REF} \tag{4}$$

kde U_{OUT} je hodnota výstupního napětí, N je rozlišení, respektive bitová šířka převodníku, B_i je hodnota i-tého bitu vstupního digitálního slova, člen 2^{-i} vyjadřuje váhu tohoto bitu a U_{REF} je hodnota referenčního napětí použitého pro převod [4].

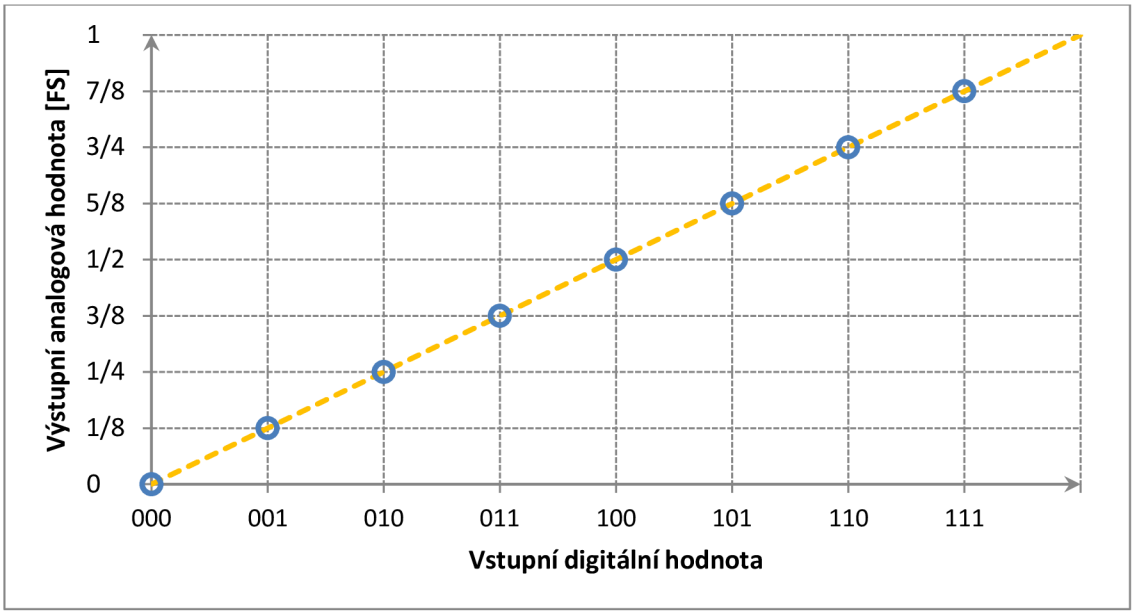
Elektronické obvody realizující digitálně-analogový převod jsou označovány jako digitálně-analogové převodníky, příklad principiálního schématu digitálně-analogového převodníku zachycuje Obrázek 9:



Obrázek 9 - schematická značka digitálně-analogového převodníku [7]

Funkce digitálně-analogového převodníku je vyjadřována pomocí převodní charakteristiky, což je grafické vyjádření velikosti výstupního analogového signálu v závislosti na vstupním digitálním signálu. Příklad ideální převodní charakteristiky pro tříbitový převodních zachycuje Obrázek 10. Převodní charakteristika digitálně-analogového převodníku je diskrétní, v tomto případě vyjádřena modrými kroužky, oranžová přerušovaná čára slouží pouze pro zdůraznění linearity. Za povšimnutí stojí fakt, že pro vstupní hodnotu odpovídající maximální hodnotě vstupního digitálního slova není výstup roven plnému rozsahu FS („Full Scale“), ale hodnotě FS – 1 LSB. To je dáno prostou skutečností, že 2^N intervalů je vymezeno 2^N+1 hodnotami. Využití rozsahu $0 \div 2^N$-1 vychází z konvence přijaté výrobci převodníků.

V současné době je pro výrobu digitálně-analogových převodníků využívána celá řada funkčních principů, obvodových řešení a cílových technologií. Každý z těchto prvků určitým způsobem ovlivňuje vlastnosti výsledného převodníku, při klasifikaci potom, spíše než o jednotlivých parametrech, uvažujeme celou množinu parametrů. Jejich popisu a vysvětlení jsou věnovány následující podkapitoly této práce, které vychází z pramenů [4], [7] a [8].



Obrázek 10 - ideální převodní charakteristika 3bitového D/A převodníku [7]

2.3 Vlastnosti D/A převodníků

2.3.1 Statické vlastnosti

Statické vlastnosti digitálně-analogového převodníku jsou plně vyjádřeny jeho převodní charakteristikou. Převodní charakteristika však nemusí být ve všech situacích ideálním způsobem pro jejich vyjádření. Například při vytváření katalogů je mnohem vhodnější použít statické parametry převodníku odvozené z jeho převodní charakteristiky. Nejčastěji jsou využívány tyto parametry:

- **Výstupní rozsah převodníku**

Je definován jako rozdíl maximální a minimální analogové výstupní hodnoty. Určuje tedy, v jakém napěťovém prostoru se budou pohybovat výstupní hodnoty digitálně-analogového převodníku.

- **Rozlišení převodníku**

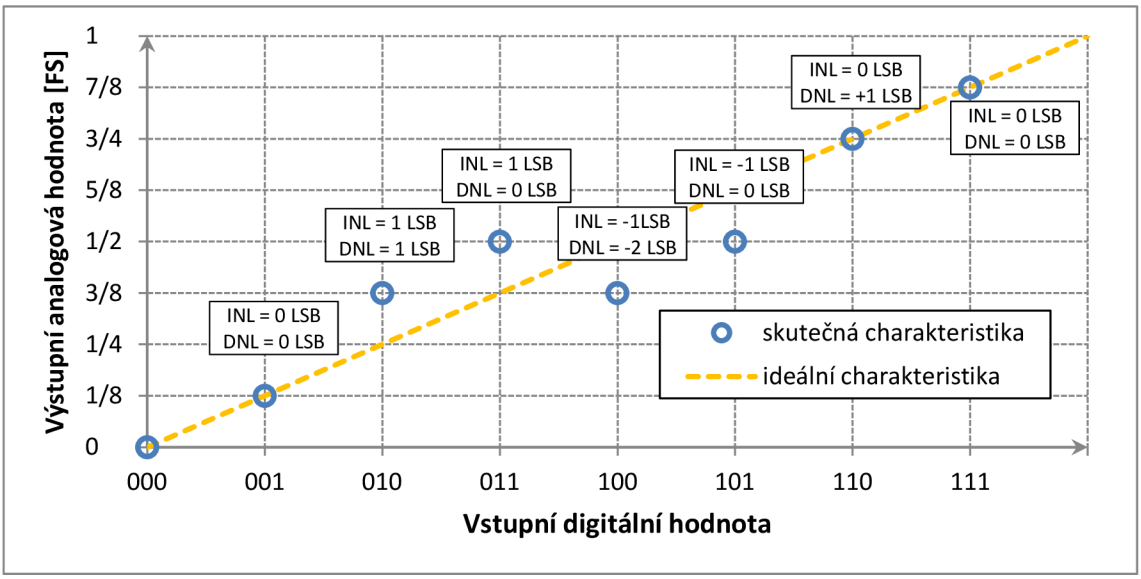
Bývá definováno jako hodnota odpovídající velikosti změny výstupního signálu na základě zvýšení hodnoty vstupního digitálního signálu o jedničku. Jiná forma vyjádření je prostřednictvím bitové šířky vstupního digitálního slova.

- **Integrální nelinearita (INL - Integral Non Linearity)**

Představuje velikost odchylky reálné převodní charakteristiky od ideální. Bývá specifikována jako číselný údaj vyjadřující její maximální hodnotu, nebo formou grafického vyjádření závislosti své velikosti na vstupním digitálním signálu. Vliv integrální nelinearity na převodní charakteristiku ilustruje Obrázek 11.

- **Diferenciální nelinearita (DNL - Differential Non Linearity)**

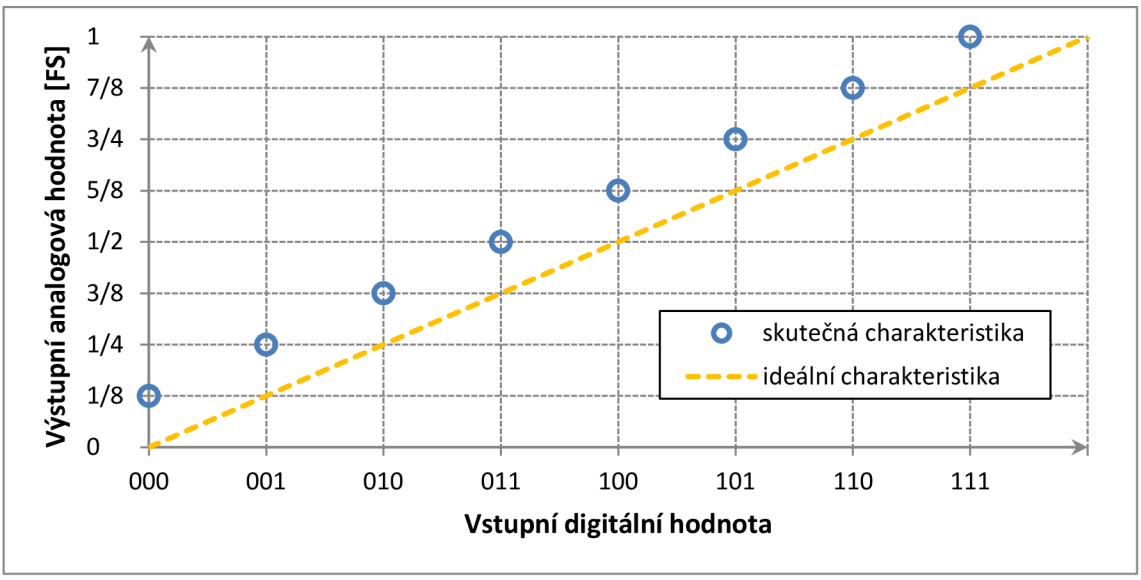
Vyjadřuje rozdíl mezi předpokládanou a reálnou změnou velikosti výstupního signálu, při změně vstupního digitálního signálu o jedničku. V ideálním případě by tato změna měla být rovna 1 LSB. Vliv diferenciální nelinearity na převodní charakteristiku ilustruje Obrázek 11.



Obrázek 11 - převodní charakteristika D/A převodníku - ilustrace integrální a diferenciální chyby

- **Chyba nastavení nuly (Offset error)**

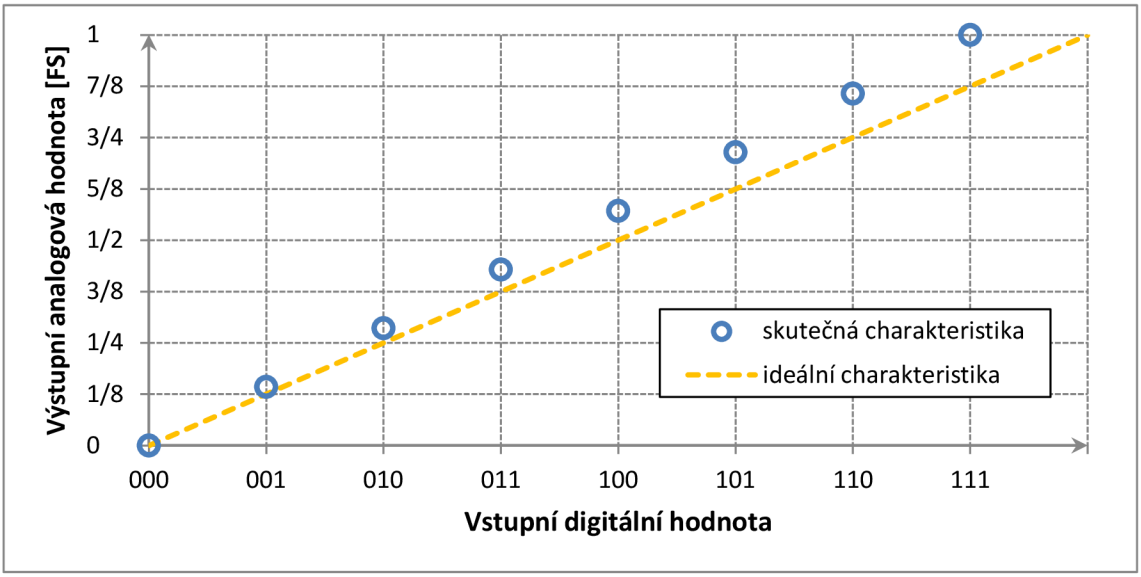
Projevuje se vertikálním posunutím převodní charakteristiky. Příklad převodní charakteristiky vykazující chybu nastavení nuly zachycuje Obrázek 12.



Obrázek 12 - převodní charakteristika D/A převodníku vykazující chybu nastavení nuly

- **Chyba zesílení (Gain error)**

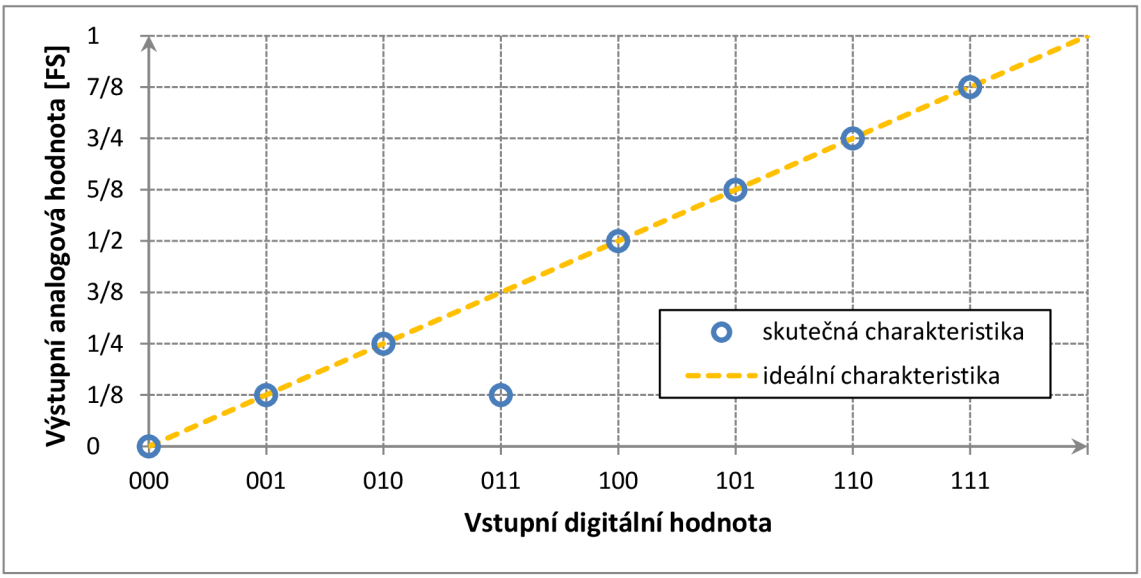
Projevuje se změnou strmosti převodní charakteristiky. Příklad převodní charakteristiky vykazující chybu zesílení zachycuje Obrázek 13.



Obrázek 13 - převodní charakteristika D/A převodníku vykazující chybu zesílení

- **Monotónnost (Monotonicity)**

Je vlastnost převodní charakteristiky, vypovídající o jejím průběhu. Pokud hodnota diferenciální nelinearity neklesne pod -1 LSB, lze charakteristiku označit za neklesající, a tedy monotónní. V opačném případě hovoříme o chybě monotónnosti. Příklad převodní charakteristiky vykazující chybu monotónnosti zachycuje Obrázek 14.



Obrázek 14 - převodní charakteristika D/A převodníku vykazující chybu monotónnosti

2.3.2 Dynamické vlastnosti

Dynamické vlastnosti popisují chování převodníků při zpracovávání střídavých signálů. Velmi často jsou vyjadřovány v kmitočtové oblasti. Při výpočtech i měření je uvažován harmonický signál, ačkoli v technické praxi jsou obvykle zpracovávány složitější signály.

- **Odstup signál šum (SNR - Signal to Noise Ratio)**

Je poměr středních hodnot výkonu odpovídajících FS hodnotě vstupního digitálního slova a šumu na výstupu převodníku. Někdy bývá také označován jako dynamický rozsah. Budeme-li uvažovat harmonický signál, lze pro výpočet SNR použít zjednodušený vztah: [4]:

$$SNR = 6{,}02 \cdot N + 1{,}76 \qquad [dB] \tag{5}$$

- **Odstup signál šum a zkreslení (SINAD / SNDR – Signal to Noise and Distortion Ratio)**

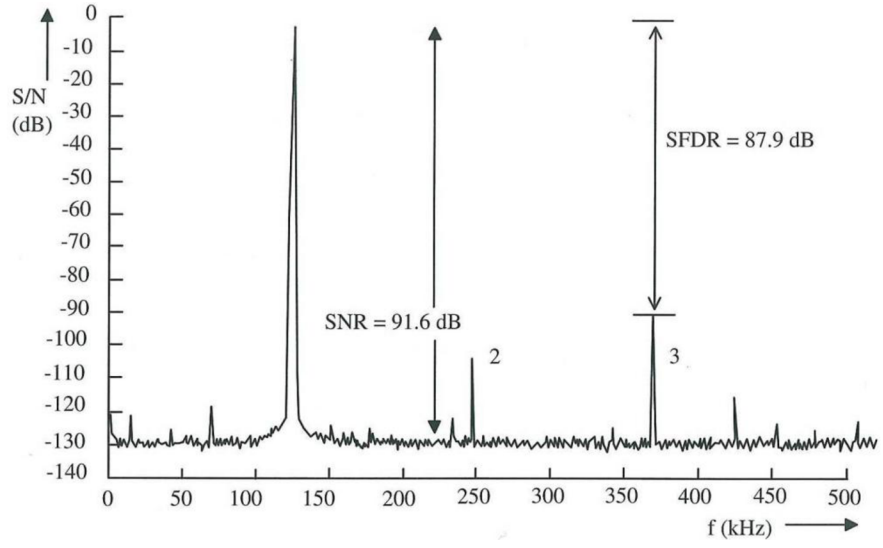
Představuje rozšíření parametru SNR, neboť zahrnuje také nelineární zkreslení. Lze jej vypočítat jako poměr efektivní hodnoty signálu a geometrického součtu vyšších harmonických složek signálu doplněného o šum. Tento parametr je závislý jak na frekvenci, tak na amplitudě zpracovávaného signálu.

- **Dynamický rozsah bez parazitních složek (SFDR - Spurious Free Dynamic Range)**

Je poměr maximálních hodnot užitečného a parazitního signálu. Význam je podobný jako v případě parametru SNR, tento parametr je však přísnější, jak ilustruje Obrázek 15. SFDR bývá využíván především u převodníků s vysokým převzorkováním, nebo u převodníků s požadavkem na vysokou čistotu spektra. Hodnotu SFDR lze vypočítat podle vztahu [4]:

$$SFDR = \frac{1}{|INL| \cdot 2^{-N} + 2^{-1{,}5N}} \qquad [-] \tag{6}$$

kde INL vyjadřuje integrální nelinearitu a N rozlišení převodníku.



Obrázek 15 - ilustrace dynamického rozsahu bez parazitních složek [4]

- **Efektivní počet bitů (ENOB - Effective Number Of Bits)**

Je alternativním způsobem pro vyjádření rozlišení převodníku. Tento parametr má dobrou vypovídací hodnotu, neboť v sobě zahrnuje působení většiny negativních jevů vznikajících při převodu. Pro výpočet lze použít vztah [8]:

$$ENOB = \frac{SINAD_{DB} - 1{,}76}{6{,}02} \qquad [-] \tag{7}$$

- **Celkové harmonické zkreslení (THD - Total Harmonic Distortion)**

Je vyjádřeno jako podíl geometrického součtu vyšších harmonických složek signálu a efektivní hodnoty první harmonické složky. Pro měření je obvykle využíván

$$THD = \frac{\sqrt{U_2^2 + U_3^2 + + \cdots + U_\infty^2}}{U_1} \qquad [-] \tag{8}$$

- **Doba ustálení výstupu (Settling Time)**

Je definována jako nejdelší časový interval, který, při konstantní hodnotě vstupního digitálního signálu, uplyne od zahájení převodu do ustálení výstupní hodnoty s chybou převodu nepřesahující specifikaci převodníku.

2.3.3 Další vlastnosti

Do skupiny další vlastnosti byly zařazeny ty, které nemají přímou souvislost ani s jednou z předchozích kategorií, nicméně mají vliv při posuzování signálových převodníků

- **Princip převodu**

Je to způsob, kterým je převod realizován. Pro ilustraci lze zmínit převodníky s váhovými rezistory, převodníky se sítí R-2R, převodníky založené na pulzně-šířkové modulaci, nebo převodníky založené na modulaci sigma-delta.

- **Poměr šířky pásma přenášeného signálu k šířce pásma převodníku**

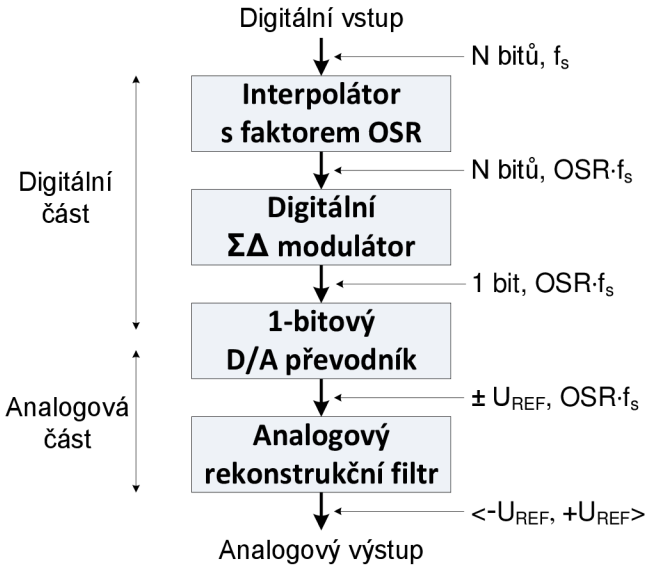
Existují dvě základní třídy. První třídou jsou převodníky pracující v Nyquistově oblasti („Nyquist range Converters“), u kterých je maximální frekvence přenášeného signálu blízká polovině vzorkovací frekvence. Druhou třídou jsou převodníky s převzorkováním („Oversampling Converters“), jejichž vzorkovací frekvence je výrazně vyšší, než je maximální frekvence přenášeného signálu.

- **Typ výstupu převodníku**

Prvním z aspektů této vlastnosti je výstupní veličina, nejčastěji to může být napětí, nebo proud. Dalším z aspektů je symetrie, nebo nesymetrie výstupu.

2.4 D/A převodníky typu sigma-delta

Digitálně-analogové převodníky typu sigma delta patří do skupiny převodníků s převzorkováním. Charakteristické jsou dobrou linearitou převodní charakteristiky a vysokým dosažitelným rozlišením. Velice významným aspektem je dále možnost digitální realizace hlavních částí převodníku, jedinou nutně analogovou součást tvoří výstupní rekonstrukční filtr. Blokové schéma digitálně-analogového převodníku typu sigma delta zachycuje Obrázek 16.



Obrázek 16 - blokové schéma digitálně-analogového převodníku typu sigma-delta [9]

2.4.1 Interpolátor

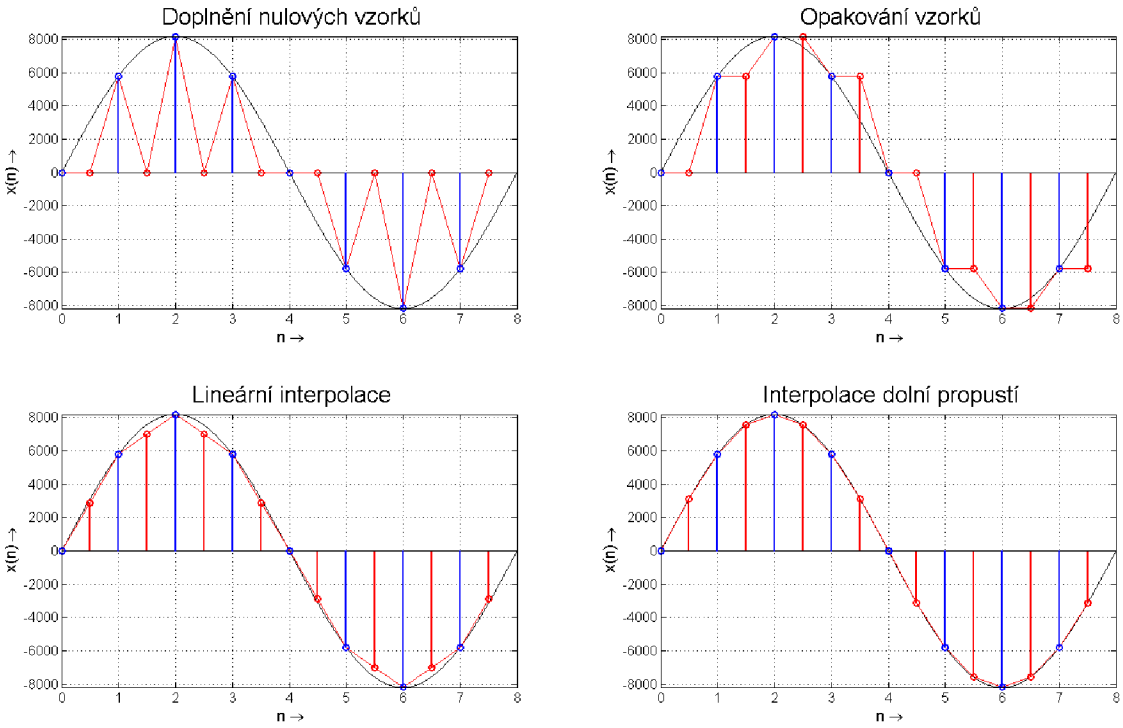
Jak již bylo zmíněno v úvodu, digitálně-analogové převodníky typu sigma-delta pracují na výrazně vyšších frekvencích, než je vzorkovací frekvence převáděného signálu. Před vstupem signálu do digitálního sigma-delta modulátoru je tedy nutné provést zvýšení jeho vzorkovací frekvence. K tomuto účelu slouží interpolátor. Jeho funkce spočívá ve vkládání nových vzorků mezi vzorky vstupního signálu tak, aby bylo dosaženo požadovaného zvýšení vzorkovací frekvence, při minimálním dopadu na spektrum zpracovávaného signálu. Počet vzorků, které musí být přidány, závisí na poměru převzorkování a lze jej stanovit:

$$L = (OSR - 1) = \frac{f_{S2}}{f_{S1}} - 1 \tag{9}$$

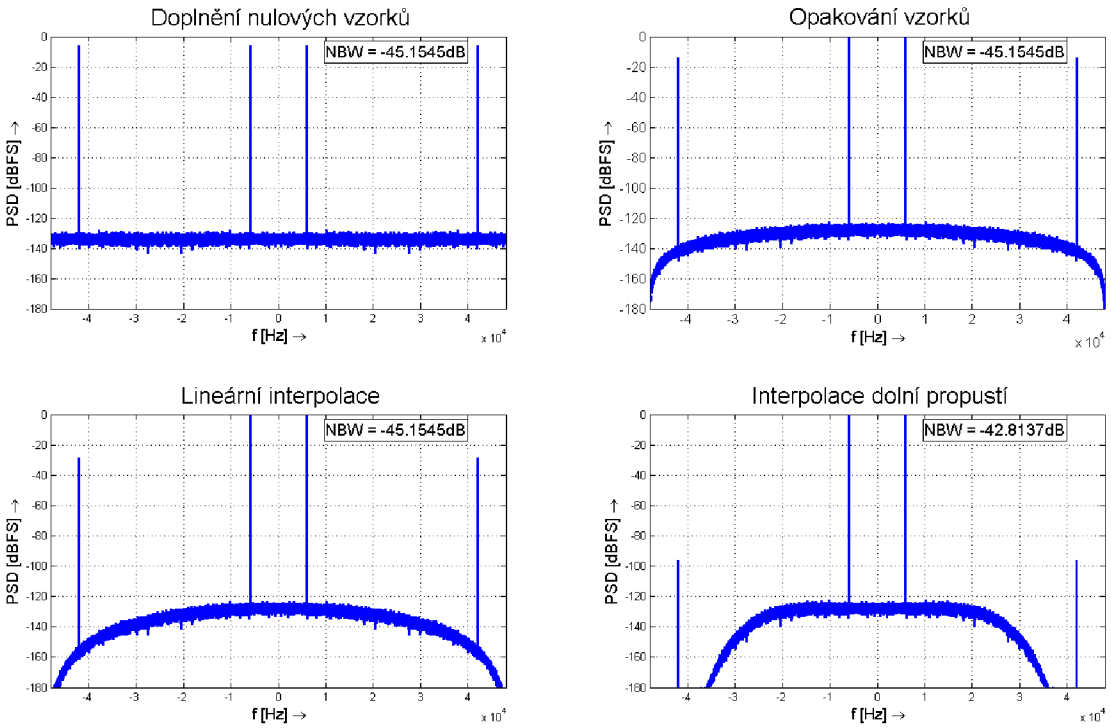
kde L je počet vzorků vložených mezi dva vzorky původního signálu, OSR je poměr převzorkování, f_{S1} je původní vzorkovací frekvence a f_{S2} je nová vzorkovací frekvence.

Interpolátor lze realizovat různými způsoby, od vkládání nulových vzorků, či opakování vzorků, přes lineární interpolaci až ke vkládání nulových vzorků s následnou filtrací dolní propustí. Výhodou prvních dvou postupů je jednoduchost, druhé dva postupy mají menší dopad na spektrum interpolovaného signálu. Princip jednotlivých interpolačních metod je dobře patrný

z časových průběhů, které zachycuje Obrázek 17. Pro vytvoření představy o vlivu na spektrum dobře poslouží PSD signálu po interpolaci, které zachycuje Obrázek 18



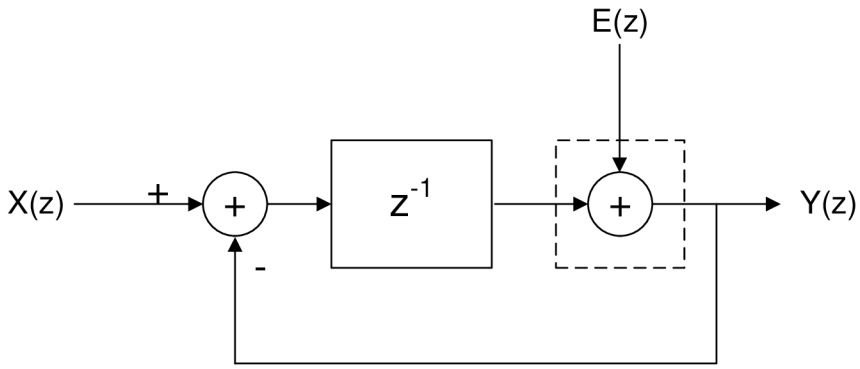
Obrázek 17 - časový průběh signálu před a po interpolaci: černá křivka - nekvantovaný signál, modré píky - vstupní signálu, červené píky - výstupního signálu



Obrázek 18 - spektrální výkonová hustota signálu po interpolaci

2.4.2 Digitální modulátor sigma-delta

Modulátor sigma-delta převádí vstupní digitální hodnotu prostřednictvím pulzně-hustotní modulace na posloupnost impulzů - „bitstream“. Okrajové podmínky představuje stav, kdy je vstupní hodnota rovna minimu nebo maximu vstupního rozsahu, v tom případě je výstup trvale roven úrovní “L“ nebo “H“. Hlavním přínosem modulátoru sigma-delta je však jedna z jeho vlastností, která se nazývá tvarování šumu („noiseshaping“). Modulátor v podstatě mění rozložení energie šumu ve spektru zpracovávaného signálu a to takovým způsobem, že šum z nízkých frekvencí je vytlačován směrem k vyšším frekvencím.



Obrázek 19 - blokové schéma digitálního modulátoru sigma-delta[10]

Pro vysvětlení mechanizmu tvarování šumu poslouží blokové schéma digitálního modulátoru sigma-delta prvního řádu, které zachycuje Obrázek 19. Výstupní signál modulátoru Y(z) je získáván ze vstupního signálu X(z) a chybového signálu (bílého šumu) E(z) podle vztahu [10]:

$$Y(z) = X(x) \cdot z^{-1} + E(z) \cdot (1 - z^{-1}) \tag{10}$$

Z této rovnice ovšem vyplývá jedna velmi zajímavá skutečnost. Přenosová funkce STF pro signál („signal transfer function“) je jiná než přenosová funkce NTF pro šum („noise transfer function“), jak dokumentují vztahy (11) a (12). Pro upřesnění výraz z^{-1} představuje zpoždění o jeden vzorek, zatímco výraz $(1\text{-}z^{-1})$ představuje filtraci horní propustí prvního řádu.

$$STF(z) = z^{-1} \tag{11}$$

$$NTF(z) = (1 - z^{-1}) \tag{12}$$

Tento princip lze samozřejmě využít také pro vytvoření modulátorů vyšších řádů, které umožní odstranění většího množství šumu z pracovního pásma. S nárůstem řádu modulátoru však roste riziko vzniku nestability systému, která je pochopitelně nežádoucí. Modulátory prvního řádu jsou však stabilní vždy.

2.4.3 Analogový rekonstrukční filtr

Poslední částí převodníku je rekonstrukční filtr. Jeho realizace není možná prostřednictvím digitálních filtrů, neboť jejich výstupem je opět digitální signál. Vzhledem k požadavkům na co nejnižší grupové zpoždění, které je závislé na linearitě fázové charakteristiky filtru, bývá obvykle využita besselova, nebo butterworthova aproximace. Velmi příjemnou konsekvencí vysokého pracovního kmitočtu modulátoru je uvolnění nároků na strmost rekonstrukčního filtru. Proto ani řád filtru, který využívá besselovu aproximaci obvykle nepřekročí 10.

3 Návrh a simulace modelu převodníku

Proces návrhu digitálně-analogového převodníku je rozdělen do čtyř dílčích kroků, kterými jsou analýza zadání, návrh interpolačního filtru, návrh digitálního modulátoru sigma-delta a návrh analogového rekonstrukčního filtru. Stejné dělení bylo použito také pro tuto kapitolu. Pro účely návrhu, modelování a simulace v krocích dva až čtyři je využito v prostředí MATLAB R2012a (verze 7.14.0.739 win64), jeho nástavba SIMULINK (verze 7.9 2012a), DSP System Toolbox (verze 8.2), SD Toolbox (verze 2.0) a Delta-Sigma Toolbox (verze 7.3).

3.1 Analýza zadání

Prvním krokem každého návrhového procesu je analýza zadání. V jejím rámci dochází nejprve k vyjádření požadavků zadání prostřednictvím dobře kvantifikovatelných parametrů a jejich zhodnocení z hlediska opodstatněnosti, dosažitelnosti a ceny. Následuje hledání metod, postupů, obvodových principů a koncepcí, které umožní realizaci obvodové struktury s požadovanými parametry.

Cílem této práce je navrhnout digitálně analogový převodník typu sigma-delta s efektivním rozlišením 14 bitů, určený ke zpracování signálů ve frekvenčním pásmu 0 ÷ 20 kHz. Tento převodník je určen pro implementaci do digitálních obvodů ASIC, bude tedy realizován jako digitální obvodová struktura. Výjimku tvoří pouze rekonstrukční filtr, který musí být z principiálních důvodů realizován analogovou formou.

Ze znalosti šířky pásma zpracovávaného signálu je možné stanovit vzorkovací kmitočet vstupního signálu. Pro zadaný rozsah, který odpovídá audiopásmu, je obvyklé využít vzorkovací frekvenci 44,1 kHz nebo 48 kHz. Vzhledem k tomu, že při neměnné šířce pásma se zvýšení vzorkovací frekvence projeví snížením nároků na strmost filtrů interpolátoru, byla zvolena frekvence $f_s = 48$ kHz.

Na základě zadaného efektivního rozlišení převodníku (efektivního počtu bitů ENOB) jsou vlastně determinovány požadavky jeho na šumové vlastnosti. Parametr ENOB je definován prostřednictvím parametru SINAD, který kombinuje parametry SNR a THD. Pro účely návrhu je možné vliv parametru THD zanedbat a potom platí:

$$ENOB = \frac{SINAD_{DB} - 1{,}76}{6{,}02} \approx \frac{\mathrm{SNR} - 1{,}76}{6{,}02} \qquad [-] \tag{13}$$

$$SNR = 6{,}02 \cdot N + 1{,}76 = 6{,}02 \cdot 14 + 1{,}76 = \underline{\underline{86{,}04\ dB}} \tag{14}$$

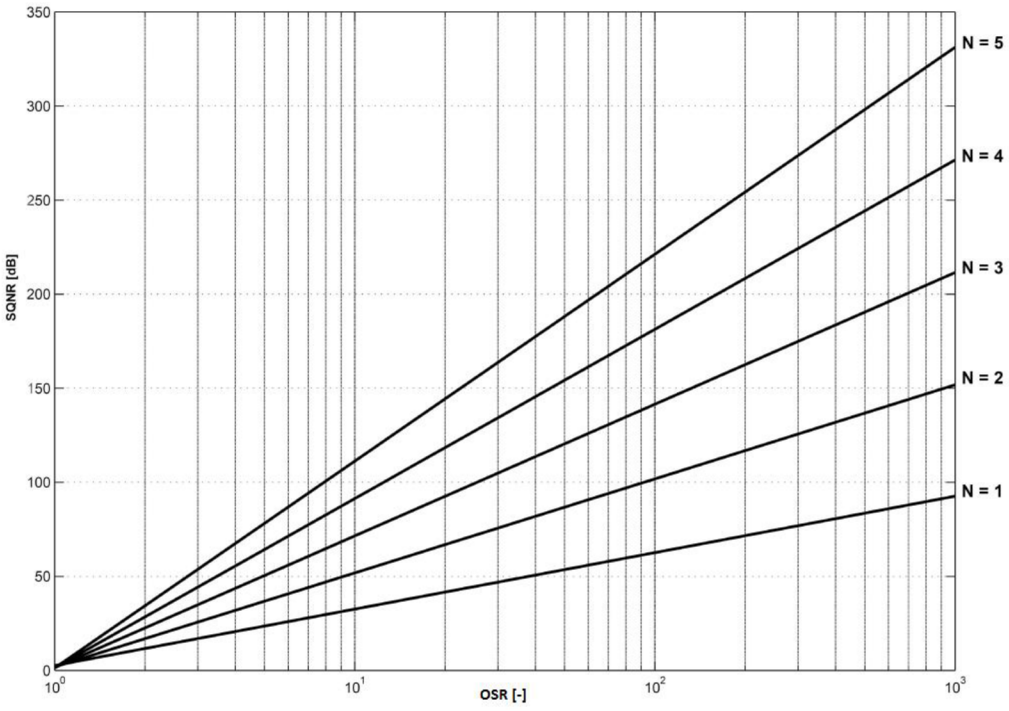
Jak ilustruje Obrázek 16, struktura digitálně-analogových převodníků typu sigma-delta má sériový charakter a zpracování signálu tedy probíhá postupně. Při sériovém zpracování platí, že výsledné vlastnosti systému jsou dány vlastnostmi nejméně výkonného subsystému. Aby bylo možno dosáhnout zadaných šumových vlastnosti celého převodníku, musí být šumové vlastnosti každého ze subsystémů lepší, nebo alespoň rovny požadavkům na celý systém.

Prvním subsystémem, do kterého signál vstupuje, je interpolátor. V interpolátoru dochází ke vkládání nulových vzorků a následnému odstranění nežádoucích replik základního spektra. Toto odstranění probíhá formou filtrace, požadované SNR se tedy odráží v požadavcích na minimálním potlačení filtrů v zádržném pásmu. Z hlediska správné funkce je však také důležité, aby zvlnění modulové přenosové charakteristiky, v propustném pásmu, bylo co nejmenší. V praxi jsou obvykle vyžadovány hodnoty v intervalu 0.1 ÷ 0.001 dB. V rámci tohoto návrhu bude vyžadována hodnota cca 0.05dB.

Druhým subsystémem, který zpracovává signál je digitální modulátor sigma-delta. Modulátor sám svojí funkcí vnáší určité množství šumu do zpracovávaného signálu, je tedy nutno najít vhodnou kombinaci řádu modulátoru a koeficientu převzorkování, tak aby tento šum negativně neovlivnil vlastnosti celého systému. Vhodným vodítkem pro volbu může být graf, který zachycuje Obrázek 20. Tento graf vyjadřuje relativní velikost šumu vzniklého uvnitř modulátoru v závislosti na koeficientu převzorkování a řádu modulátoru využívajícího funkci $(1-z^{-1})^N$. Parametr SNQR je ekvivalentem SNR, jedná se o odstup signál-kvantovací šum. Z grafu je patrné, že nejvhodnější kombinací je 2. řád filtru při OSR = 128. Dalším z parametrů modulátoru je počet kvantovacích úrovní na výstupu modulátoru. Po konzultaci s vedoucím práce byla vzhledem k předpokládaným implementačním účelům zvolena hodnota M = 2, výstupem tedy bude digitální signál.

Třetím, a posledním, subsystémem je rekonstrukční analogový filtr. Pro jeho realizaci byla zvolena butterworthova aproximace, která představuje kompromis mezi chebyshevovou aproximací s velkou strmostí v přechodovém pásmu a besselovou aproximací s lineární fázovou charakteristikou. Mezní kmitočet filtru vychází z šířky pracovního pásma.

Závěry analýzy zadání shrnuje Tabulka 1.



Obrázek 20 - velikost SNQR v závislosti na OSR a řádu sigma-delta modulátoru realizující funkci $(1-z^{-1})^N$ [10]

Tabulka 1 - požadované parametry navrhovaného převodníku

Šířka frekvenčního pásma	f_{bw}	20 kHz
Vzorkovací frekvence	f_s	48 kHz
Efektivní rozlišení	ENOB	> 14 bitů
Šířka vstupního slova	N	14
Odstup signál šum	SNR	> 86 dB
Koeficient převzorkování	OSR	128
Řád modulátoru	-	2
Počet úrovní výstupního kvantizátoru	M	2

Během návrhového procesu bylo zjištěno, že dosažení takových šumových poměrů uvnitř navrhovaných subsystémů, aby zůstalo zachováno SNR vstupního digitálního signálu, pravděpodobně nebude možné. Snížení SNR zpracovávaného signálu by mělo za následek snížení efektivního rozlišení převodníku. Bylo proto zvoleno řešení běžné u komerčně vyráběných převodníků. Došlo ke zvětšení šířky vstupního digitálního slova, což dále ovlivnilo některé dílčí parametry. Tabulka 2 zachycuje korigované požadavky na parametry navrhovaného převodníku.

Tabulka 2 - korigované požadavky na parametry navrhovaného převodníku

Šířka frekvenčního pásma	f_{bw}	20 kHz
Vzorkovací frekvence	f_s	48 kHz
Efektivní rozlišení	ENOB	> 14 bitů
Šířka vstupního slova	N	15
Odstup signál šum	SNR	> 92,06 dB
Koeficient převzorkování	OSR	128
Řád modulátoru	-	2
Počet úrovní výstupního kvantizátoru	M	2

3.2 Interpolátor

Obvodová funkce interpolátoru, nároky na něj kladené a možné principy jeho realizace byly rozebrány v předchozí kapitole. V technické praxi je nejčastěji využíván interpolační princip, vycházející z doplnění nulových vzorků a následného vyfiltrování signálu FIR filtrem typu dolní propust. V rámci této práce bude tento princip využit také.

Ačkoli z hlediska hardwarových nároků jsou FIR filtry výrazně náročnější než filtry typu IIR, umožňují vytvoření filtračních struktur s lineární fází, což determinuje konstantního grupového zpoždění u takovéhoto filtru. Dalším pozitivem je stabilita, neboť FIR filtry jsou vždy stabilní.

Ze závěrů analýzy signálu lze odvodit požadavky na parametry interpolátoru, respektive jeho filtru typu dolní propust. Tyto parametry shrnuje Tabulka 3. Pracovní frekvence představuje OSR-násobek vzorkovací frekvence vstupního signálu, propustné pásmo odpovídá

pracovnímu pásmu převodníku a velikost přechodového pásma je stanovena tak, aby byly potlačeny veškeré nežádoucí obrazy základního spektra signálu. Maximální velikost zvlnění v propustném pásmu a minimální potlačení v zádržném pásmu byly stanoveny již ve fázi analýzy zadání.

Tabulka 3 - požadované vlastnosti filtru interpolátoru

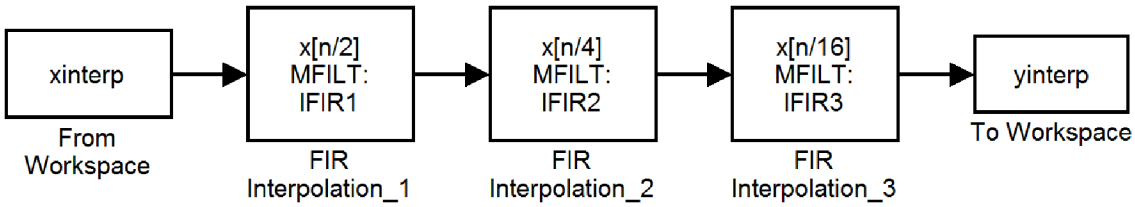
Pracovní frekvence	6144 kHz
Propustné pásmo	0 ÷ 20 kHz
Přechodové pásmo	20 ÷ 28 kHz
Zádržné pásmo	28 ÷ 3072 kHz
Zvlnění v propustném pásmu	< 0,05 dB
Potlačení v zádržném pásmu	> 92,06 dB

Ačkoli to nemusí být na první pohled zcela zřejmé, takovýto filtr není možné efektivně realizovat. Požadavek na relativně velmi úzké propustné i přechodové pásmo, společně s požadavkem na malé zvlnění v propustném pásmu a vysoké potlačení v zádržném pásmu totiž povedou k velmi vysokému řádu filtru, který v takovémto případě může být i několik tisíc [11].

Mnohem efektivnějším způsobem jak dosáhnout požadovaných vlastností je vytvoření kaskády filtrů, viz [12]. Tento postup byl také zvolen v rámci této práce. Po důkladném zvážení jsem zvolil kaskádu tvořenou třemi filtry, jak ilustruje Obrázek 21. Tabulka 4 zachycuje požadované parametry jednotlivých filtrů kaskády. Je třeba si uvědomit, že výsledná přenosová charakteristika bude součtem dílčích přenosových charakteristik, a proto byly nároky na maximální zvlnění v propustném a potlačení v zádržném pásmu mírně zvýšeny. Do nutnosti vytvořit tuto rezervu se promítá také fakt, že výsledná realizace těchto filtrů bude pracovat s fixed point aritmetikou, což bude vyžadovat nakvantování koeficientů. Nakvantování koeficientů se s velkou pravděpodobností projeví deformací přenosové charakteristiky.

Zdrojový kód 1 je výpisem skriptu sloužícího k návrhu filtrů IFIR 1, IFIR2 a IFIR3. Tabulka 5 zaznamenává parametry takto navržených filtrů. Obrázek 22 dokumentuje modulovou převodní charakteristiku jak jednotlivých filtrů, tak celé kaskády. Je z něj dále patrné, že navržená kaskáda vyhovuje požadavkům na interpolátor.

Posledním krokem návrhu interpolátoru je převod kaskády filtrů z aritmetiky s plovoucí řádovou čárkou na aritmetiku s pevnou řádovou čárkou, která umožní efektivnější implementaci. To je provedeno doplněním návrhového skriptu parametry popisující vnitřní struktury aritmetických bloků, které mají být využity. Zdrojový kód 2 zachycuje výpis takto modifikovaného skriptu. Obrázek 23 zachycuje modulové přenosové charakteristiky filtrů po této modifikaci. Jak je z těchto charakteristik patrné, interpolátor vyhovuje.



Obrázek 21 - model interpolátoru na blokové úrovni (Simulink)

Tabulka 4 - požadované parametry jednotlivých filtrů kaskády

	IFIR1	IFIR2	IFIR3
Koeficient převzorkování	2	4	8
Pracovní frekvence	96 kHz	384 kHz	6144 kHz
Propustné pásmo	0 ÷ 20 kHz	0 ÷ 20 kHz	0 ÷ 20 kHz
Přechodové pásmo	20 ÷ 28 kHz	20 ÷ 68 kHz	20 ÷ 320 kHz
Zádržné pásmo	28 ÷ 48 kHz	68 ÷ 192 kHz	320 ÷ 372 kHz
Zvlnění v propustném pásmu	0,01 dB	0,01 dB	0,01 dB
Potlačení v zádržném pásmu	100 dB	100 dB	100 dB

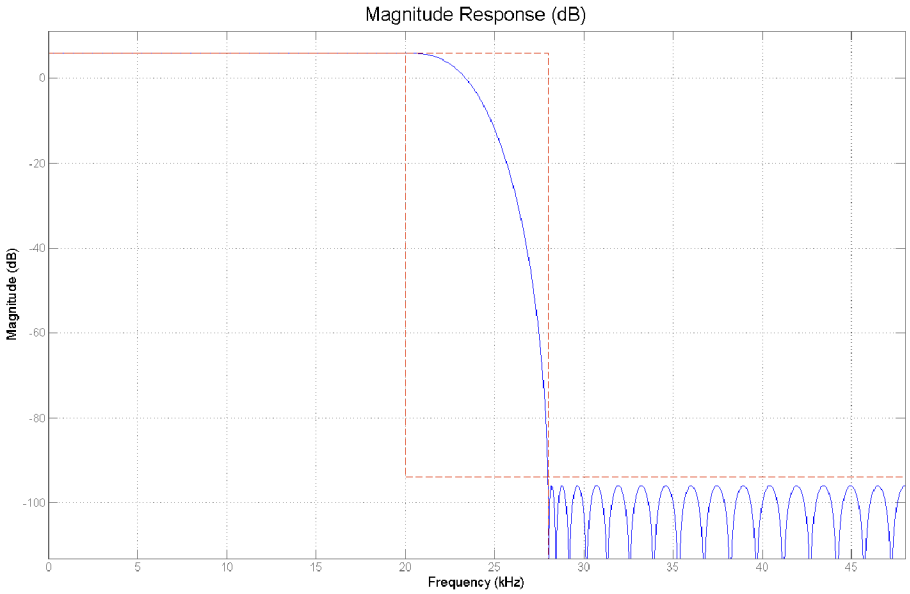
```
%%  Parameters of System

    fbw     = 20000;                            % Band width [Hz]
    fs      = 48000;                            % Sampling Frequency [Hz]
    osr     = [2,4,16];                         % OverSampling Ratio [-]
    fss     = [fs*prod(osr(1:1)) ...            % System Sampling Frequency
               fs*prod(osr(1:2)) ...
               fs*prod(osr(1:3))];

%%  Design of Interpolation FIR Filter 1
    Fpass = fbw;                                % Passband Frequency
    Fstop = fs-fbw;                             % Stopband Frequency
    Apass = 0.01;                               % Passband Ripple (dB)
    Astop = 100;                                % Stopband Attenuation (dB)
    Fs    = fss(1);                             % Sampling Frequency
    h = fdesign.interpolator(osr(1),'Lowpass','fp,fst,ap,ast', ...
        Fpass,Fstop,Apass,Astop,Fs);
    IFIR1 = design(h, 'ifir');

%%  Design of Interpolation FIR Filter 2
    Fpass = fbw;                                % Passband Frequency
    Fstop = fss(1)-(fs-fbw);                    % Stopband Frequency
    Apass = 0.01;                               % Passband Ripple (dB)
    Astop = 100;                                % Stopband Attenuation (dB)
    Fs    = fss(2);                             % Sampling Frequency
    h = fdesign.interpolator(osr(2),'Lowpass','fp,fst,ap,ast', ...
        Fpass,Fstop,Apass,Astop,Fs);
    IFIR2 = design(h, 'ifir');

%%  Design of Interpolation FIR Filter 3
    Fpass = fbw;                                % Passband Frequency
    Fstop = fss(2)-(fss(1)-fs+fbw);             % Stopband Frequency
    Apass = 0.01;                               % Passband Ripple (dB)
    Astop = 100;                                % Stopband Attenuation (dB)
    Fs    = fss(3);                             % Sampling Frequency
    h = fdesign.interpolator(osr(3),'Lowpass','fp,fst,ap,ast', ...
        Fpass,Fstop,Apass,Astop,Fs);
    IFIR3 = design(h, 'ifir');
```

Zdrojový kód 1 - výpis skriptu provádějícího návrh digitálních filtrů tvořících interpolátor (Matlab)

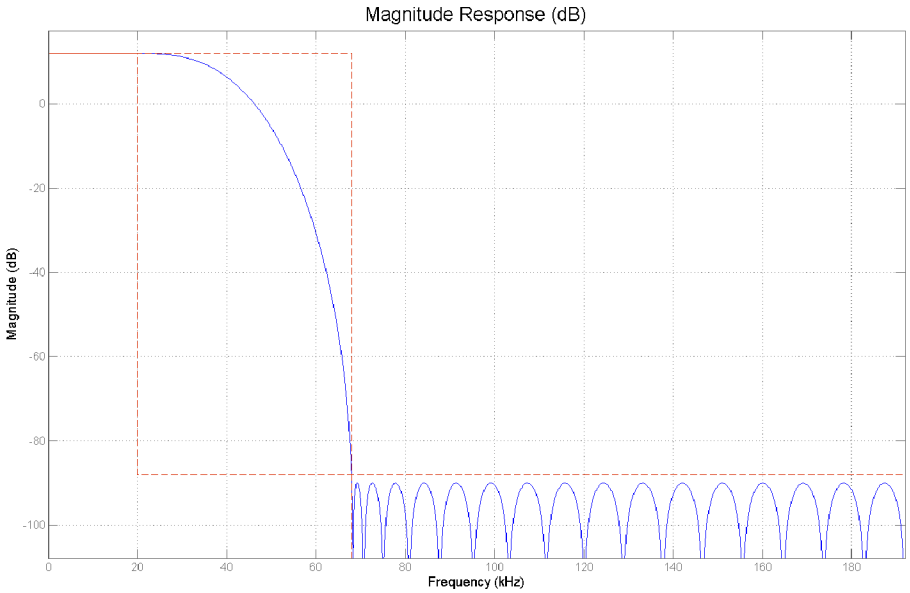
Tabulka 5 - parametry navržených filtrů

Parametr	IFIR1	IFIR2	IFIR3
Typ filtru	Direct-Form FIR Polyphase Interpolator	Direct-Form FIR Polyphase Interpolator	Direct-Form FIR Polyphase Interpolator
Řád	58	39	97
OSR	2	4	16
Stabilní	Ano	Ano	Ano

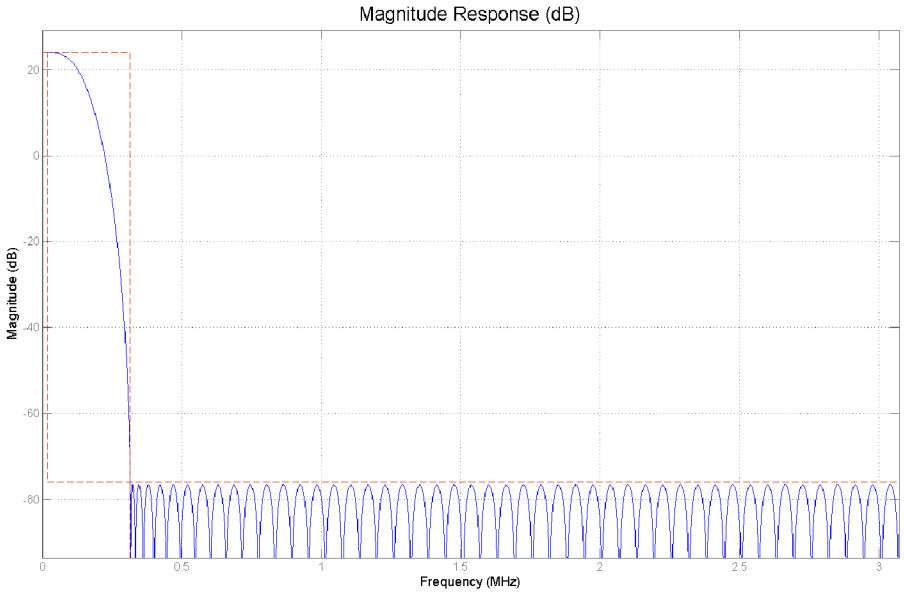
a)


Magnitude Response (dB)
Magnitude (dB)
Frequency (kHz)
0
-20
-40
-60
-80
-100
0
5
10
15
20
25
30
35
40
45

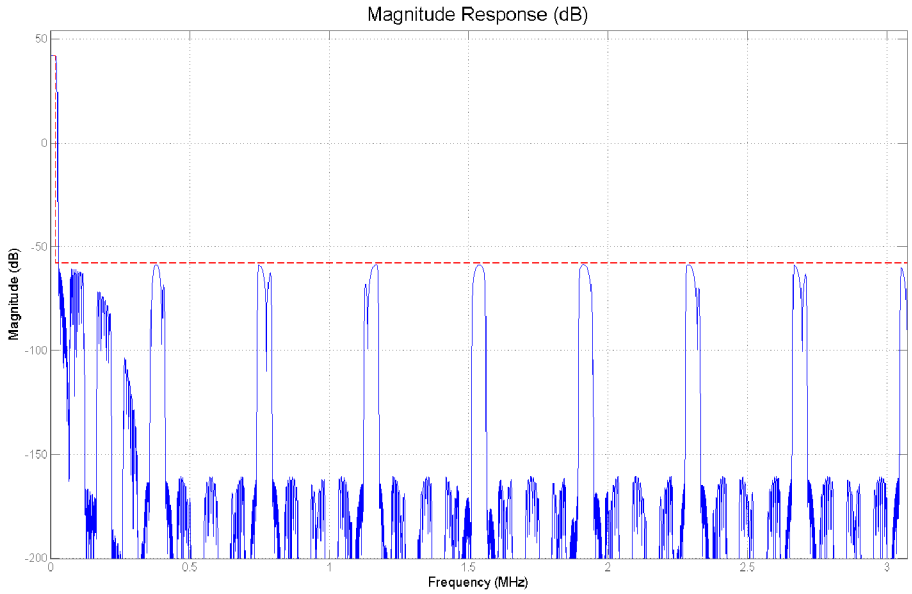
b)


Magnitude Response (dB)
Magnitude (dB)
Frequency (kHz)
0
-20
-40
-60
-80
-100
0
20
40
60
80
100
120
140
160
180

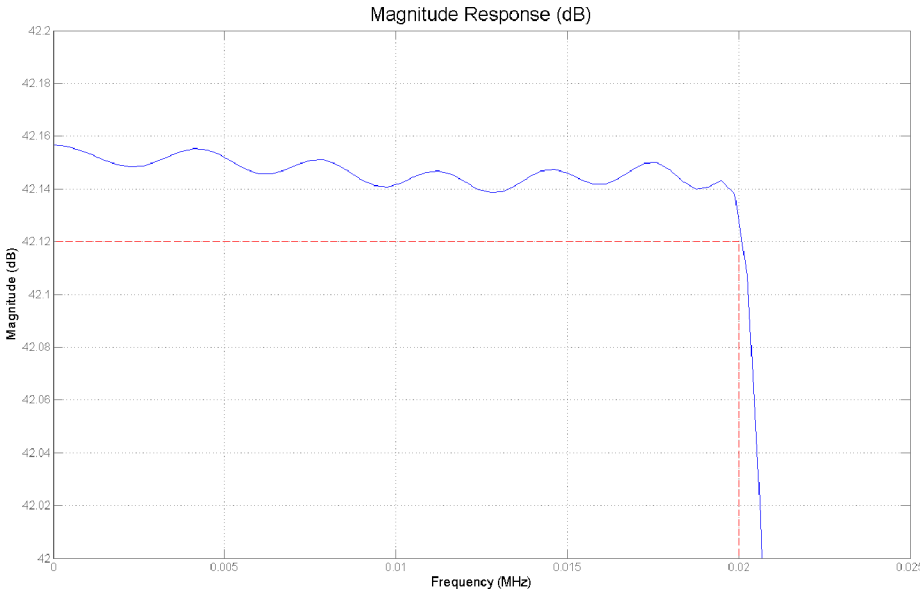
c)


Magnitude Response (dB)
Magnitude (dB)
Frequency (MHz)
20
0
-20
-40
-60
-80
0
0.5
1
1.5
2
2.5
3

d)



e)



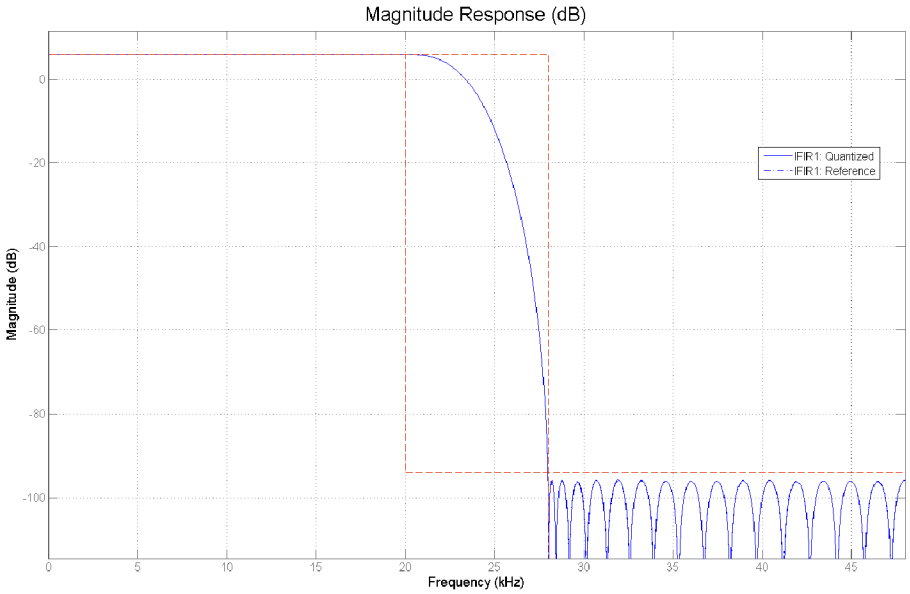
Obrázek 22 - modulové přenosové charakteristiky: a) první stupeň interpolátoru (IFIR1), b) druhý stupeň interpolátoru (IFIR2), c) třetí stupeň interpolátoru (IFIR3), d) celková přenosová charakteristika interpolátoru (kaskádní spojení IFIR1, IFIR2 a IFIR3), e) detail celkové přenosové charakteristiky interpolátoru – zvlnění v propustném pásmu

```
%%  Parameters of System
    fbw     = 20000;                          % Band width [Hz]
    fs      = 48000;                          % Sampling Frequency [Hz]
    osr     = [2,4,16];                       % OverSampling Ratio [-]
    fss     = [fs*prod(osr(1:1)) ...          % System Sampling Frequency
               fs*prod(osr(1:2)) ...
               fs*prod(osr(1:3))];

%%  Design of Interpolation FIR Filter 1
    Fpass = fbw;                              % Passband Frequency
    Fstop = fs-fbw;                           % Stopband Frequency
    Apass = 0.01;                             % Passband Ripple (dB)
    Astop = 100;                              % Stopband Attenuation (dB)
    Fs    = fss(1);                           % Sampling Frequency
```

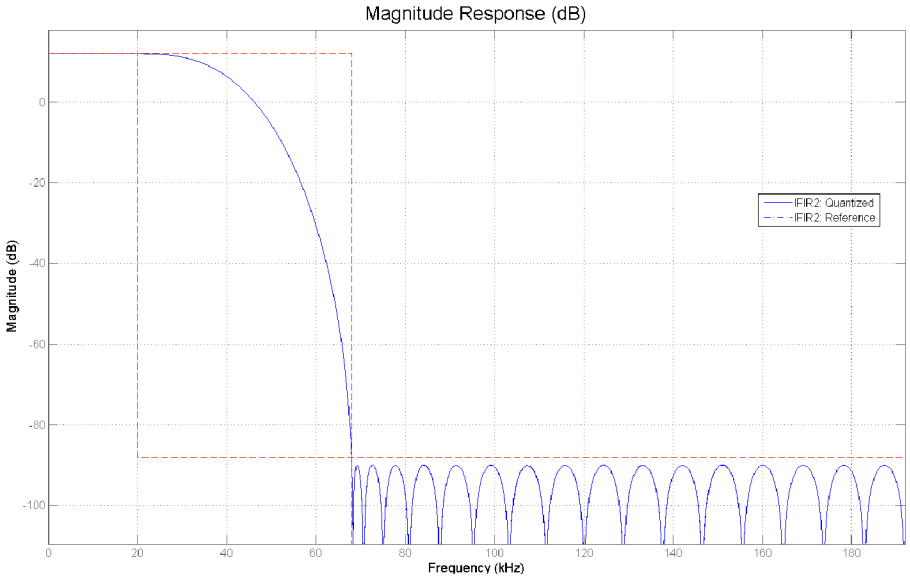
```
    h = fdesign.interpolator(osr(1),'Lowpass','fp,fst,ap,ast', ...
        Fpass,Fstop,Apass,Astop,Fs);
    IFIR1 = design(h, 'ifir');
    set(IFIR1, 'Arithmetic', 'fixed', ...
        'InputWordLength', res, ...
        'InputFracLength', 0, ...
        'CoeffWordLength', 24, ...
        'CoeffAutoScale', true, ...
        'FilterInternals', 'Specifyprecision', ...
        'ProductWordLength', 32, ...
        'ProductFracLength', 14, ...
        'AccumWordLength', 32, ...
        'AccumFracLength', 14, ...
        'OutputWordLength', res, ...
        'OutputFracLength', 0, ...
        'RoundMode', 'convergent', ...
        'OverflowMode', 'saturate');

%%  Design of Interpolation FIR Filter 2
    Fpass = fbw;                                % Passband Frequency
    Fstop = fss(1)-(fs-fbw);                    % Stopband Frequency
    Apass = 0.01;                               % Passband Ripple (dB)
    Astop = 100;                                % Stopband Attenuation (dB)
    Fs    = fss(2);                             % Sampling Frequency
    h = fdesign.interpolator(osr(2),'Lowpass','fp,fst,ap,ast', ...
        Fpass,Fstop,Apass,Astop,Fs);
    IFIR2 = design(h, 'ifir');
    set(IFIR2, 'Arithmetic', 'fixed', ...
        'InputWordLength', res, ...
        'InputFracLength', 0, ...
        'CoeffWordLength', 24, ...
        'CoeffAutoScale', true, ...
        'FilterInternals', 'Specifyprecision', ...
        'ProductWordLength', 32, ...
        'ProductFracLength', 14, ...
        'AccumWordLength', 32, ...
        'AccumFracLength', 14, ...
        'OutputWordLength', res, ...
        'OutputFracLength', 0, ...
        'RoundMode', 'convergent', ...
        'OverflowMode', 'saturate');

%%  Design of Interpolation FIR Filter 3
    Fpass = fbw;                                % Passband Frequency
    Fstop = fss(2)-(fss(1)-fs+fbw);             % Stopband Frequency
    Apass = 0.01;                               % Passband Ripple (dB)
    Astop = 100;                                % Stopband Attenuation (dB)
    Fs    = fss(3);                             % Sampling Frequency
    h = fdesign.interpolator(osr(3),'Lowpass','fp,fst,ap,ast', ...
        Fpass,Fstop,Apass,Astop,Fs);
    IFIR3 = design(h, 'ifir');
    set(IFIR3, 'Arithmetic', 'fixed', ...
        'InputWordLength', res, ...
        'InputFracLength', 0, ...
        'CoeffWordLength', 24, ...
        'CoeffAutoScale', true, ...
        'FilterInternals', 'Specifyprecision', ...
        'ProductWordLength', 32, ...
        'ProductFracLength', 14, ...
        'AccumWordLength', 32, ...
        'AccumFracLength', 14, ...
        'OutputWordLength', res, ...
        'OutputFracLength', 0, ...
        'RoundMode', 'convergent', ...
        'OverflowMode', 'saturate');
```

Zdrojový kód 2 - výpis skriptu provádějícího návrh digitálních filtrů tvořících interpolátor - optimalizace pro použití aritmetiky s pevnou řádovou čárkou (Matlab)

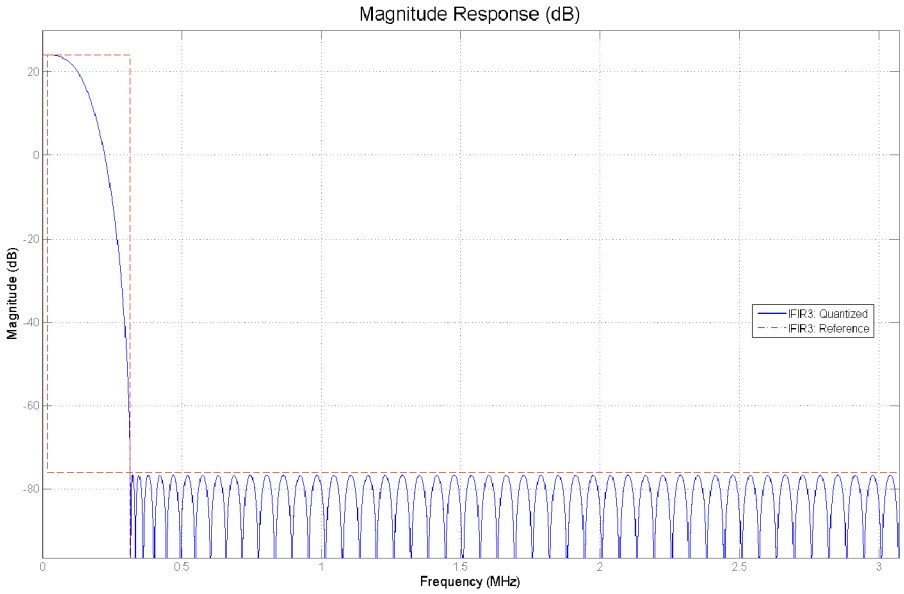
a)


Magnitude Response (dB)
Magnitude (dB)
Frequency (kHz)
IFIR1: Quantized
IFIR1: Reference

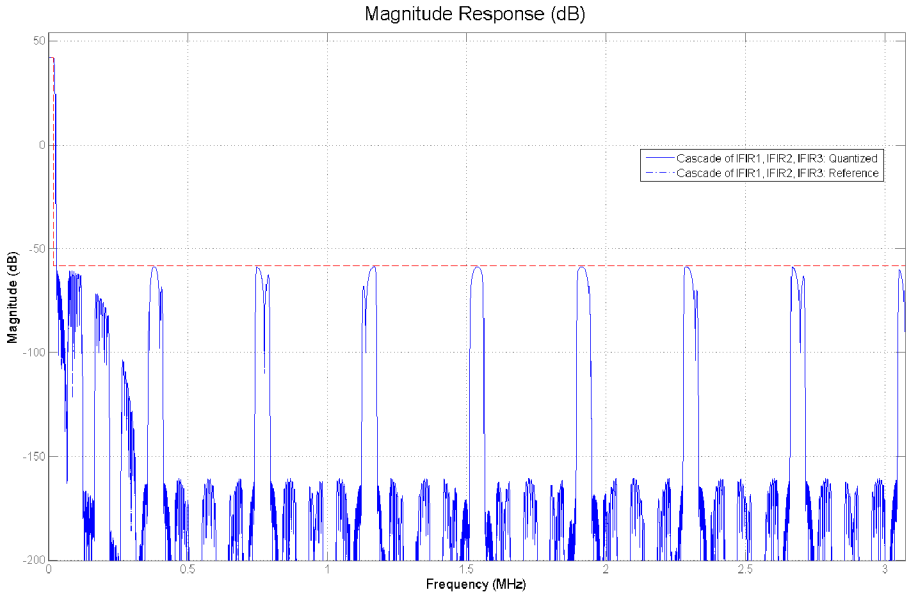
b)


Magnitude Response (dB)
Magnitude (dB)
Frequency (kHz)
IFIR2: Quantized
IFIR2: Reference

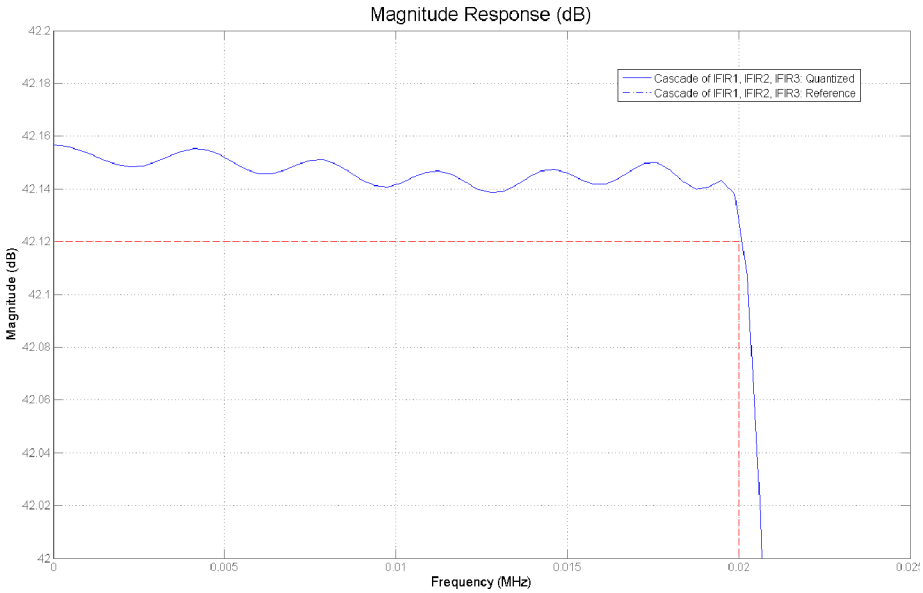
c)


Magnitude Response (dB)
Magnitude (dB)
Frequency (MHz)
IFIR3: Quantized
IFIR3: Reference

d)



e)

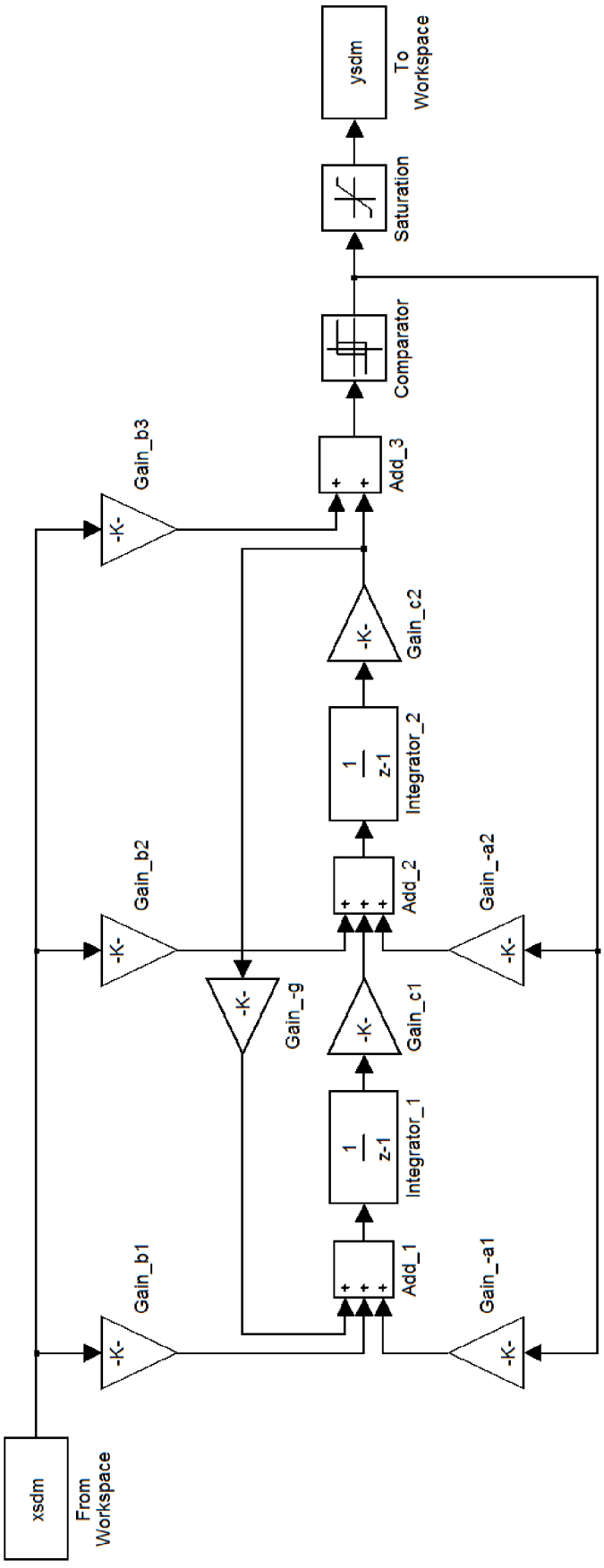


Obrázek 23 - modulové přenosové charakteristiky po převodu koeficientů do formátu pevné řádové čárky: a) první stupeň (IFIR1), b) druhý stupeň (IFIR2), c) třetí stupeň (IFIR3), d) celková přenosová charakteristika (kaskádní spojení IFIR1, IFIR2 a IFIR3), e) detail celkové přenosové charakteristiky

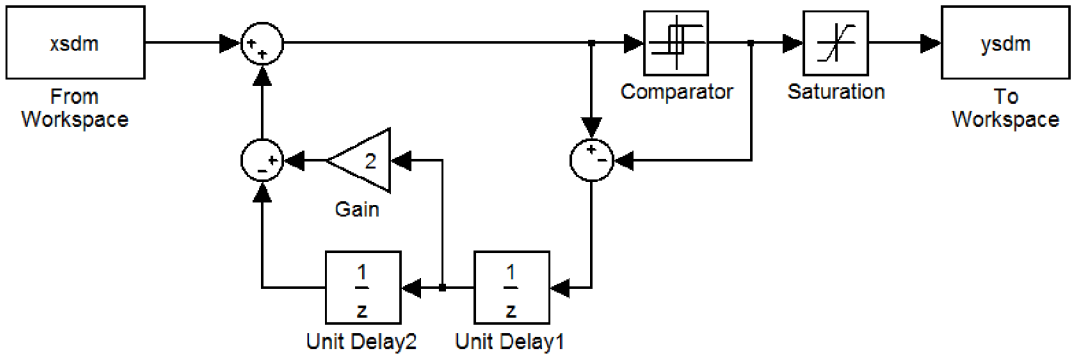
3.3 Modulátor sigma-delta

Základní parametry modulátoru sigma-delta, tedy řád a koeficient převzorkování, byly stanoveny v rámci analýzy zadání. Cílem třetího návrhového kroku je volba vhodné systémové struktury a výpočet koeficientů modulátoru sigma-delta. Následovat bude vytvoření idealizovaného modelu, ověření, vytvoření reálného modelu a finální ověření.

Jako vhodné varianty byly, s využitím pramene [6], vybrány struktury CIFB a ERRB, jejichž bloková schémata zachycují Obrázek 24 a Obrázek 25. Rozdílnost těchto struktur je patrná na první pohled. Zatímco struktura CIFB bývá využívána jak k realizaci digitálních, tak analogových modulátorů sigma-delta, struktura ERFB se v technické praxi využívá pouze v digitální formě, viz [6].



Obrázek 24 - model digitálního modulátoru sigma delta, struktura CIFB 2. řád (Simulink)



Obrázek 25 - model digitálního modulátoru sigma delta, struktura ERFB 2. řád (Simulink)

Pro strukturu ERFB není třeba provádět návrh koeficientů, pro strukturu CIFB však ano. K tomuto účelu byl využit Delta-Sigma Toolbox. Nejlepších vlastností bylo dosaženo s hodnotami koeficientů, které zachycuje Zdrojový kód 3:

```
a     = [0.2487  0.2060     ];
b     = [0.2487  0.0     0.0];
c     = [0.2836  5.0594     ];
g     = 7.080320797367080e-04
```

Zdrojový kód 3 - hodnoty koeficientů pro strukturu CRFB 2. Řád

Dalším krokem bylo ověření vlastností obou struktur. K tomuto účelu byly vytvořeny skripty „Script_11_System_Simulation“ a „Script_12_SNR_Screening“. První ze skriptů slouží k získání informací o spektru výstupního signálu, pro jednu konkrétní velikost frekvence a amplitudy vstupního signálu, což dokumentuje Obrázek 26 a Obrázek 27. Druhý skript slouží k získání informací o velikosti SNR v závislosti na změně frekvence a amplitudy vstupního signálu. Koeficient k, přitom vyjadřuje poměr mezi maximální a použitou amplitudou vstupního signálu. Grafický výstup druhého skriptu zachycuje Obrázek 28 a Obrázek 29.

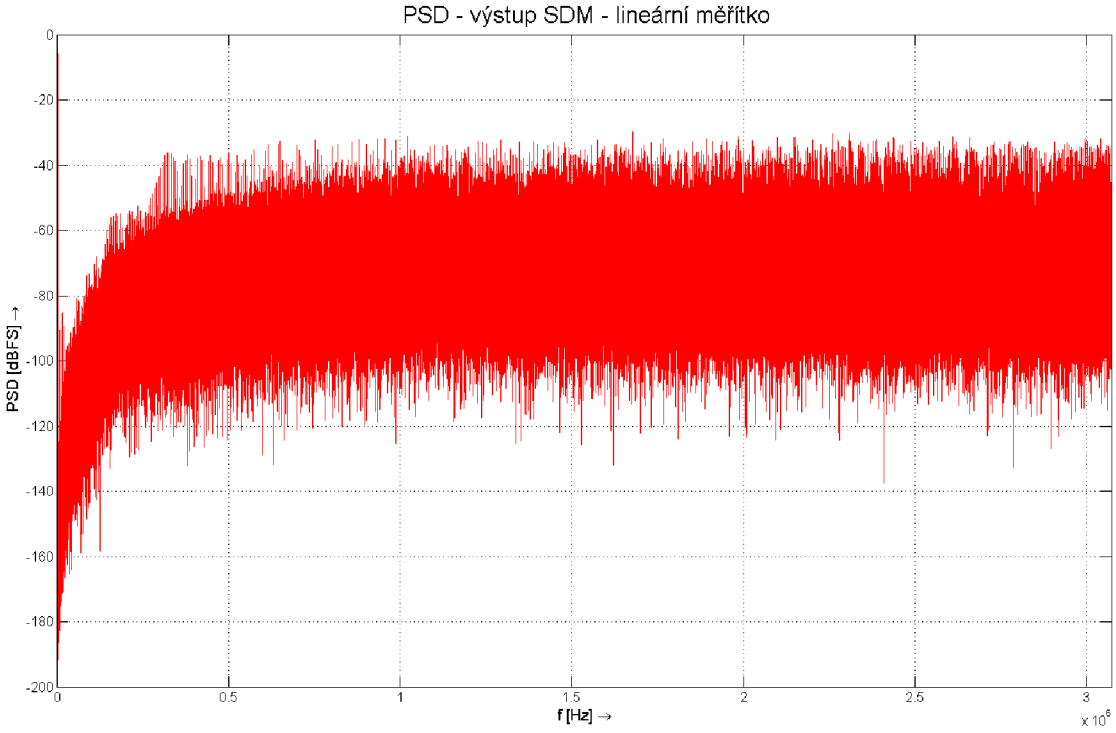
Obdobně jako u interpolačního filtru je třeba provést optimalizaci navržených struktur pro použití aritmetiky s pevnou řádovou čárkou. To v případě modulátoru ERFB znamená stanovit bitovou šířku vnitřních signálů. Pro šířku vstupních slov N = 15 bitů je nutná bitová šířka ostatních signálů uvnitř modulátoru 40 bitů.

V případě modulátoru CIFB je třeba mimo stanovení bitové šířky vnitřních signálů také provést kvantování koeficientů. Hodnoty kvantovaných koeficientů modulátoru CIFB dokumentuje Zdrojový kód 4, nutná bitová šířka vnitřních signálů byla stanovena na 64 bitů. Pro účely simulace byl vytvořen nový model, který dokumentuje Obrázek 30.

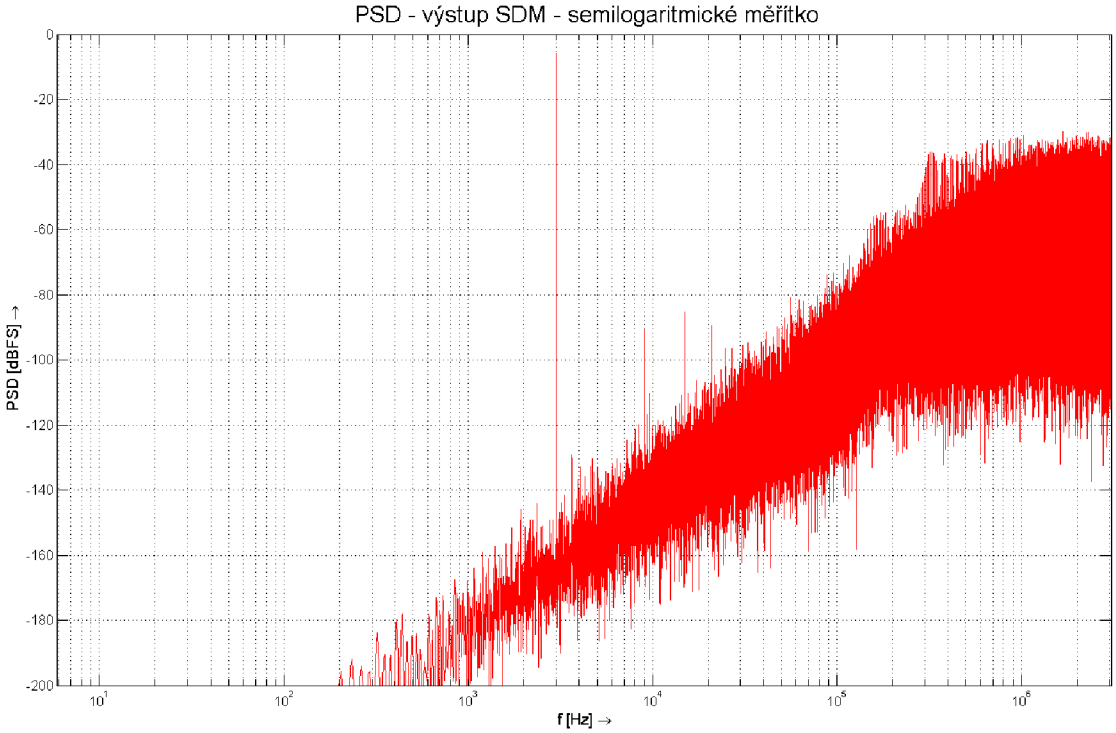
```
a     = [0.25    0.1875];
b     =  0.25;
c     = [0.28125 5.0   ];
g     =  0;
```

Zdrojový kód 4 - hodnoty koeficientů pro strukturu CRFB 2. Řád po kvantování

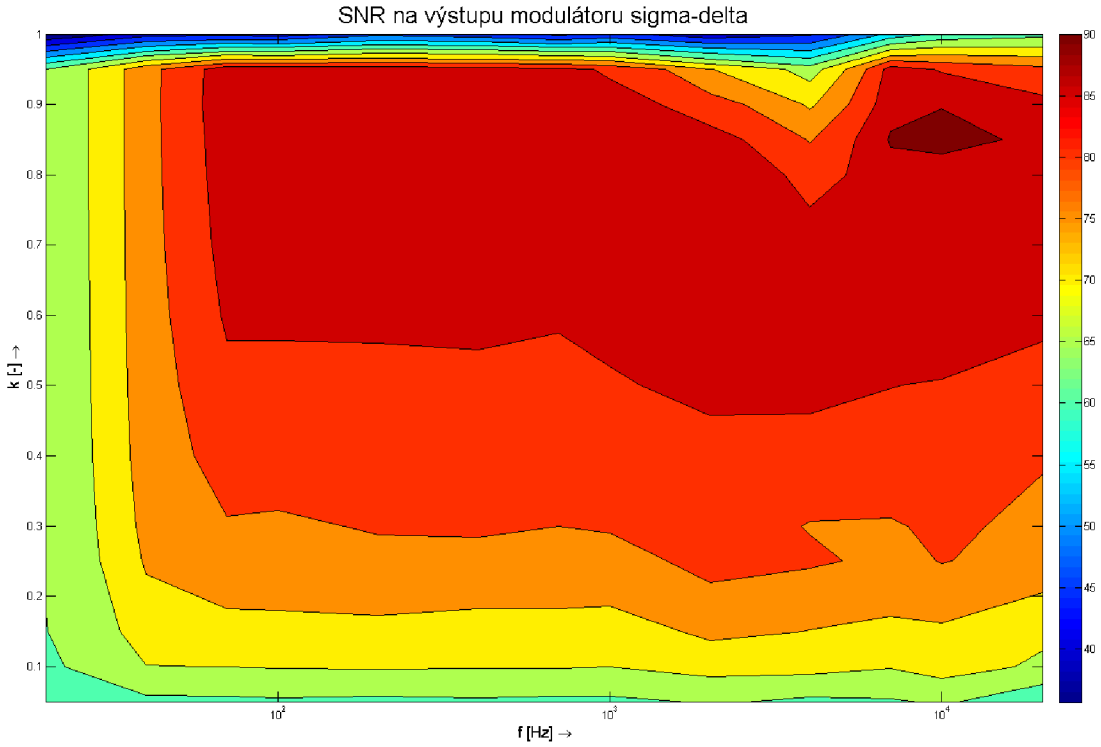
Pro finální ověření byl využit skript „Script_22_SNR_Screening“ který na rozdíl od skriptu „Script_12_SNR_Screening“ vyhodnocuje také výstupní signál z rekonstrukčního filtru. Jeho výstupy zachycuje Obrázek 31 a Obrázek 32.



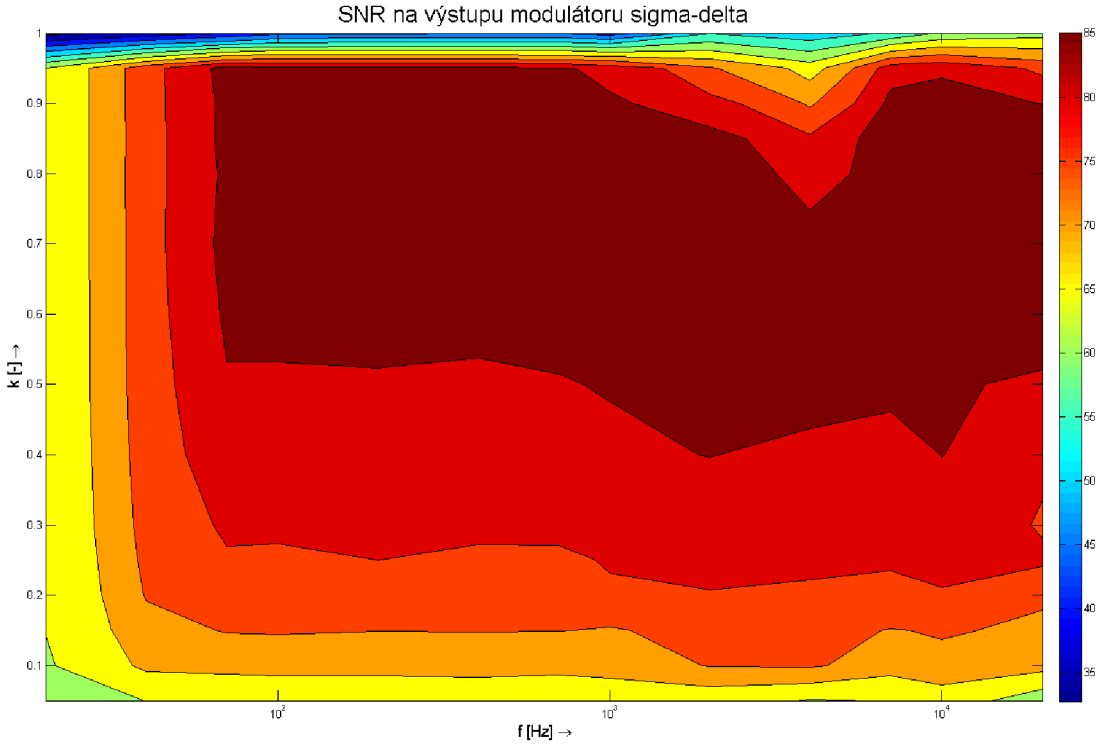
Obrázek 26 - spektrální výkonový hustota výstupního signálu modulátoru ERFB, 2. řád, pro vstupní sinusový signál s amplitudou 0.9*Fs/2 a frekvencí 3 kHz



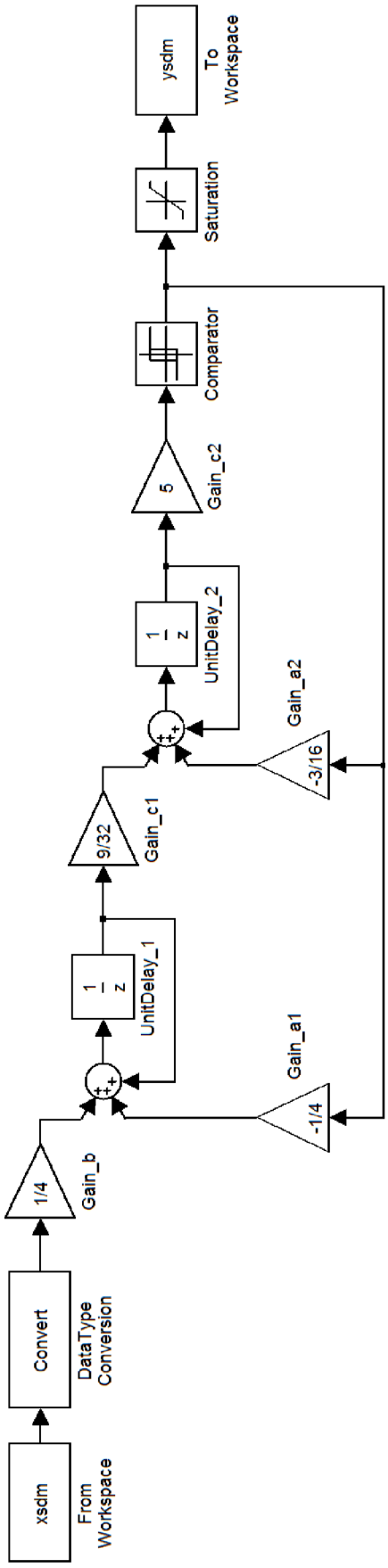
Obrázek 27 - spektrální výkonový hustota výstupního signálu modulátoru ERFB, 2. řád, pro vstupní sinusový signál s amplitudou 0.9*Fs/2 a frekvencí 3 kHz



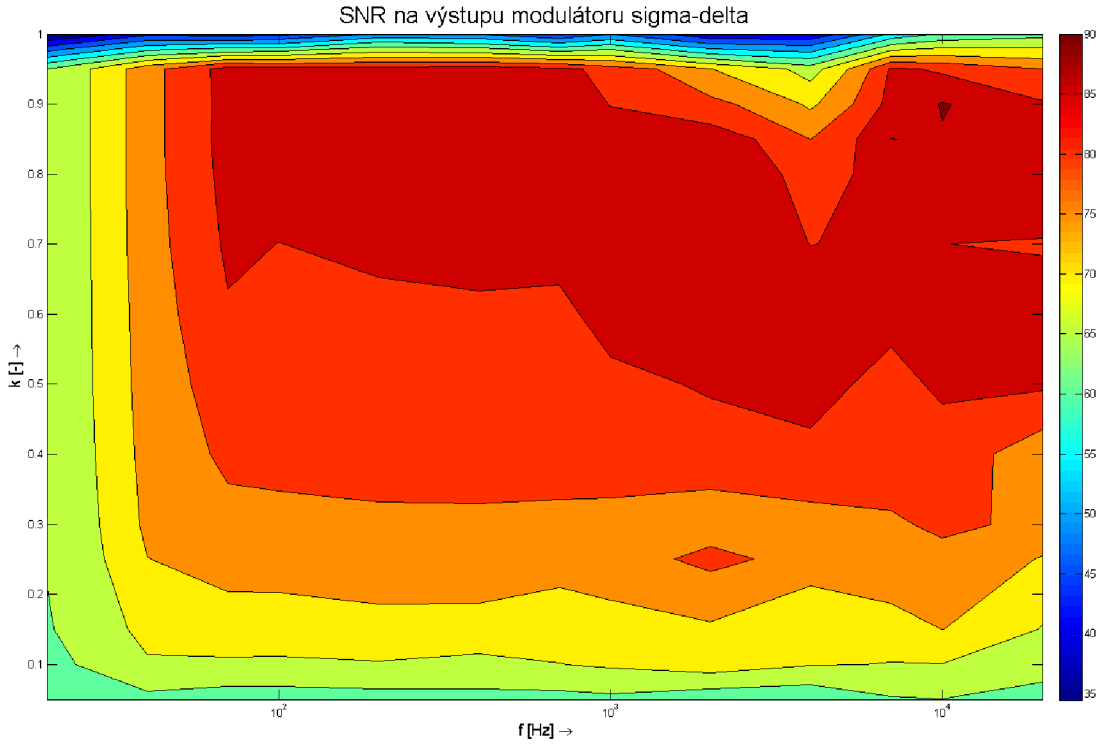
Obrázek 28 - velikost SNR na výstupu modulátoru sigma-delta v závislosti na amplitudě a frekvenci vstupního signálu - struktura CIFB



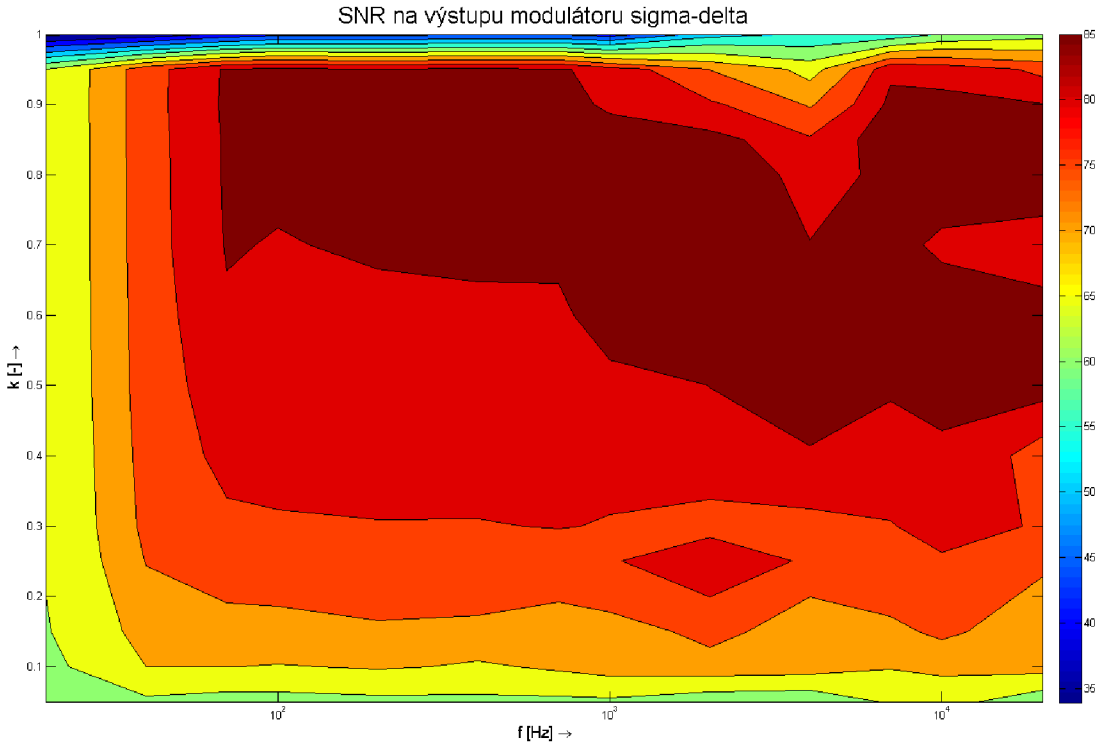
Obrázek 29 - velikost SNR na výstupu modulátoru sigma-delta pro v závislosti na amplitudě a frekvenci vstupního signálu - struktura ERFB



Obrázek 30 - model digitálního modulátoru sigma-delta, struktura CIFB 2. řád - modifikace pro aritmetiku s pevnou řádovou čárkou (Simulink)



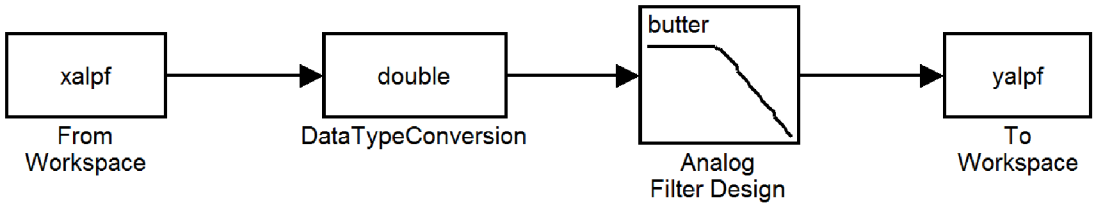
Obrázek 31 - velikost SNR na výstupu modulátoru sigma-delta v závislosti na amplitudě a frekvenci vstupního signálu - struktura CIFB - řetězec převodníku modifikován pro aritmetiku s pevnou řádovou čárkou



Obrázek 32 - velikost SNR na výstupu modulátoru sigma-delta v závislosti na amplitudě a frekvenci vstupního signálu - struktura ERFB - řetězec převodníku modifikován pro aritmetiku s pevnou řádovou čárkou

3.4 Analogový rekonstrukční filtr

Posledním blokem řetězce digitálně-analogového převodníku typu sigma-delta je analogový rekonstrukční filtr typu dolní propust. V rámci analýzy zadání byla zvolena butterworthova aproximace. Mezní kmitočet filtru odpovídá šířce pásma zpracovávaného signálu. Jedinou neznámou je tedy řád filtru. Řád by měl být takový, aby součet modulové přenosové charakteristiky rekonstrukčního filtru a přenosové funkce šumu modulátoru byl nižší než požadovaný odstup signál šum. Vzhledem k vysoké pracovní frekvenci modulátoru sigma delta byl pro první přiblížení během simulací využíván filtr 4 řádu, který se jevil jako dostatečný. Při simulacích převodníku byl analogový filtr realizován pomocí bloku Analog Filter Design, jak zachycuje Obrázek 33.



Obrázek 33 - model bloku analogového rekonstrukčního filtru (Simulink)

4 Popis převodníku v jazyce VHDL

4.1 Interpolátor

Interpolátor je tvořen kaskádním spojením tří interpolačních FIR filtrů, které byly navrženy v prostředí Matlab s využitím nástroje „filterbuilder“. Dalším krokem je vytvoření jejich VHDL popisu. K tomuto účelu byl využit HDL generátor prostředí Matlab. Výpisy Zdrojový kód 5, Zdrojový kód 6 a Zdrojový kód 7 dokumentují jeho konfiguraci, při generování VHDL popisu filtrů IFIR1, IFIR2 IFIR3. Vlastní interpolátor je popsán zdrojovým kódem, jenž zachycuje Zdrojový kód 7. Ten využívá vygenerované popisy filtrů a provádí jejich kaskádní spojení.

Ověření vlastností interpolátoru bylo provedeno prostřednictvím ověření vlastností jednotlivých interpolačních filtrů. K tomuto účelu byly při generování VHDL popisu filtrů vygenerovány také testovací kódy, pokrývající odezvu na jednotkový impuls, odezvu na jednotkový skok, rampový signá, chirp a šum. Ověření bylo provedeno v prostředí ModelSim.

```
%  ------------------------------------------------------------
%  HDL Code Generation Options:
%  TargetLanguage: VHDL
%  OptimizeForHDL: on
%  UseRisingEdge: on
%  Name: IFIR1
%  SerialPartition: 15
%  TestBenchName: IFIR1_tb
%  TestBenchStimulus: impulse step ramp chirp noise
%  GenerateHDLTestbench: on
%
%  Filter Settings:
%  Discrete-Time FIR Multirate Filter (real)
%  -----------------------------------------
%  Filter Structure      : Direct-Form FIR Polyphase Interpolator
%  Interpolation Factor  : 2
%  Polyphase Length      : 30
%  Filter Length         : 59
%  Stable                : Yes
%  Linear Phase          : Yes (Type 1)
%  Arithmetic            : fixed
%  Numerator             : s24,23 -> [-1 1)
%  Input                 : s15,0 -> [-16384 16384)
%  Filter Internals      : Specify Precision
%    Output              : s15,0 -> [-16384 16384)
%    Product             : s32,14 -> [-131072 131072)
%    Accumulator         : s32,14 -> [-131072 131072)
%    Round Mode          : convergent
%    Overflow Mode       : saturate
%  ------------------------------------------------------------
```

Zdrojový kód 5 - výpis nastavení HDL generátoru pro vytvoření popisu interpolačního FIR filtru IFIR1 (Matlab)

```
%  ------------------------------------------------------------
%  HDL Code Generation Options:
%  TargetLanguage: VHDL
%  OptimizeForHDL: on
%  UseRisingEdge: on
%  Name: IFIR2
%  SerialPartition: 10
%  TestBenchName: IFIR2_tb
%  TestBenchStimulus: impulse step ramp chirp noise
%  GenerateHDLTestbench: on
%
%  Filter Settings:
%  Discrete-Time FIR Multirate Filter (real)
%  -----------------------------------
%  Filter Structure      : Direct-Form FIR Polyphase Interpolator
%  Interpolation Factor  : 4
%  Polyphase Length      : 10
%  Filter Length         : 40
%  Stable                : Yes
%  Linear Phase          : Yes (Type 2)
%  Arithmetic            : fixed
%  Numerator             : s24,23 -> [-1 1)
%  Input                 : s15,0 -> [-16384 16384)
%  Filter Internals      : Specify Precision
%    Output              : s15,0 -> [-16384 16384)
%    Product             : s32,14 -> [-131072 131072)
%    Accumulator         : s32,14 -> [-131072 131072)
%    Round Mode          : convergent
%    Overflow Mode       : saturate
%  ------------------------------------------------------------
```

Zdrojový kód 6 - výpis nastavení HDL generátoru pro vytvoření popisu interpolačního FIR filtru IFIR2 (Matlab)

```
%  ------------------------------------------------------------
%  HDL Code Generation Options:
%  TargetLanguage: VHDL
%  OptimizeForHDL: on
%  UseRisingEdge: on
%  Name: IFIR3
%  SerialPartition: 7
%  TestBenchName: IFIR3_tb
%  TestBenchStimulus: impulse step ramp chirp noise
%  GenerateHDLTestbench: on
%
%  Filter Settings:
%  Discrete-Time FIR Multirate Filter (real)
%  -----------------------------------
%  Filter Structure      : Direct-Form FIR Polyphase Interpolator
%  Interpolation Factor  : 16
%  Polyphase Length      : 7
%  Filter Length         : 98
%  Stable                : Yes
%  Linear Phase          : Yes (Type 2)
%  Arithmetic            : fixed
%  Numerator             : s24,23 -> [-1 1)
%  Input                 : s15,0 -> [-16384 16384)
%  Filter Internals      : Specify Precision
%    Output              : s15,0 -> [-16384 16384)
%    Product             : s32,14 -> [-131072 131072)
%    Accumulator         : s32,14 -> [-131072 131072)
%    Round Mode          : convergent
%    Overflow Mode       : saturate
%  ------------------------------------------------------------
```

Zdrojový kód 7 - výpis nastavení HDL generátoru pro vytvoření popisu interpolačního FIR filtru IFIR3 (Matlab)

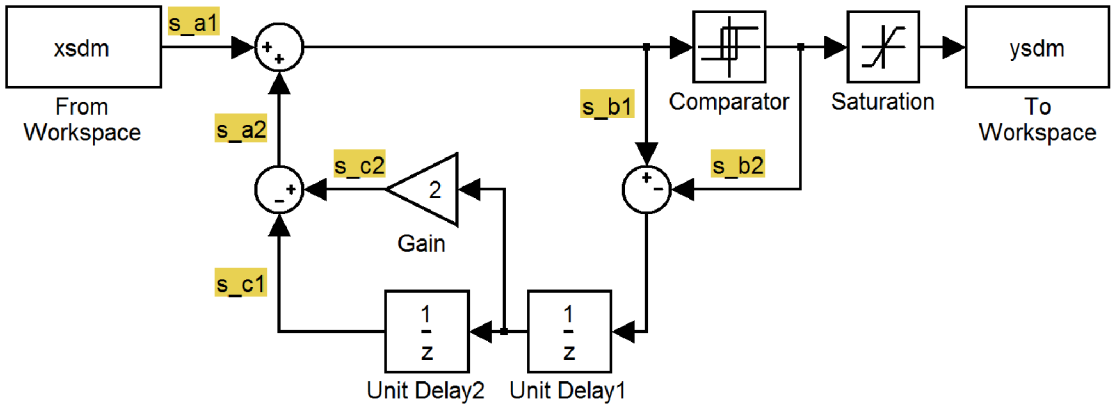
```
----------------------------------------------------------------------------------
-- Engineer:              Soukup
-- Module Name:           Interpolator - STRUCTURAL
-- Project Name:          DAC
-- Revision:              1.00
----------------------------------------------------------------------------------
library ieee;
use ieee.std_logic_1164.all;
----------------------------------------------------------------------------------
entity Interpolator is
   port( clk              :   in    std_logic;
         clk_enable       :   in    std_logic;
         reset            :   in    std_logic;
         filter_in        :   in    std_logic_vector(14 downto 0);
         filter_out       :   out   std_logic_vector(14 downto 0);
         ce_out           :   out   std_logic);
end Interpolator;
----------------------------------------------------------------------------------
architecture Structural of Interpolator is

   signal  connection_1:   std_logic_vector(14 downto 0) := (others => '0');
   signal  connection_2:   std_logic_vector(14 downto 0) := (others => '0');
   signal  ce_out_vector:  std_logic_vector( 1 downto 0) := (others => '0');

begin

   FIR1:   entity IFIR1(rtl)
      port map(
           clk             => clk,
           clk_enable      => clk_enable,
           reset           => reset,
           filter_in       => filter_in,
           filter_out      => connection_1,
           ce_out          => ce_out_vector(0));

   FIR2:   entity IFIR2(rtl)
      port map(
           clk             => clk,
           clk_enable      => ce_out_vector(0),
           reset           => reset,
           filter_in       => connection_1,
           filter_out      => connection_2,
           ce_out          => ce_out_vector(1));

   FIR3:   entity IFIR3(rtl)
      port map(
           clk             => clk,
           clk_enable      => ce_out_vector(1),
           reset           => reset,
           filter_in       => connection_2,
           filter_out      => filter_out,
           ce_out          => ce_out);

end Structural;
----------------------------------------------------------------------------------
```

Zdrojový kód 8 - popis struktury interpolátoru (VHDL)

4.2 Modulátor sigma delta

Z výsledků simulací prováděných během procesu návrhu vyplývá, že vlastnosti struktury ERFB jsou velmi podobné navržené struktuře CIFB. Vzhledem k tomu, že struktura ERFB má jednodušší vnitřní strukturu, je pro implementaci vhodnější, neboť pro její implementaci bude potřebná menší plocha.

Zdrojový kód 9 zachycuje VHDL popis ERFB sigma-delta modulátoru. Obrázek 34 ilustruje metodiku značení signálů uvnitř struktury modulátoru. Ověření VHDL popisu modulátoru jsem provedl pomocí porovnání výsledků simulací v prostředích Matlab a ModelSim.



Obrázek 34 - metodika pojmenování signálů uvnitř modulátoru

```
--------------------------------------------------------------------------------
-- Engineer:                Soukup
-- Module Name:             SigmaDeltaModulator - STRUCTURAL
-- Project Name:            DAC
-- Revision:                1.00
--------------------------------------------------------------------------------
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_signed.all;
--------------------------------------------------------------------------------
entity SigmaDeltaModulator is
   port( clk              :   in    std_logic;
         clk_enable       :   in    std_logic;
         reset            :   in    std_logic;
         sdm_in           :   in    std_logic_vector(14 downto 0);
         sdm_out          :   out   std_logic);
end SigmaDeltaModulator;
--------------------------------------------------------------------------------
architecture Structural of SigmaDeltaModulator is
   signal  s_a1:           std_logic_vector (39 downto 0) := (others => '0');
   signal  s_a2:           std_logic_vector (39 downto 0) := (others => '0');
   signal  s_b1:           std_logic_vector (39 downto 0) := (others => '0');
   signal  s_b2:           std_logic_vector (39 downto 0) := (others => '0');
   signal  s_c1:           std_logic_vector (39 downto 0) := (others => '0');
   signal  s_c2:           std_logic_vector (39 downto 0) := (others => '0');
   signal  unitdelay1_in:  std_logic_vector (39 downto 0) := (others => '0');
   signal  unitdelay1_out: std_logic_vector (39 downto 0) := (others => '0');
   signal  unitdelay2_in:  std_logic_vector (39 downto 0) := (others => '0');
```

```
   signal  unitdelay2_out: std_logic_vector (39 downto 0) := (others => '0');
   signal  quantizer_out:  std_logic := '0';
begin
   s_a1(14 downto  0) <= sdm_in;
   a1:for i in 39 downto 15 generate
      s_a1(i) <= sdm_in(14);
   end generate;
   s_a2(39 downto  0) <= s_c2 - s_c1;
   s_b1(39 downto  0) <= s_a1 + s_a2;
   s_b2(14 downto  0) <= "100000000000000";
   b2:for i in 39 downto 15 generate
       s_b2(i) <= not(quantizer_out);
   end generate;
   s_c1(39 downto  0) <= unitdelay2_out;
   s_c2(39 downto  0) <= unitdelay1_out(38 downto  0) & "0";

   unitdelay1_in <= s_b1 -s_b2;
   unitdelay2_in <= unitdelay1_out;

   delay_1: process (clk, reset) begin
      if (reset = '1') then
         unitdelay1_out <= (others => '0');
      elsif rising_edge(clk) then
         if (clk_enable = '1') then
            unitdelay1_out <= unitdelay1_in;
         end if;
      end if;
   end process;

   delay_2: process (clk, reset) begin
      if (reset = '1') then
         unitdelay2_out <= (others => '0');
      elsif rising_edge(clk) then
         if (clk_enable = '1') then
            unitdelay2_out <= unitdelay2_in;
         end if;
      end if;
   end process;

   quantizer_out <= '1' when s_b1(39) = '0' else '0';

   sdm_out <= quantizer_out;

end Structural;
-------------------------------------------------------------------------------
```

Zdrojový kód 9 - popis struktury modulátoru sigma-delta (VHDL)

4.3 Spojení interpolátoru a modulátoru

Spojení obou dílčích digitálních bloků, tedy interpolátoru a modulátoru sigma-delta, jsem provedl prostřednictvím architektury „TopLevel“, následuje výpis jejího zdrojového kódu:

```
-- Engineer:              Soukup
-- Module Name:           TopLevel - STRUCTURAL
-- Project Name:          DAC
-- Revision:              1.00
----------------------------------------------------------------------------------
library ieee;
use ieee.std_logic_1164.all;
----------------------------------------------------------------------------------
entity TopLevel is
   port( clk            :   in    std_logic;
         clk_enable     :   in    std_logic;
         reset          :   in    std_logic;
         dac_in         :   in    std_logic_vector(14 downto 0);
         dac_out        :   out   std_logic);
end TopLevel;
----------------------------------------------------------------------------------
architecture Structural of TopLevel is
   signal connection_1:  std_logic_vector(14 downto 0) := (others => '0');
   signal ce_out:        std_logic := '0';
begin
   InterpStage: entity Interpolator (Structural)
      port map(
           clk            => clk,
           clk_enable     => clk_enable,
           reset          => reset,
           filter_in      => dac_in,
           filter_out     => connection_1,
           ce_out         => ce_out);

   SDMStage: entity SigmaDeltaModulator(Structural)
      port map(
           clk            => clk,
           clk_enable     => ce_out,
           reset          => reset,
           SDM_in         => connection_1,
           SDM_out        => dac_out);

end Structural;
```

Zdrojový kód 10 - popis struktury navrhovaného převodníku (VHDL)

Závěr

V rámci této diplomové práce jsem provedl návrh digitálního systému realizujícího funkci digitálně-analogového převodníku typu sigma-delta a analogového rekonstrukčního filtru typu dolní propust, pro tento převodník.

S využitím nástroje Simulink jsem vytvořil model navrženého systému a v prostředí MATLAB provedl simulace. Jejich úkolem bylo ověřit nejen funkci navrženého převodníku, ale také jeho parametry. Simulace potvrdily schopnost převodníku pracovat s rozlišením 14 bitů i schopnost převádět signály ve frekvenčním pásmu $(0 \div 20)$ kHz. Navržený digitální systém tedy svými parametry vyhovuje požadavkům, jež byly zadány vedoucím diplomové práce.

Dále jsem vytvořil popis digitálních subsystémů navrženého převodníku v jazyce VHDL a provedl dílčí ověření jejich chování. Posledním bodem zadání bylo provedení syntézy a ověření vlastností navrženého převodníku v prostředí Cadence RTL Compiler. Tento bod zadání se mi bohužel, vzhledem k neočekávaným komplikacím během závěrečné fáze ověřování modelu převodníku, nepodařilo splnit.

Seznam použitých zkratek a symbolů

A/D	Analogové-digitální
ADC	Analog to Digital Converter - analogové-digitální převodník
CAD	Computer Aided Design - třída počítačového softwaru, sloužící k podpoře návrhu
CMOS	Complementary MOS
D/A	Digitálně-analogový
DAC	Digital to Analog Converter - digitálně analogový převodník
DNL	Differential Non Linearity - diferenciální nelinearita
ENOB	Effective Number Of Bits - efektivní počet bitů
FS	Full Scale - plný rozsah
HDL	Hardware description Language - zkratka označující jazyky pro popis hardvaru
INL	Integral Non Linearity - integrální nelinearita
LSB	Least Significant Bit -bit s nejnižší váhou
MOS	Metal Oxide Semiconductor
MSB	Most Significant Bit - bit s nejvyšší váhou
NTF	Noise Transfer Function - přenosová funkce šumu
OSR	Oversampling Ratio - poměr převzorkování
PLD	Programmable Logic Device
PSD	Power Spectral Density - spektrální hustota výkonu
RTL	Register Transfer Logic
SFDR	Spurious Free Dynamic Range - dynamický rozsah bez parazitních složek
SINAD	Signal to Noise and Distortion Ratio - Odstup signál šum a zkreslení
SNDR	Signal to Noise and Distortion Ratio - Odstup signál šum a zkreslení
SNR	Signal to Noise Ratio - odstup signál šum
STF	Signal Transfer Function - přenosová funkce užitečného signálu
THD	Total Harmonic Distortion - celkové harmonické zkreslení
VHDL	Very High Speed Integrated Circuits HDL - jazyk pro popis hardwaru

Seznam použitých informačních zdrojů

[1] **Pinker, Jiří a Poupa, Martin.** *Číslicové systémy a jazyk VHDL.* Praha : BEN-technická literatura, 2006. ISBN: 80-7300-198-5.

[2] FPGA Design Flow. *VLSI-World.* [Online] [Citace: 11. 04 2011.] http://www.vlsi-world.com/content/view/28/47/1/2/.

[3] **Smith, Michael John Sebastian.** *Application-Specific Integrated Circuits.* Boston : ADDISON-WESLEY, 2001. ISBN: 0-201-50022-1.

[4] **van de Plassche, Rudy J.** *CMOS Integrated Analog-to-Digital and Digital-to-Analog Converters 2nd Edition.* Boston : Kluwer Academic Publisher, 2003. ISBN: 1-4020-7500-6.

[5] **Jan, Jiří.** *Číslicová filtrace, analýza a rekonstrukce signálů.* Brno : VUTIUM, 2002. ISBN: 80-214-2911-9.

[6] **Schreier, Richard a Temes, Gabor C.** *Understanding delta-sigma data converters.* Chichester : Wiley, 2005. ISBN: 0-471-46585-2.

[7] **Kester, Walt, [editor].** *Data Conversion Handbook.* Amsterdam; Boston : Elsevier : Newnes, 2005. ISBN: 0-7506-7841-0.

[8] **Maloberti, Franco.** *Data Converters.* Dordrecht : Springer, 2007. ISBN: 0-387-32485-2.

[9] **Bourdopoulos, George I., a další.** *Delta-Sigma Modulators : Modeling, Design and Applications.* London : Imperial College Press, 2003. ISBN: 1-86094-369-1.

[10] *Noise Transfer Function Design and Optimization for Digital Sigma-Delta Audio DAC.* **Lewandowski, Marcin.** místo neznámé : Warsaw Univerity of Technology, 2011. Archives of acoustics. stránky 87-108.

[11] *Interpolation Filters for Oversampled Audio DACs.* **Løkken, Ivar.** místo neznámé : Norwegian University of Science and Technology, 2006.

[12] *Interpolator in a Sigma-Delta Digital-to-Analog Converter.* **V. Puidokas, A. J. Marcinkevičius.** místo neznámé : ELECTRONICS AND ELECTRICAL ENGINEERING, 2009. Sv. 02. ISSN 1392 – 1215.

Seznam příloh

Příloha A – Elektronická verze textu diplomové práce

Příloha B – Zdrojové kódy programu Matlab a modely programu Simulink

Příloha C – Zdrojové kódy v jazyce VHDL a skripty programu ModelSim