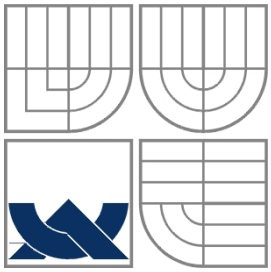
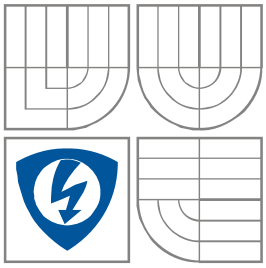


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV RADIOELEKTRONIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF RADIO ELECTRONICS

REALIZACE OFDM KODÉRU PRO POTŘEBY DVB-T

REALIZATION OF OFDM CODER FOR DVB-T SYSTEM

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — Bc. Petr Zelinka
AUTHOR

VEDOUCÍ PRÁCE — Ing. Tomáš Frýza, Ph.D.
SUPERVISOR

BRNO, 2008

VYSOKÉ UČENÍ
TECHNICKÉ V BRNĚ
Fakulta elektrotechniky
a komunikačních technologií
Ústav radioelektroniky

Diplomová práce

magisterský navazující studijní obor
Elektronika a sdělovací technika

Student: Zelinka Petr, Bc. ***ID:*** 89493
Ročník: 2 ***Akademický rok:*** 2007/08

NÁZEV TÉMATU:

Realizace OFDM kodéru pro potřeby DVB-T

POKYNY PRO VYPRACOVÁNÍ:

Prostudujte možnosti softwarové realizace kodéru a dekodéru modulačního řetězce OFDM na vývojové desce se signálovým procesorem řady TMS320C6000. Popište jednotlivé bloky kodéru, přičemž uvažujte možnost zpracování reálného signálu.

V prostředí Code Composer Studio (v jazyce C) naprogramujte funkci dílčích bloků realizujících kodér a dekodér OFDM v základním pásmu. Správnou funkci jednotlivých bloků odsimulujte.

Na základě simulací zkompletujte zdrojový kód celého kodéru a dekodéru OFDM. Zaměřte se na optimalizaci kódu z hlediska velikosti a rychlosti. Proveďte testování celého zařízení na vývojové desce Texas Instruments.

DOPORUČENÁ LITERATURA:

[1] Wikipedia. Orthogonal frequency-division multiplexing. [online]. 2007 - [cit. 25. září 2007]. Dostupné na WWW: http://en.wikipedia.org/wiki/OFDM

[2] Texas Instruments. Oficiální stránka firmy Texas Instruments. [online]. 2007 - [cit. 25. září 2007]. Dostupné na WWW: http://www.ti.com/

Termín zadání: 5.10.2007 ***Termín odevzdání:*** 30.5.2008

Vedoucí projektu: Ing. Tomáš Frýza, Ph.D.

prof. Dr. Ing. Zbyněk Raida
předseda oborové rady

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ Fakulta elektrotechniky a komunikačních technologií Ústav radioelektroniky

UPOZORNĚNÍ:

LICENČNÍ SMLOUVA
POSKYTOVANÁ K VÝKONU PRÁVA UŽÍT ŠKOLNÍ DÍLO

uzavřená mezi smluvními stranami:

1. Pan/paní

Jméno a příjmení:	Bc. Petr Zelinka
Bytem:	Zahraničního odboje 919, Třebíč, 674 02
Narozen/a (datum a místo):	24. prosince 1982 ve Zlíně

(dále jen „autor“)

a

2. Vysoké učení technické v Brně

Fakulta elektrotechniky a komunikačních technologií
se sídlem Údolní 53, Brno, 602 00
jejímž jménem jedná na základě písemného pověření děkanem fakulty:
prof. Dr. Ing. Zbyněk Raida, předseda rady oboru Elektronika a sdělovací technika
(dále jen „nabyvatel“)

Čl. 1

Specifikace školního díla

1. Předmětem této smlouvy je vysokoškolská kvalifikační práce (VŠKP):

   ☐ disertační práce
   ☒ diplomová práce
   ☐ bakalářská práce
   ☐ jiná práce, jejíž druh je specifikován jako .....
   (dále jen VŠKP nebo dílo)

Název VŠKP:	Realizace OFDM kodéru pro potřeby DVB-T
Vedoucí/ školitel VŠKP:	Ing. Tomáš Frýza, Ph.D.
Ústav:	Ústav radioelektroniky
Datum obhajoby VŠKP:	_____

   VŠKP odevzdal autor nabyvateli*:

   ☒ v tištěné formě – počet exemplářů: 2
   ☒ v elektronické formě – počet exemplářů: 2

2. Autor prohlašuje, že vytvořil samostatnou vlastní tvůrčí činností dílo shora popsané a specifikované. Autor dále prohlašuje, že při zpracovávání díla se sám nedostal do rozporu s autorským zákonem a předpisy souvisejícími a že je dílo dílem původním.

3. Dílo je chráněno jako dílo dle autorského zákona v platném znění.

4. Autor potvrzuje, že listinná a elektronická verze díla je identická.

* hodící se zaškrtněte

Článek 2

Udělení licenčního oprávnění

1. Autor touto smlouvou poskytuje nabyvateli oprávnění (licenci) k výkonu práva uvedené dílo nevýdělečně užít, archivovat a zpřístupnit ke studijním, výukovým a výzkumným účelům včetně pořizovaní výpisů, opisů a rozmnoženin.

2. Licence je poskytována celosvětově, pro celou dobu trvání autorských a majetkových práv k dílu.

3. Autor souhlasí se zveřejněním díla v databázi přístupné v mezinárodní síti

   - ☒ ihned po uzavření této smlouvy
   - ☐ 1 rok po uzavření této smlouvy
   - ☐ 3 roky po uzavření této smlouvy
   - ☐ 5 let po uzavření této smlouvy
   - ☐ 10 let po uzavření této smlouvy

   (z důvodu utajení v něm obsažených informací)

4. Nevýdělečné zveřejňování díla nabyvatelem v souladu s ustanovením § 47b zákona č. 111/ 1998 Sb., v platném znění, nevyžaduje licenci a nabyvatel je k němu povinen a oprávněn ze zákona.

Článek 3

Závěrečná ustanovení

1. Smlouva je sepsána ve třech vyhotoveních s platností originálu, přičemž po jednom vyhotovení obdrží autor a nabyvatel, další vyhotovení je vloženo do VŠKP.

2. Vztahy mezi smluvními stranami vzniklé a neupravené touto smlouvou se řídí autorským zákonem, občanským zákoníkem, vysokoškolským zákonem, zákonem o archivnictví, v platném znění a popř. dalšími právními předpisy.

3. Licenční smlouva byla uzavřena na základě svobodné a pravé vůle smluvních stran, s plným porozuměním jejímu textu i důsledkům, nikoliv v tísni a za nápadně nevýhodných podmínek.

4. Licenční smlouva nabývá platnosti a účinnosti dnem jejího podpisu oběma smluvními stranami.

V Brně dne: 30. května 2008

……..	……
Nabyvatel	Autor

Abstrakt

Obsahem této práce je rozbor standardu ETSI EN 300 744 pro pozemní digitální televizní vysílání (DVB-T) a dále popis vytvořeného kodéru a dekodéru OFDM pro přenos signálu v módu 2K v základním pásmu bez zabezpečení dat proti chybám. Funkce obou zařízení je ověřena simulacemi v Matlabu a realizována pomocí přípravků se signálovým procesorem TMS320C6711 firmy Texas Instruments.

Abstract

The contents of this thesis is a delineation of the European Standard ETSI EN 300 744 for terrestrial digital video broadcasting (DVB-T) and a description of created OFDM coder and decoder for baseband signal transmission in 2K mode without error correction capabilities. The proper function of both devices is verified by means of Matlab simulations and practically implemented into Texas Instruments' digital signal processor TMS320C6711 using Starter Kits.

Klíčová slova

DVB-T, OFDM, kodér, dekodér, DSP

Keywords

DVB-T, OFDM, coder, decoder, DSP

Bibliografická citace

ZELINKA, P. *Realizace OFDM kodéru pro potřeby DVB-T.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2008. 62 s. Vedoucí diplomové práce Ing. Tomáš Frýza, Ph.D.

Prohlášení

Prohlašuji, že svou diplomovou práci na téma Realizace OFDM kodéru pro potřeby DVB-T 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 30. května 2008 .....
podpis autora

Poděkování

Děkuji vedoucímu diplomové práce Ing. Tomáši Frýzovi, Ph.D. za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé diplomové práce.

V Brně dne 30. května 2008 .....
podpis autora

Obsah

Úvod

Rostoucí požadavky diváků na kvalitu a rozsah služeb poskytovaných televizními společnostmi vedly k rozmachu digitálního vysílání v mnoha zemích světa a v nedaleké budoucnosti čeká kompletní přechod od analogového k digitálnímu standardu i Českou republiku.

Český telekomunikační úřad ve svém „Technickém plánu přechodu zemského analogového televizního vysílání na zemské digitální televizní vysílání“ (opatření OOP/15/XX.2006-Y) stanoví pro potřeby digitálního vysílání televizního signálu v České republice systém DVB-T (digital video broadcasting – terrestrial) s modulací OFDM (orthogonal frequency division multiplexing) podle standardu ETSI EN 300 744 a doporučení ITU-R, BT.1306-3 a BT.1368-6.

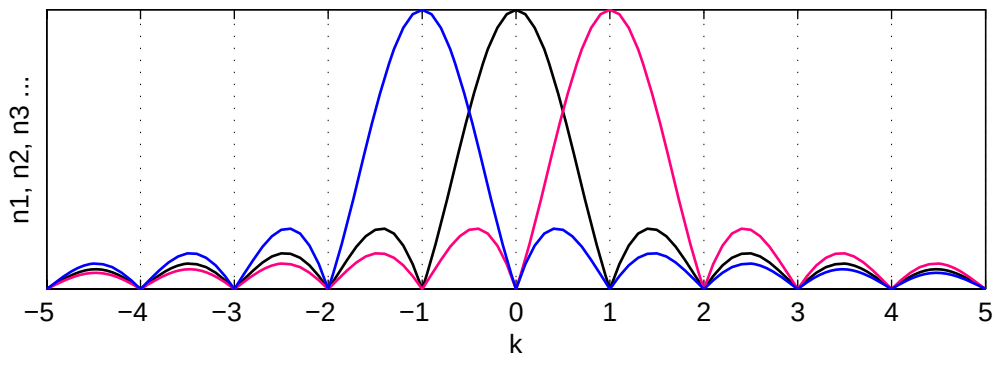
Cílem této práce je popis všech bloků kodéru DVB-T, jak je specifikuje uvedená norma, a dále praktická realizace základních bloků kodéru a dekodéru OFDM pro přenos signálu v základním pásmu s využitím vývojových desek se signálovým procesorem TMS320C6711 firmy Texas Instruments. Potřebné algoritmy jsou naprogramovány v jazyce C za použití knihoven optimalizovaných funkcí a ověřeny simulací v Matlabu. Program počítá s neideálním přenosovým kanálem, který je simulován AWGN (average white gaussian noise) kanálem případně s vřazeným filtrem pro simulaci frekvenčního zkreslení. Při praktické realizaci je propojení kodéru a dekodéru provedeno koaxiálními kabely s možností připojit analogový filtr typu dolní propust.

1 Základní principy přenosu DVB-T

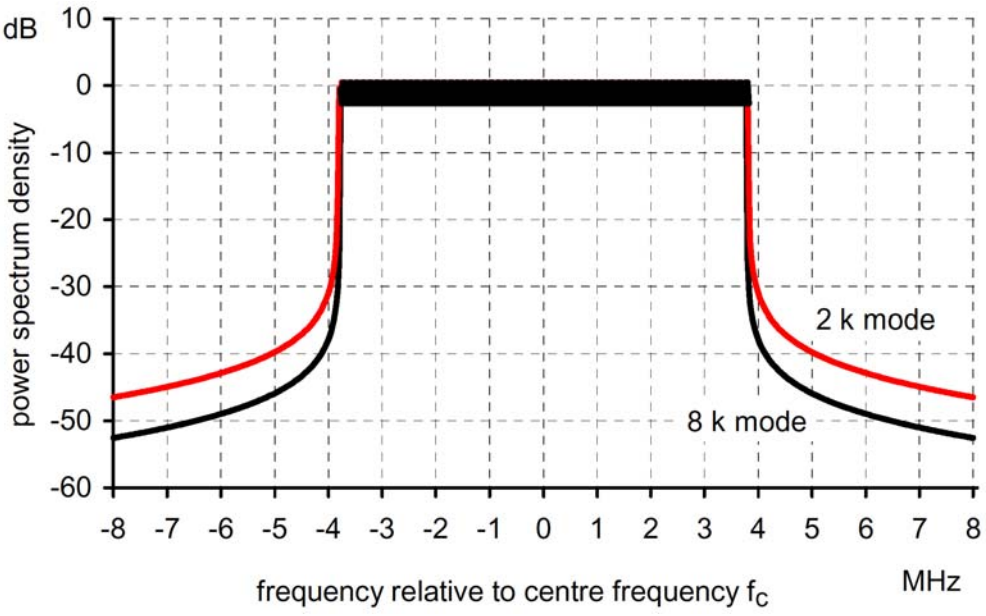
1.1 Vlastnosti modulace OFDM

Při vysílání digitálních signálů na jediném nosném kmitočtu v reálném komunikačním kanálu dochází k nežádoucímu ovlivňování přenášené informace vlivem šumu, impulzního rušení, interferencí s jinými zdroji vysokofrekvenčního signálu a také vlivem mnohacestného šíření. Při zvyšování přenosových rychlostí dochází k rozšiřování zabraného frekvenčního pásma, čímž se však výrazněji uplatní šum. Při pozemním vysílání signálů s vysokou přenosovou rychlostí, zejména pak tehdy, je-li délka jednoho signálového prvku srovnatelná s dobou šíření signálu od vysílače k přijímači, může vlivem odrazů docházet ke značnému zhoršení mezisymbolových interferencí ISI (inter symbol interferences). Tento efekt lze potlačit snížením přenosové rychlosti připadající na jeden nosný kmitočet, a tedy prodloužením symbolové periody. Má-li být zachována požadovaná přenosová kapacita, je ovšem nutné použít větší množství nosných vln. Při použití oddělených nosných se zavádí ochranné frekvenční intervaly, aby nedocházelo k vzájemnému ovlivňování. Toto však vede ke snížení efektivity využití daného frekvenčního pásma. Výrazného zvýšení spektrální efektivity lze dosáhnout použitím modulace OFDM.

Modulační princip OFDM je založen na využití většího množství nosných vln, které jsou navzájem ortogonální, tj. jejich skalární součin je nulový. Díky tomu je možné v přijímači jednotlivé nosné oddělit a získat tak informaci, kterou přenáší [2]. Klíčovým prvkem OFDM modulátoru je inverzní diskrétní fourierova transformace IDFT (inverse discrete Fourier transform), jejíž produkt představuje časovou reprezentaci vzájemně nekorelovaných frekvenčních složek, jejichž amplitudu a počáteční fázi lze ovlivňovat a vtisknout jim tak přenášenou informaci. Opačnou operaci, tedy diskrétní fourierovu transformaci DFT (discrete Fourier transform), pak lze chápat jako sadu korelátorů vstupního signálu s jednotlivými bázovými funkcemi, které jsou tvořeny sinusovkami na kmitočtech odpovídajících jednotlivým nosným. Každý korelátor vybere ze vstupního signálu jen tu část, která má nenulovou energii na kmitočtu odpovídající dané bázové funkci. Je tak umožněna jednoznačná dekódovatelnost přenášeného signálu, přestože spektra jednotlivých nosných kmitočtů se překrývají [3]. Na Obr. 1 je znázorněno překrývání spekter (k odpovídá frekvenčnímu intervalu nosných, n_x reprezentuje amplitudu jednotlivých nosných). V uzlových bodech má nenulovou energii vždy jen jedna nosná. Z Obr. 2 je patrné celkové spektrum jednoho kanálu DVB-T (pro 8 MHz kanál s ochranným intervalem $\Delta = T_U / 4$, kde T_U je délka užitečné části symbolu).



Obr. 1 Princip překrývání spekter nosných



Obr. 2 Požadované výsledné spektrum kanálu DVB-T (převzato z normy [1]).

Vzhledem k tomu, že každá nosná může nést m = 2, 4 nebo 6 bitů v jednom OFDM symbolu, jeden OFDM symbol tedy nese $K \cdot m$ bitů, kde K je počet nosných a m je počet bitů na jednu nosnou. Použitím OFDM modulace se tedy symbolová perioda zvýší $K \cdot m$-krát. Pokud vlivem odrazů přijde k přijímači jeden symbol zpožděn, dojde k ovlivnění jen malé části symbolu následujícího. Aby však bylo eliminováno i toto nežádoucí ovlivnění, je v systému zaveden tzv. ochranný interval, čímž je původní délka symbolu T_U prodloužena o Δ. Toto prodloužení je vyplněno cyklickým opakováním aktuálního symbolu. Díky dostatečně dlouhému Δ a precizní synchronizaci vysílačů je možné systém DVB-T provozovat v tzv. jednofrekvenční síti SFN (single frequency network), kdy všechny vysílače pracují ve stejném frekvenčním pásmu.

1.2 Varianty OFDM pro DVB-T

Norma [1] definuje několik variant OFDM použitelných pro systém DVB-T:

- varianta 2K: využívá 1705 nosných, spolu s definicí ochranných intervalů a způsobu zabezpečovacího kódování je určena zejména pro samostatné vysílače nebo pro SFN malého rozsahu,
- varianta 4K: určena specificky pro ruční terminály DVB-H (digital video broadcasting – handheld),
- varianta 8K: využívá 6817 nosných, umožňuje provoz samostatných vysílačů, SFN malého i velkého rozsahu. Jedná se o primární variantu pro uvažované použití v Evropě, jak uvádí [4].

Podle šířky využitého pásma se systémy DVB-T rozdělují:

- šířka pásma 8 MHz, primární varianta pro Evropu,
- 7 MHz, 6 MHz a 5 MHz systém, liší se pouze vzorkovacím kmitočtem výstupního signálu a s tím souvisejících časových parametrů (délka symbolu,

frekvenční rozestup nosných a možné bitové rychlosti). Norma nevylučuje použití i jiných šířek pásma na jiných než běžných televizních kmitočtech, pro využití v jiných částech světa.

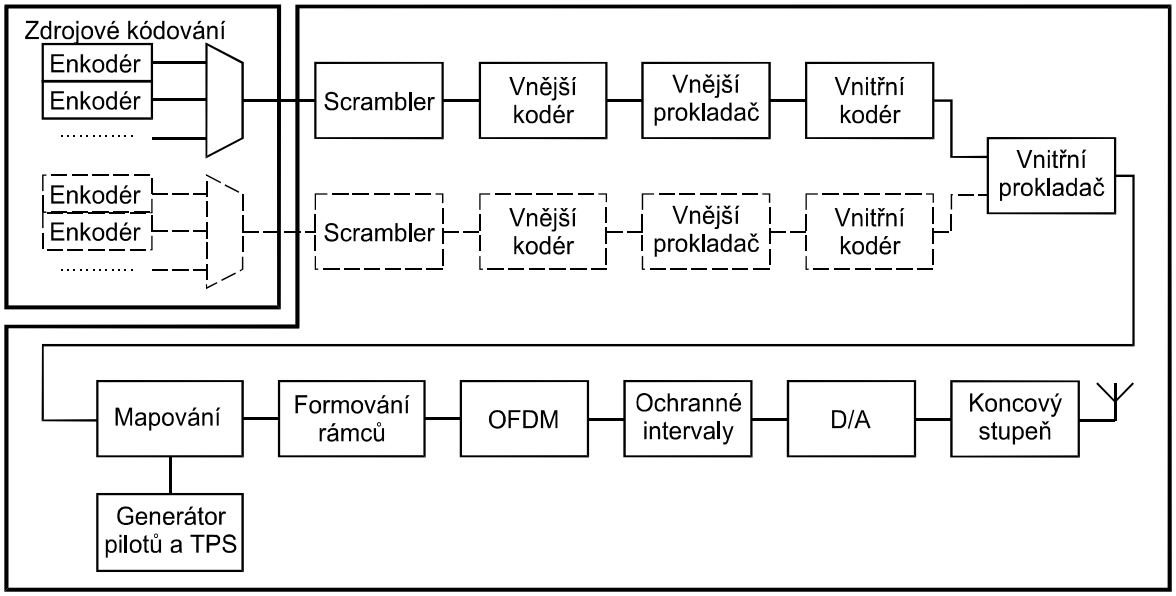
Dalším rozdělením je použití tzv. hierarchického nebo nehierarchického systému. U nehierarchického systému se přenáší jeden MPEG (moving picture experts group) multiplex s danou úrovní zabezpečení proti chybám. Použitím hierarchického systému je možné v jednom pásmu přenášet dva MPEG multiplexy, každý s jinou úrovní zabezpečení, přičemž jeden z nich je pak chápán jako prioritní. Je tedy možné přenášet stejný datový obsah s různou úrovní zabezpečení nebo použít pro každý vstup jiný multiplex, jak uvádí [5].

Norma též umožňuje volit různou úroveň zabezpečení proti chybám, což je dáno kódovým poměrem použitých kodérů, délkou ochranného intervalu a též způsobem modulace jednotlivých nosných. Mezi použitelné modulace patří: QPSK (quadrature phase shift keying, 2 bity na nosnou), 16-QAM (quadrature amplitude modulation, 4 bity na nosnou) a 64-QAM (6 bitů na nosnou). U QAM modulací existuje možnost volby mezi uniformním a neuniformním rozmístěním bodů v konstelačním diagramu.

2 Zpracování datového toku ve vysílačích DVB-T

2.1 Funkční bloky kodéru

Na Obr. 3 je blokové schéma vysílacího řetězce DVB-T. Základem pro digitální přenos televizního signálu je převod obrazových a zvukových dat do formy datového toku. Pro snížení redundance a irelevance se na zdrojová data (audio i video) aplikuje ztrátová komprese. Norma definuje použití algoritmu MPEG 2, v současnosti se již však prosazuje algoritmus MPEG-4 AVC. Výstupní datový tok z více kanálů se multiplexuje do jednoho výsledného toku, který je přiváděn na vstup COFDM (coded orthogonal frequency division multiplexing) modulátoru. Modulátor pak změní digitální signál do podoby vhodné pro vysílání (po D/A převodu, přeložení do vyššího frekvenčního pásma a zesílení).

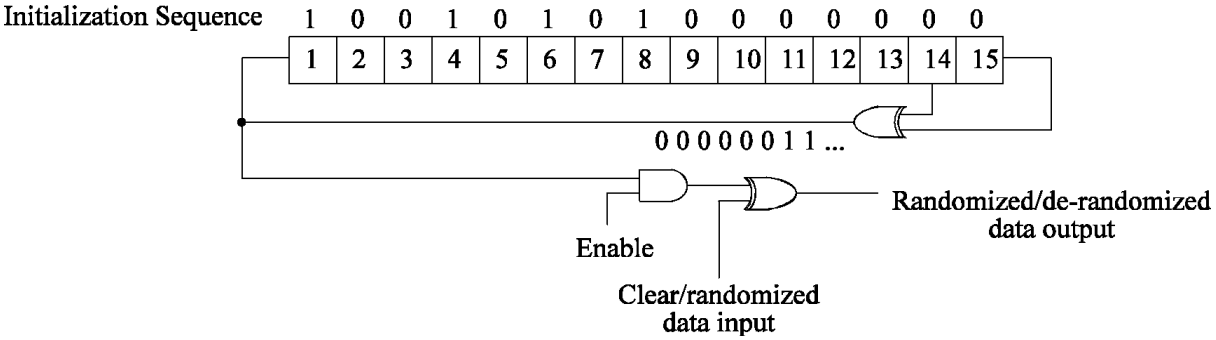


Obr. 3 Blokové vysílacího řetězce DVB-T (struktura dle normy[1])

V následujícím textu budou popsány jednotlivé části tohoto kodéru a následně i dekodéru.

2.2 Energetické rozprostření signálu

Na Obr. 4 je znázorněn pseudonáhodný binární generátor PRBS (pseudo random binary sequence) pro energetické rozprostření vstupního signálu (scrambling). Vstupní data jsou organizována v paketech pevné délky 188 bajtů, přičemž první bajt (tzv. synchronizační) má vždy hodnotu 47_{HEX}. Vstupní data se přivádí na vstup „Clear“, první jde nejvyšší bit synchronizačního bajtu a po něm následuje zbylých 187 informačních bajtů.



Obr. 4 Rozprostírací PRBS generátor (převzato z normy [1])

Na začátku každých 8 paketů se do registrů generátoru načte sekvence „0b100 1010 1000 0000“, přičemž v prvním paketu je synchronizační bajt invertován (tj. $B8_{HEX}$). Tento synchronizační bajt je použit rovnou pro výstup, tj. neprochází přes XOR člen generátoru. První scramblovaný je následující bajt 1. paketu. V době, kdy je přiváděn na vstup generátoru synchronizační bajt 2., 3. ... 8. paketu je signál „Enable“ přepnut do „0“, takže synchronizační bajty nejsou ovlivněny, ale registry generátoru se stále aktualizují. Při průchodu informačních bajtů je „Enable“ v „1“.

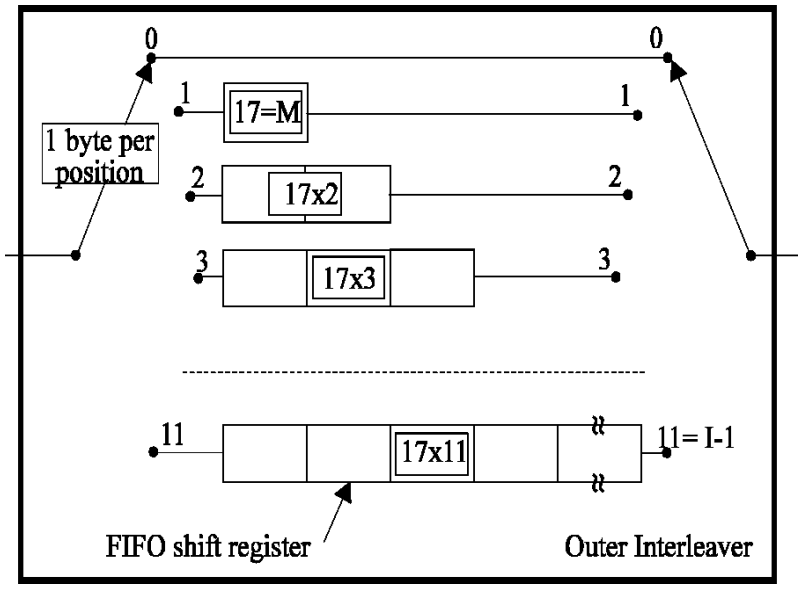
2.3 Vnější kódování

Pro vnější kodér je použit Reed-Slolomonův (204,188) kód, který umožňuje opravu až 8 bajtů v 204-bajtovém paketu. Kódovány jsou bloky po 188 bajtech sestavené ze sériových dat vycházejících z předchozího bloku (PRBS generátor).

2.4 Vnější prokládání

Vnější konvoluční bajtový prokladač je založen na Forneyově metodě [1] s hloubkou prokládání $I = 12$. Prokládání je aplikováno na jednotlivé bajty v rámci 204-bajtových paketů, takže je zachována periodicita po 204 bajtech ohraničených synchronizačními bajty.

Prokladač je složen z 12 větví (viz Obr. 5), každá obsahuje FIFO (first-in-first-out) registr s počtem buněk $j \cdot M$, kde j je pořadí větve ($j = 0, 1, 2 \ldots 11$) a $M = 17$ (odpovídá podílu počtu bajtů a hloubky prokládání). Na vstupu a výstupu prokladače jsou synchronizované přepínače, které cyklicky přepínají mezi jednotlivými větvemi.



Obr. 5 Vnější prokladač – funkční schéma (převzato z normy [1])

V Tab. 1 je ideově znázorněn systém prokládání jednotlivých bajtů v rámci 204-bajtového paketu. Jednotlivé bajty jsou do tabulky zapisovány ve směru řádků, čtení se provádí ve směru sloupců. Sousední bajty jsou tedy po proložení vzdáleny o 12 pozic, což je dáno hloubkou prokládání $I = 12$. Na výstupu prokladače se objeví sekvence:

1. bajt, 18. bajt, 35. bajt, ... 171. bajt, 188. bajt, 2. bajt, 19. bajt, ... 204. bajt.

Prvním bajtem v každém paketu je synchronizační bajt, pomocí nějž lze provést synchronizaci inverzního prokladače v přijímači. V přijímači se přijaté bajty poté zapisují do sloupců a čtou se z řádků, čímž se obnoví původní pořadí bajtů.

Tab. 1 Vnější prokladač - systém řazení bajtů v rámci jednoho paketu

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34
35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51
52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68
69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85
86	87	88	89	90	91	92	93	94	95	96	97	98	99	100	101	102
103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119
120	121	122	123	124	125	126	127	128	129	130	131	132	133	134	135	136
137	138	139	140	141	142	143	144	145	146	147	148	149	150	151	152	153
154	155	156	157	158	159	160	161	162	163	164	165	166	167	168	169	170
171	172	173	174	175	176	177	178	179	180	181	182	183	184	185	186	187
188	189	190	191	192	193	194	195	196	197	198	199	200	201	202	203	204

2.5 Vnitřní kódování

Základem vnitřního bitového kodéru je 64-stavový konvoluční kodér s kódovým poměrem $R = 1/2$. Pomocí punkturování je možné dosáhnout kódových poměrů 1/2, 2/3, 3/4, 5/6 a 7/8. Pro větev *A* je dán generující polynom: $G_1 = 1 + x + x^2 + x^3 + x^6$, pro větev *B* je

generující polynom: $G_2 = 1 + x^2 + x^3 + x^5 + x^6$. Norma [1] specifikuje punkturovací vzory pro každý z použitých kódových poměrů (viz Tab. 2).

Tab. 2 Punkturovací vzory a sekvence výstupních bitů

Kódový poměr *R*	Punkturovací vzor	Výstupní sekvence
1/2	X:1 Y:1	A_1B_1
2/3	X:10 Y:11	$A_1B_1B_2$
3/4	X:101 Y:110	$A_1B_1B_2A_3$
5/6	X:10101 Y:11010	$A_1B_1B_2A_3B_4A_5$
7/8	X:1000101 Y:1111010	$A_1B_1B_2B_3B_4A_5B_6B_7$

2.6 Vnitřní prokládání, mapování

Nejprve se provádí demultiplex sériových bitů do 2, 4 nebo 6 větví podle použité modulace (QPSK, 16-QAM, 64-QAM). Způsob řazení bitů shrnuje Tab. 3 (je uvedena pouze nehierarchická varianta DVB-T).

Tab. 3 Demultiplex vstupních bitů do větví

QPSK:	$x_0 \rightarrow b_0$ $x_1 \rightarrow b_1$
16-QAM:	$x_0 \rightarrow b_0$ $x_1 \rightarrow b_2$ $x_2 \rightarrow b_1$ $x_3 \rightarrow b_3$
64-QAM:	$x_0 \rightarrow b_0$ $x_1 \rightarrow b_2$ $x_2 \rightarrow b_4$ $x_3 \rightarrow b_1$ $x_4 \rightarrow b_3$ $x_5 \rightarrow b_5$

V Tab. 3 x_n odpovídá vstupním bitům a b_n odpovídá jednotlivým větvím. Postupným taktováním se do každé větve načte 126 bitů, na kterých se pak v rámci jedné větve provádí bitové prokládání. Celkem je tedy použito až 6 prokladačů (pro 64-QAM). Každý prokladač používá jinou permutační funkci.

Tab. 4 Permutační funkce pro prokladače v jednotlivých větvích

Prokladač:	Permutační funkce:
0	w
1	(w+63) mod 126
2	(w+105) mod 126
3	(w+42) mod 126
4	(w+21) mod 126
5	(w+84) mod 126

V Tab. 4 w odpovídá pořadí bitů v dané větvi (w = 0, 1, 2, ... 125), mod představuje funkci modulo, tj. zbytek po celočíselném dělení.

Všech v větví dohromady tvoří v-bitové datové slovo, celkem na jednu periodu prokladačů tak vznikne 126 slov. Následující blok provádí prokládání těchto slov pro jejich namapování na 1512 užitečných nosných (ze 1705 v 2K módu) respektive 6048 užitečných nosných (z 6817 v 8K módu). Pro dosažení odpovídajícího počtu symbolů se bere 12 skupin po 126 slovech pro 2K mód a 48 skupin po 126 slovech pro 8K mód.

Pro účely prokládání datových slov se zavádí permutační funkce $H(q)$, jejíž definice je uvedena v [1]. Tato funkce určuje záměnu pořadí datových slov, čímž vznikne výsledný vektor Y obsahující 1512 resp. 6048 v-bitových datových slov určených pro namapování do konstelačního diagramu pro jednotlivé nosné. Struktura konstelací pro všechny typy modulací (uniformní i neuniformní) je definována v normě [1]. Je využito Grayova mapování pro minimalizaci celkové chyby vzniklé při záměně sousedních konstelačních bodů vlivem neideálního příjmu.

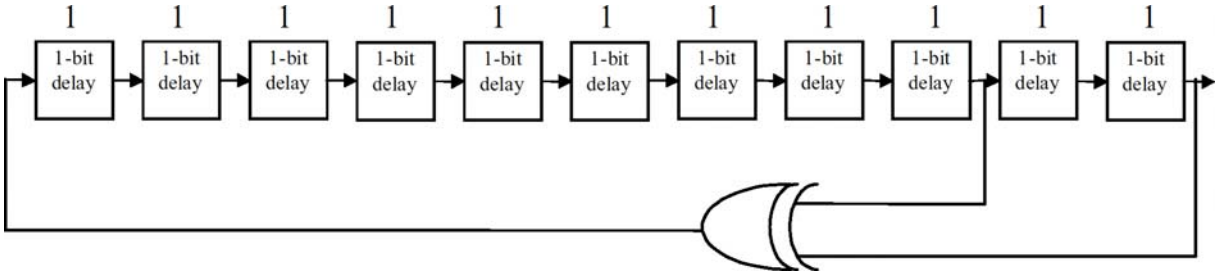
Komplexní obálka, tj. vysílaný signál v základním pásmu se získá zpětnou rychlou Fourierovou transformací IFFT (inverse fast Fourier transform) vstupních datových slov. Každý (komplexní) vstup IFFT odpovídá jedné nosné. Výstupem IFFT jsou časové vzorky komplexní obálky se vzorkovací periodou T, která závisí na použité šířce pásma (7/64 μs pro 8 MHz kanál, 1/8 μs pro 7 MHz kanál, 7/48 μs pro 6 MHz kanál a 7/40 μs pro 5 MHz kanál – vše při uvažování minimální použitelné šířky IFFT). Všechny vzorky z jedné transformace dohromady tvoří užitečnou část jednoho OFDM symbolu. Před přeložením na vysílací frekvenci se ještě ke každému OFDM symbolu přidává ochranný interval, který představuje vzorky z konce daného OFDM symbolu, které se umístí před jeho začátek. Délka ochranného intervalu se může měnit v několika hodnotách, jak uvádí norma [1].

2.7 Struktura rámců OFDM, pilotní signály

Každý rámec se skládá z 68 OFDM symbolů. Čtyři rámce pak tvoří jeden superrámec. Každý OFDM symbol je tvořen 6817 nosnými v 8K módu a 1705 nosnými v 2K módu. Daný počet nosných v každém symbolu je určen pro přenos speciálních, tzv. pilotních signálů. Ostatní nosné přenáší datovou informaci. Délka trvání jednoho symbolu je T_S a skládá se ze 2 částí: po dobu T_U se přenáší užitečná informace, po dobu Δ se přenáší cyklické opakování symbolu, čímž je tvořen ochranný interval.

Pilotní signály jsou využity v přijímači pro synchronizaci rámců, časovou synchronizaci, ekvalizaci, identifikaci vysílacího módu a případně i pro eliminaci fázového šumu. Existují 3 typy pilotních signálů: první dva, tzv. continual pilots (CP) a scattered pilots (SP), se liší jen způsobem alokace nosných a jsou modulovány sekvencí generovanou PRBS generátorem podle Obr. 6, třetí přenáší informaci o parametrech vysílání (transmission

parameters signaling, TPS). Výstup PRBS generátoru se mění pro každou nosnou (i těch, které nejsou pilotní) tak, že počáteční stav (0b111 1111 1111) odpovídá nejnižší nosné v daném OFDM symbolu. Piloty odpovídající CP a SP mají zvýšenou energetickou úroveň (boosted level), piloty TPS se přenáší na stejné energii jako data.



Obr. 6 PRBS generátor pro modulaci pilotů (převzato z normy [1])

TPS informace obsahují veškeré údaje pro správné nakonfigurování přijímače, takže může být po jejich dekódování schopen zpracovat všechny módy DVB-T. Tyto informace jsou přenášeny DPSK modulací v 68 bitech se zabezpečením pomocí BCH (Bose, Ray-Chaudhuri, Hocquenghem) kódu.

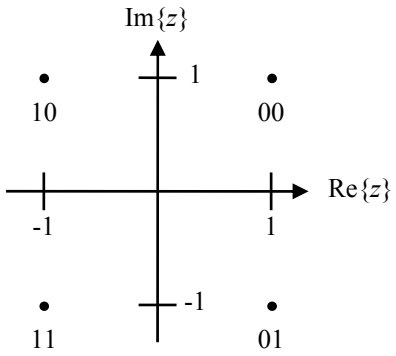
3 Navržený kodér OFDM, výsledky simulací

Pro navrhovaný systém byla z možných variant DVB-T zvolena nehierarchická verze 2K (jeden datový tok, jediná modulace nosných), která využívá 1705 nosných, tj. minimální šířka IFFT v kodéru je 2048, délka ochranného intervalu odpovídá ¼ IFFT (512 vzorků). Pro modulaci užitečných nosných byla zvolena QPSK a alternativně 16-QAM (uniformní), aby bylo možno studovat vliv lineárního zkreslení a šumu v komunikačním kanále na chybovost při rozdílné vzdálenosti konstelačních bodů (varianta 64-QAM nebyla implementována vzhledem k paměťovým a výpočetním nárokům). Vzorkovací frekvence by měla zajistit šířku pásma 8 MHz, tj. f_{vz}=9,1 MHz [1]. V navrženém kodéru nejsou použity bloky pro zabezpečení proti chybám (tj. prokladače a FEC - forward error correction kodéry) a neuvažuje se rámcová struktura.

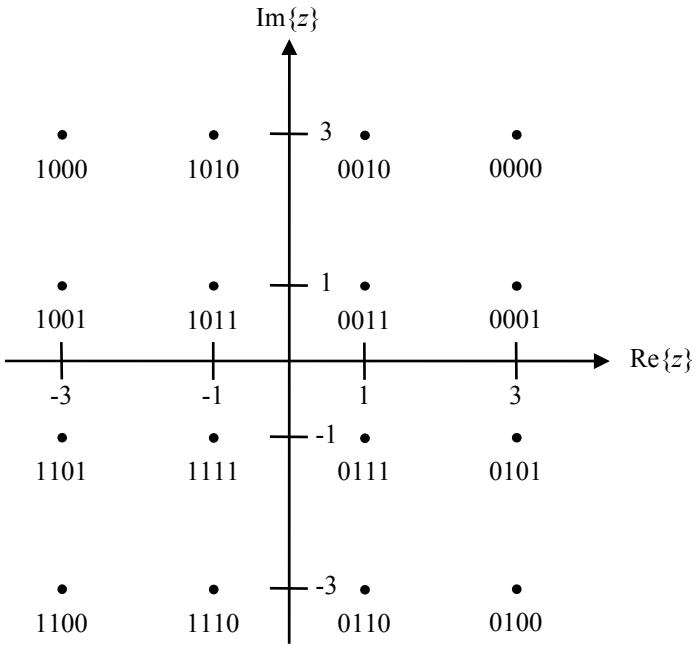
3.1 Mapování, modulace nosných

Vstupem kodéru je sériový datový tok, přičemž se uvažuje uspořádání bitů ve směru MSB-first, tj. první se zpracovává nejvyšší bit daného bajtu (v souladu s normou [1]).

V prvním kroku jsou vstupní bajty rozděleny na dibity (QPSK) nebo na skupiny po 4 bitech (16-QAM). Těmto skupinám je v souladu s Tab. 3 (viz kapitolu 2.6) přiřazen dibit (resp. 4-bit) b_n udávající pozici v konstelačním diagramu (viz Obr. 7, Obr. 8). Pro QPSK jsou výsledné komplexní hodnoty z normovány poděleným $\sqrt{2}$, při použití 16-QAM se z dělí $\sqrt{10}$. Tímto je zajištěno, že datové nosné mají energii E[z × z*] = 1.



Obr. 7 Konstelace QPSK (bity {b_0, b_1})

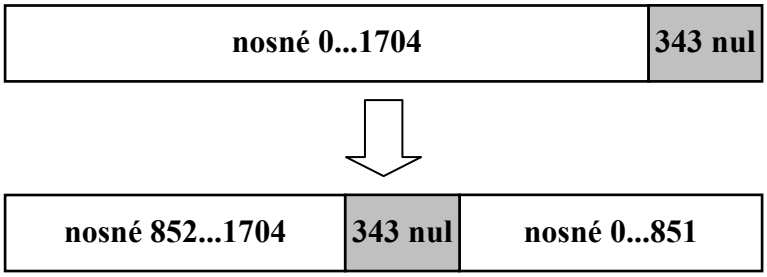


Obr. 8 Konstelace 16-QAM (bity $\{b_0, b_1, b_2, b_3\}$)

3.2 Vložení pilotních signálů, Fourierova transformace

Namodulované nosné jsou dále na předepsaných pozicích proloženy pilotními signály (viz kapitolu 2.7). Nosné odpovídající pilotům TPS jsou nastaveny na nulovou hodnotu, neboť nejsou v navrženém systému využity. Symbolová synchronizační funkce TPS není díky použitému algoritmu v dekodéru vyžadována a parametry přenosu jsou předem známy (jsou konstantně nastaveny); synchronizace rámců se neuvažuje.

Celkový počet nosných je po vložení pilotů 1705. Inverzní Fourierova transformace musí mít šířku větší, což při uvažování mocnin 2 znamená minimálně 2048. Každá nosná odpovídá jedné spektrální složce IFFT, nevyužité nosné se nastavují na nulu (na těchto frekvencích se nevysílá žádný signál). Vzhledem k cyklickému charakteru IFFT je nutné změnit uspořádání nosných, jak dokumentuje Obr. 9.



Obr. 9 Systém uspořádání vstupu IFFT (při šířce IFFT 2048)

Uvedené uspořádání zajistí, že nosná s prostředním indexem bude ve středu frekvenčního pásma výsledného signálu (0 Hz v základním pásmu) a nosné budou ve spektru uspořádány v přirozeném pořadí.

3.3 Úprava spektra signálu

Výkonové spektrum jednotlivých nosných OFDM signálu má tvar funkce sinc^2 a jejich superpozicí vzniká zdánlivě obdélníkové spektrum ve frekvenčním rozsahu $-(K - 1) / 2T_U$ až $(K - 1) / 2T_U$ [8]. Pomalé doznívání funkce sinc má za následek nezanedbatelnou úroveň výkonu signálu mimo nominální šířku pásma. Norma [1] definuje spektrální masku, tj. maximální hodnoty spektrální výkonové hustoty mimo nominální šířku pásma, které nesmí být překročeny (viz Tab. 5, graficky na Obr. 11).

Tab. 5 Spektrální maska 8 MHz kanálu

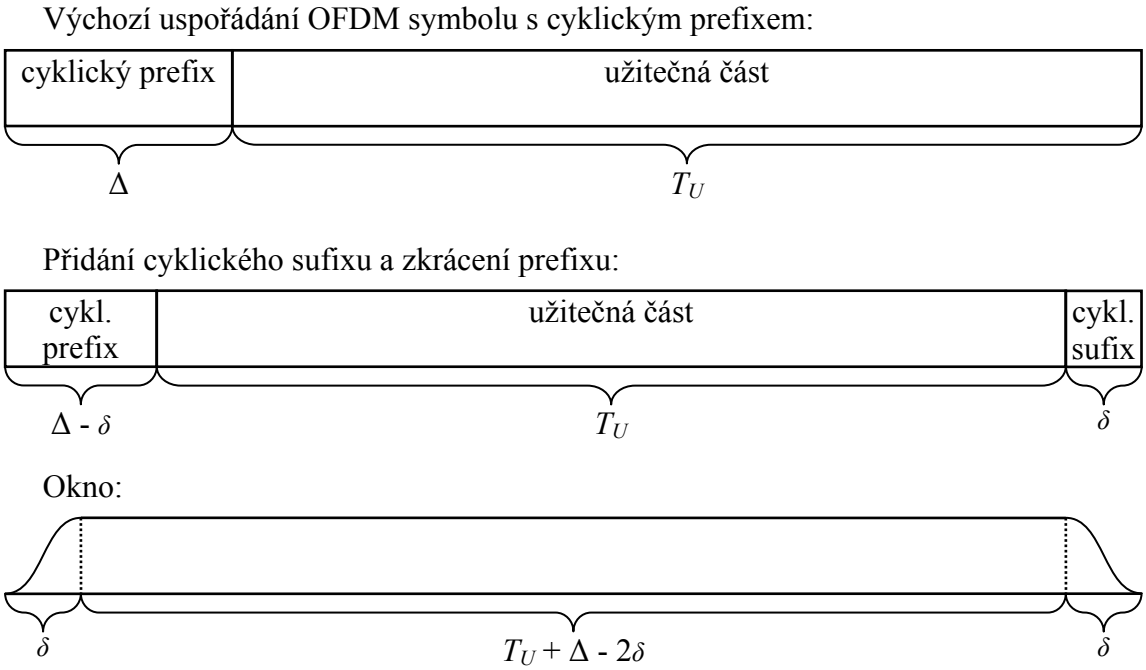
relativní frekvence [MHz]	relativní úroveň výkonu [dB]
-12	-100
-10,75	-76,9
-9,75	-76,9
-5,75	-74,2
-5,185	-60,9
-4,65	-56,9
-3,9	-32,8
+3,9	-32,8
+4,25	-64,9
+5,25	-76,9
+6,25	-76,9
+10,25	-76,9
+12	-100

Hodnoty frekvence v Tab. 5 se vztahují ke střednímu kmitočtu kanálu; hodnoty výkonu jsou vztaženy k celkovému výkonu (total output power), měří se v šířce pásma 4 kHz. Uvedená spektrální maska platí pro vysílač operující v prostředí s analogovými vysílači systémů G / PAL / NICAM se stejným výkonem. Pro dodržení uvedené masky je nutné provést vhodnou filtraci.

Použitý filtr pro úpravu spektra nesmí zavést nepřijatelné amplitudové ani fázové zkreslení v hlavním laloku OFDM spektra. V [8] je uveden příklad filtrace OFDM signálu pomocí Butterworthova filtru, který sníží úroveň nežádoucího vyzařování z -30 na -41 dB při použití filtru 5. řádu a na -51 dB při použití 10. řádu. Realizace takovéhoto filtru již představuje dosti náročný problém, a to jak při analogovém provedení (stálost parametrů, přesnost výroby apod.), tak i v softwarové podobě (nároky na výpočetní výkon).

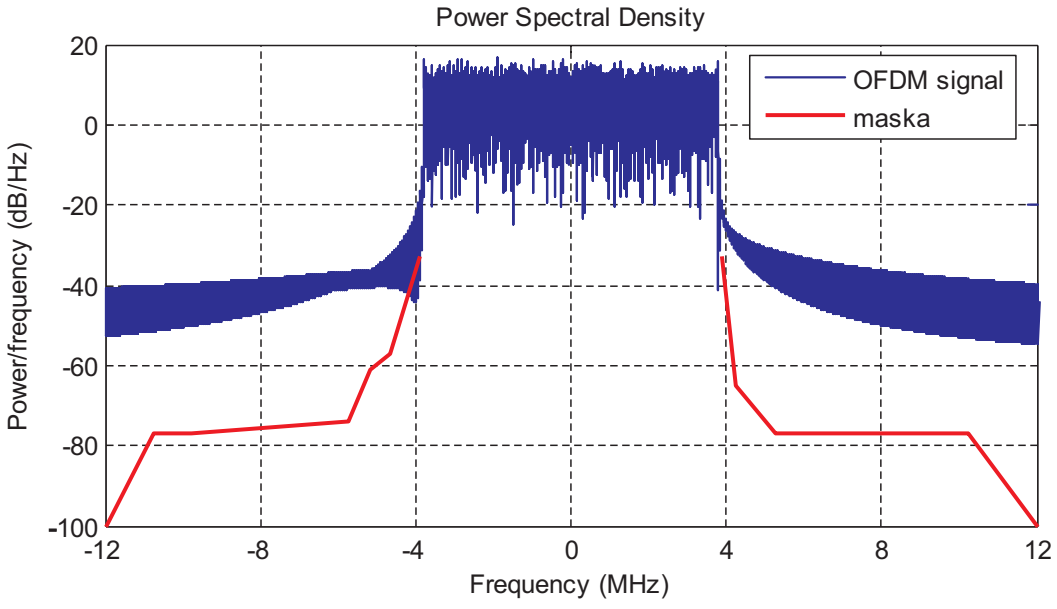
Existuje však výpočetně podstatně méně náročná metoda, která umožňuje dosáhnout výborného potlačení nežádoucích složek spektra. Je založena na tvarování původně obdélníkových pulsů nosných pomocí okénkování OFDM signálu v časové oblasti [8]. Vychází se z poznatku, že obdélníkový signál v časové oblasti se do spektra promítne jako funkce sinc, zatímco signál upravený (např. filtrem Raised Cosine) má ve spektru podstatně nižší úroveň bočních laloků. Předpokládejme OFDM signál složený z užitečné části o délce T_U a jí předcházejícího ochranného intervalu (cyklický prefix) o délce Δ. Aby nedošlo okénkováním ke zkreslení užitečné části, je provedeno její cyklické prodloužení na konci o délku δ. O stejnou délku musí být zkrácen ochranný interval Δ, aby nebyla narušena perioda

symbolů. Následuje pronásobení oknem, které je konstantní v intervalu $T_U + \Delta - 2\delta$. Mimo tyto hranice klesá k nule po dobu δ, jak je znázorněno na Obr. 10.

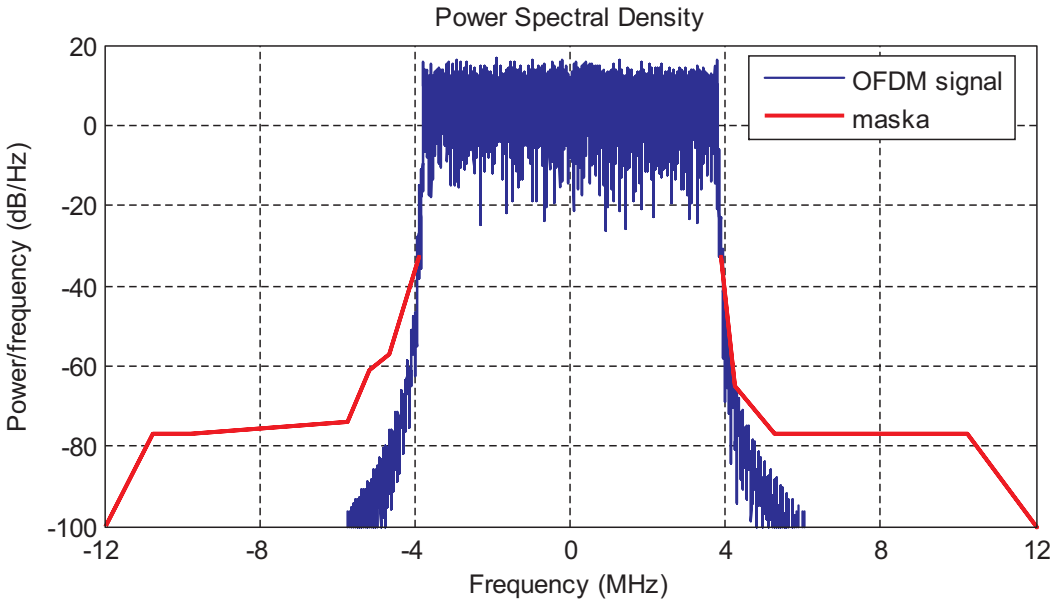


Obr. 10 Úprava tvaru OFDM symbolu

Plynule narůstající a klesající okraje odpovídají bokům Hanningova okna. Při prodlužování δ klesá výkon mimo nominální šířku pásma strměji. Zároveň se však zkracuje efektivní délka ochranného intervalu, takže systém se stává méně odolný vůči vícecestnému šíření. Výkonové spektrum nefiltrovaného signálu je zobrazeno na Obr. 11, po filtraci na Obr. 12. Díky použití filtru je dodržena spektrální maska.



Obr. 11 Spektrální výkonová hustota OFDM signálu bez filtrace



Obr. 12 Spektrální výkonová hustota OFDM signálu po úpravě oknem ($\delta = \Delta / 8$)

Digitální zpracování signálu je zakončeno převodem do analogové podoby, čímž vzniká signál s periodickým spektrem o periodě $f_{vz} = 1 / T$, kde f_{vz} je vzorkovací frekvence. Ve výsledném signálu musí být pouze složky v základním pásmu, použitý analogový filtr typu dolní propust tedy musí bez zkreslení propouštět kmitočty $|f| \leq K / 2T$ a potlačit vyšší periodická spektra, tj. $|f| \geq |f_{vz} - K / 2T|$ [8]. Tento požadavek klade na použitý analogový filtr značné nároky. Proto se někdy zavádí tzv. převzorkování, tedy použití IFFT s větší šířkou, než je nutné minimum. Tím dojde k posunu f_{vz} na násobek původní hodnoty a použitý filtr již nemusí mít tak velkou strmost. Cenou je však větší náročnost na výpočetní výkon a paměť digitální části systému. Praktickým řešením bývá obvykle jednoúčelový integrovaný obvod (oversampling circuit [8]). Aby byla zachována zamýšlená šířka pásma výstupního signálu, musí být v závislosti na použité šířce IFFT správně nastavena vzorkovací perioda výstupního signálu. Jak již bylo uvedeno v kapitole 2.6, odpovídá 8 MHz kanálu a 2048-IFFT vzorkovací perioda $T = 7/64$ µs. V závislosti na šířce IFFT bude výsledná vzorkovací perioda T' zřejmě

$$T' = T \cdot \frac{2048}{N}, \tag{1}$$

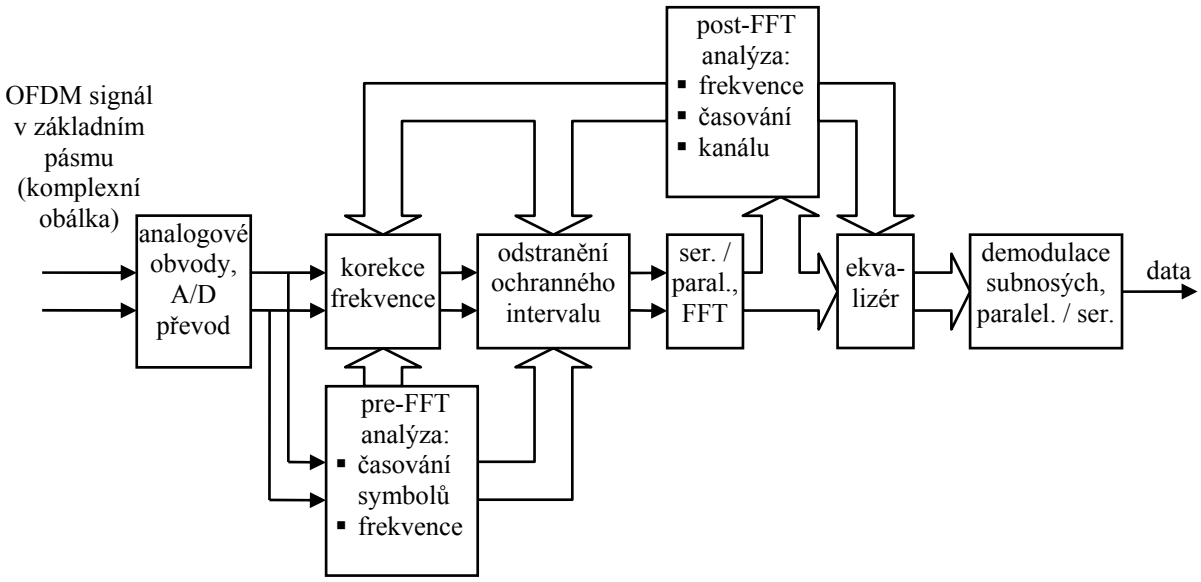
kde N je šířka použité IFFT.

4 Navržený dekodér OFDM, výsledky simulací

Příjem OFDM signálu sestává ze dvou částí: zachycení (acquisition) a sledování (tracking). Po zapnutí nemá přijímač informaci o časování vysílaného signálu a navíc bývá přítomen nezanedbatelný frekvenční ofset (např. nedokonalým nastavením oscilátorů ve směšovačích). Pro správnou demodulaci nosných je dále nutné korigovat frekvenční charakteristiku přenosového kanálu. Proto se nejprve určí přibližná synchronizace a následně se tyto odhady zpřesní s využitím informace obsažené v pilotních signálech, která rovněž umožní korekci frekvenční charakteristiky. Pilotních signálů se využívá po celou dobu příjmu (tracking) pro udržování správné časové a frekvenční koherence.

4.1 Struktura dekodéru

V principu by bylo možné provést veškerou synchronizaci před aplikací Fourierovy transformace pouze z časových parametrů signálu nebo naopak pouze vyhodnocením výstupu FFT. Tyto metody jsou však značně neefektivní, neboť bez použití FFT není k dispozici pilotní informace a naopak výlučná aplikace FFT je problematická kvůli ISI a též ICI (inter carrier interferences), které vykazuje nezasynchronizovaný signál. Proto se obvykle provádí část zpracování před FFT (hrubé odhady časového a frekvenčního ofsetu) a část po provedení FFT (jemné doladění a sledování). Na základě [13] bylo sestaveno blokové schéma dekodéru, viz Obr. 13.



Obr. 13 Blokové schéma OFDM dekodéru

V profesionálních přijímačích se ještě využívá bloku interpolace / decimace pro možnost korekce zlomkových velikostí časového ofsetu, v navrženém systému je však zvolena z důvodu výpočetní náročnosti pouze celočíselná hodnota korekce odpovídající vzorkovací periodě.

4.2 Zachycení vysílání (acquisition)

Mód acquisition trvá po dobu provádění prvotní synchronizace přijímače. Po dokončení jemné časové korekce po FFT se přechází do módu tracking.

4.2.1 Korelace ochranných intervalů, hrubé určení FFT okna

Pro hrubý odhad pozice začátku užitečné části OFDM symbolu v přijatém signálu se využívá výhodných autokorelačních vlastností daných přítomností cyklického prefixu [8]. Pro vstupní signál $s(t)$ platí

$$s(t) = s(t + T_U) \tag{2}$$

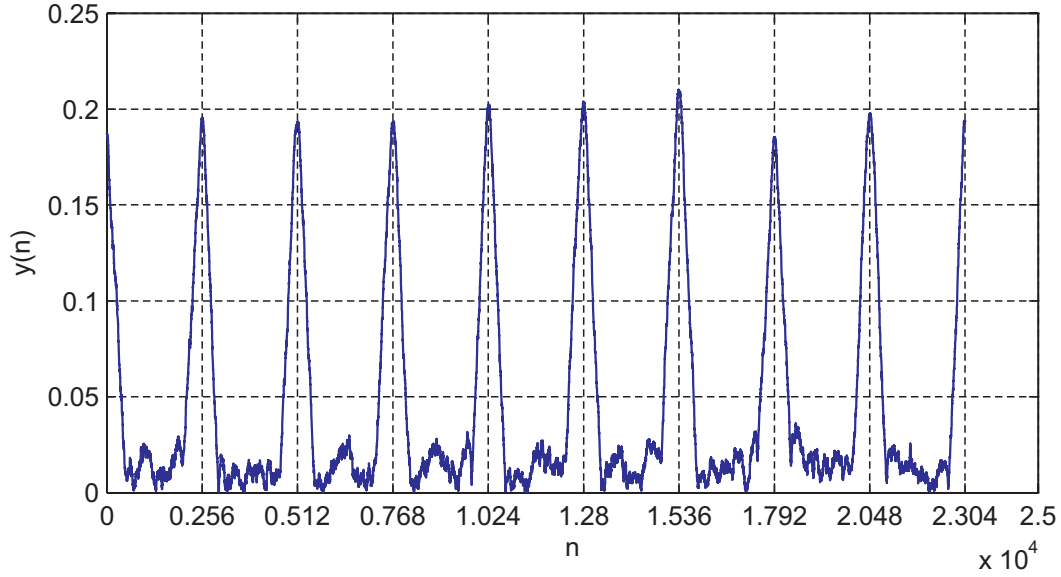
v intervalu $-\Delta \leq t \leq 0$, kde t je čas při uvažování $t = 0$ na začátku užitečné části symbolu. Koreluje se přes délku ochranného intervalu Δ, tj.

$$y(t) = \frac{1}{\Delta} \int_{t-\Delta}^{t} s(\tau) s^*(\tau + T_U) \mathrm{d}\tau \,. \tag{3}$$

Zpracovávaný signál je diskrétní, integrál tedy přejde v sumaci

$$y(n) = \sum_{i=0}^{N_G - 1} s(i) s^*(i + N_S) \,, \tag{4}$$

kde N_S je počet vzorků odpovídající jednomu symbolu a N_G je počet vzorků odpovídající délce ochranného intervalu. Na Obr. 14 je zobrazen modul výstupu korelátoru pro 10 symbolů. Navržený systém pracuje s pevnou délkou Δ, takže stačí jeden korelátor.



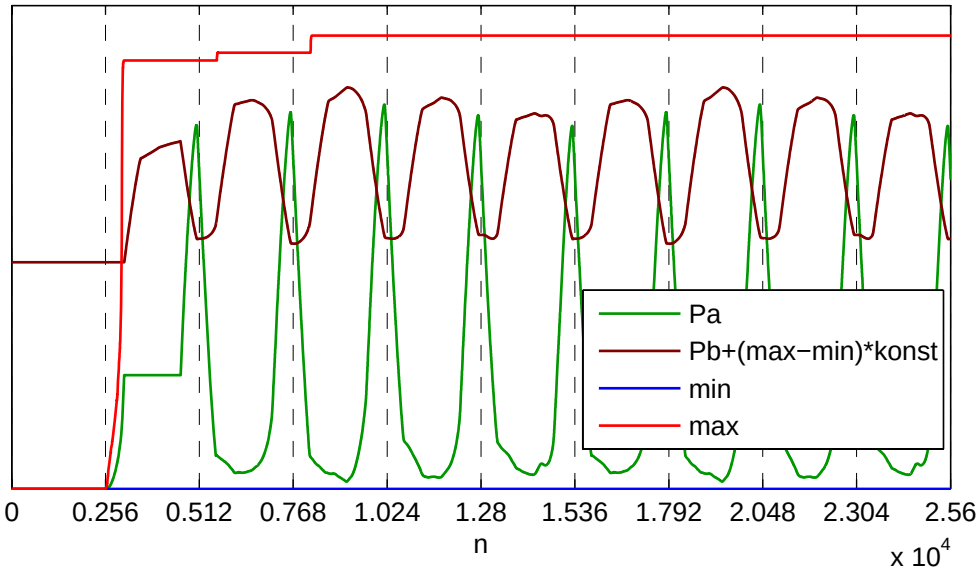
Obr. 14 Modul výstupu korelátoru pro 10 OFDM symbolů při SNR (signal-to-noise ratio) = 10 dB

Korelace se vyhodnocuje kontinuálně, cílem je najít lokální maximum modulu $y(n)$ odpovídající pozici začátku OFDM symbolu. Správná pozice FFT okna je pak o N_G vzorků dál. Aby bylo možné rozeznat vrchol korelace od náhodných špiček způsobených šumem, je

počítána dvojice průměrů s klouzavým oknem přes moduly $y(n)$. První průměr P_A se počítá přes N_G hodnot, maximum nastává při shodě s pozicí špičky korelace. Druhý průměr P_B se počítá z počtu N_S - N_G následujících hodnot (tj. je zde konstantní posun o N_G hodnot oproti P_A), takže v okamžiku maxima prvního průměru je druhý průměr minimální. Po stanovený počet OFDM symbolů se dále zjišťuje maximum *max* a minimum *min* modulů $y(n)$. Jakmile je maximum a minimum známo, začne se provádět porovnání aktuálních hodnot průměrů, a pokud

$$P_A > \left(P_B + (max - min) \cdot konst\right), \tag{5}$$

došlo k nalezení hledané špičky korelace (v realizovaném programu se uvažuje střed intervalu, kdy tato podmínka platí). Zavedení minima a maxima vytváří adaptivní práh, takže nezáleží na číselných hodnotách výsledků korelace, ale pouze na jejich časovém průběhu. Pomocnou konstantou *konst* je možné zvolit relativní práh citlivosti. Průběh zmíněných veličin pro 10 OFDM symbolů je zřejmý z Obr. 15.



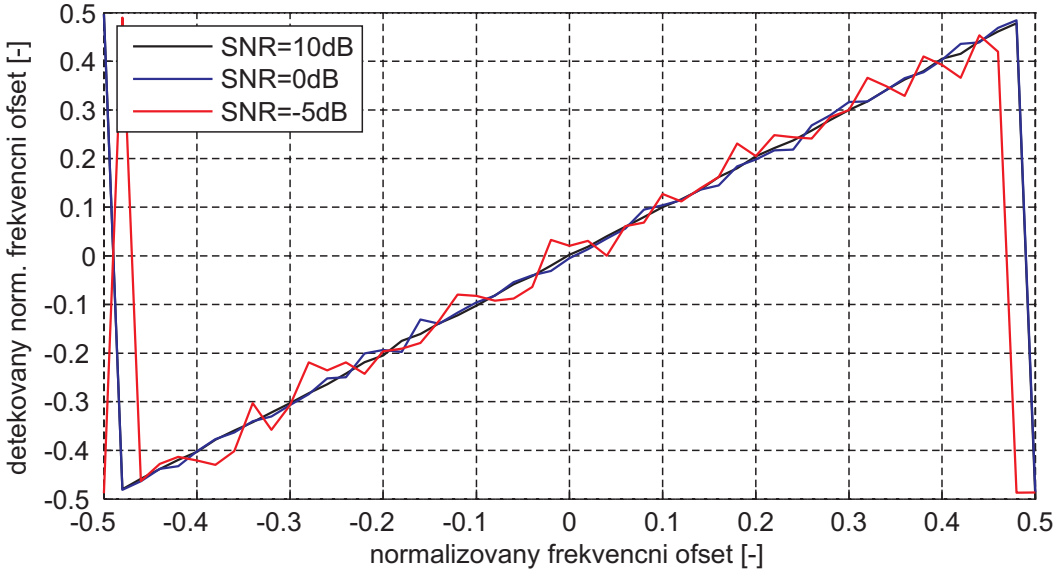
Obr. 15 Zpracování výsledků korelace (*konst* = 0,5)

4.2.2 Korekce zlomkového frekvenčního posunu před FFT

Z úhlu komplexního výstupu korelátoru v bodě odpovídající vrcholu korelace n_p lze podle [11] určit Maximum Likelihood odhad normalizovaného frekvenčního ofsetu Δf_{norm} ze vztahu

$$\Delta f_{norm} = -\frac{1}{2\pi} \arg\left(y(n_p)\right). \tag{6}$$

Vztah dává platné hodnoty v intervalu $-0{,}5 \le \Delta f_{norm} \le 0{,}5$, lze tedy určit frekvenční posun v mezích poloviny odstupu dvou sousedních nosných ($f_{norm} = f \cdot T_U$). Chybu danou přítomností šumu je možné korigovat po provedení FFT využitím pilotních signálů. Výsledek simulace tohoto bloku dekodéru je na Obr. 16.



Obr. 16 Odezva detektoru zlomkového frekvenčního ofsetu

4.2.3 Fázový derotátor

Tato jednotka je v blokovém schématu na Obr. 13 znázorněna jako „korekce frekvence“. Představuje sériově vřazený blok v cestě signálu před FFT, který na základě zjištěné Δf_{norm} provede kompenzaci frekvenčního posunu vynásobením (komplexního) signálu komplexní exponencielou, v jejímž argumentu se využívá opačná hodnota Δf_{norm}, tedy

$$s(n)_{kompenz} = s(n) \cdot \exp\left(j2\pi(-\Delta f_{norm}) \frac{n}{N} \right), \tag{7}$$

kde $s(n)$ jsou vzorky vstupního signálu a N = 2048 je šířka FFT. Pro implementaci do signálového procesoru je potřeba komplexní funkci rozdělit na reálnou a imaginární složku

$$\begin{aligned} \mathrm{Re}\{s(n)_{kompenz}\} &= \mathrm{Re}\{s(n)\} \cdot \cos\left(2\pi(-\Delta f_{norm}) \frac{n}{N} \right) - \mathrm{Im}\{s(n)\} \cdot \sin\left(2\pi(-\Delta f_{norm}) \frac{n}{N} \right) \\ \mathrm{Im}\{s(n)_{kompenz}\} &= \mathrm{Re}\{s(n)\} \cdot \sin\left(2\pi(-\Delta f_{norm}) \frac{n}{N} \right) + \mathrm{Im}\{s(n)\} \cdot \cos\left(2\pi(-\Delta f_{norm}) \frac{n}{N} \right) \end{aligned}. \tag{8}$$

4.2.4 Celočíselná korekce frekvenčního posunu po FFT

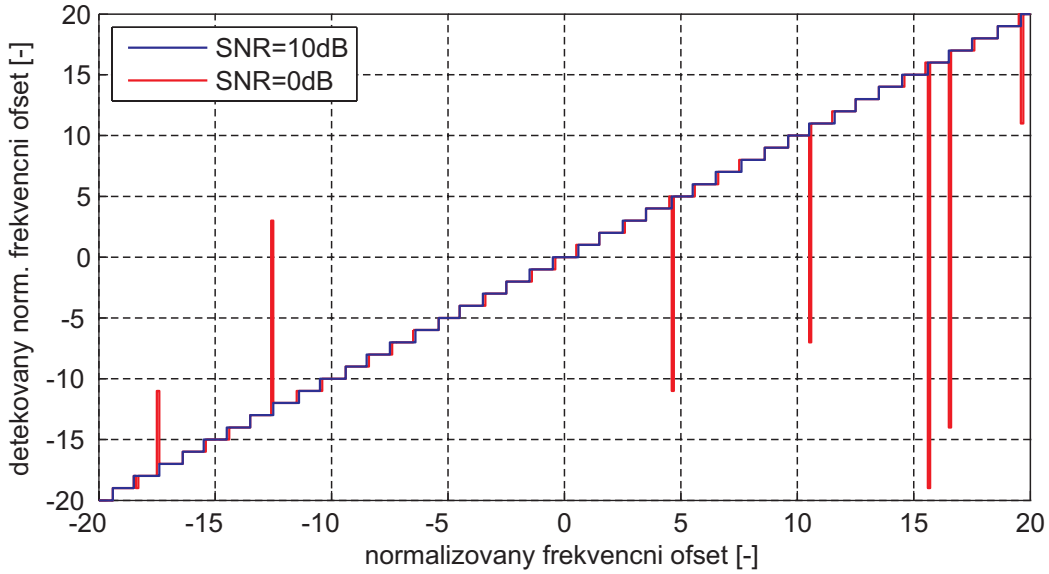
Po nalezení pozice začátku OFDM symbolu a kompenzaci zlomkového frekvenčního ofsetu je již možné provést Fourierovu transformaci. Na výstupu jsou k dispozici continual piloty, pomocí nichž je možné zjistit frekvenční posun o celočíselný násobek rozestupu nosných. CP se nachází na pevně daných nosných, jejich přemístění tedy přímo udává hledaný ofset. Hledá se maximum korelace dvou po sobě jdoucích symbolů počítané přes nominální pozice CP ± posun. Díky komplexnímu sdružení při výpočtu korelace se neuplatní zkreslení dané frekvenčně selektivním přenosovým kanálem ([11], [13]). Zjištěný ofset je

$$\varepsilon = \max\left(\arg\left|\phi(\kappa)\right|\right), \tag{9}$$

substituce $\phi(\kappa)$ odpovídá

$$\phi(\kappa) = \sum_{k \in CP} Y_{l,k+\kappa} Y^*_{l+1,k+\kappa} \,, \tag{10}$$

kde $Y_{l,k}$ je výstup FFT na k-té nosné pro l-tý OFDM symbol a κ je frekvenční ofset, který se testuje v mezích $-\kappa_{max} \leq \kappa \leq \kappa_{max}$. Při zvyšování mezí κ_{max} klesá použitelný počet pilotů. Jak však uvádí [11], i při použití pouze poloviny všech CP je pravděpodobnost nesprávného určení ε menší než 1% i při SNR < 4 dB. Simulovaná funkce je patrná z Obr. 17.



Obr. 17 Odezva detektoru celočíselného frekvenčního ofsetu (κ_{max} = 21, vstupní ofset krokován po 0,1)

4.2.5 Synchronizace scattered pilotů, jemná časová korekce

Scattered piloty se v přijatém OFDM symbolu mohou vyskytovat v jednom ze čtyř možných vzorů. Při použití metody [12] stačí vyhodnocení pouze dvou po sobě jdoucích OFDM symbolů k určení pozice SP. Výsledek lze navíc využít i k určení jemné časové korekce za předpokladu, že předtím proběhla kompenzace zlomkového frekvenčního ofsetu, jak je popsáno výše.

Vychází se z pravidelného uspořádání SP, které mají rozestup 12 nosných, v dalším symbolu jsou posunuty o 3 nosné. Pomocí korelací dvou po sobě jdoucích symbolů přes předpokládané pozice SP jsou získány čtyři různé výsledky $y_1 \div y_4$. Maximum z nich y_{max} pak určuje aktuální vzor rozmístění SP. Z úhlu tohoto výsledku pak lze zjistit hodnotu časového ofsetu pozice FFT okna [11]. Hledané výsledky jsou

$$y_1 = \sum_{k \in \{0,12,24,\ldots,1692\}} g_k g_{k+3} Y_{l,k} Y^*_{l+1,k+3} \,, \tag{11}$$

$$y_2 = \sum_{k \in \{3,15,27,\ldots,1695\}} g_k g_{k+3} Y_{l,k} Y^*_{l+1,k+3} \,, \tag{12}$$

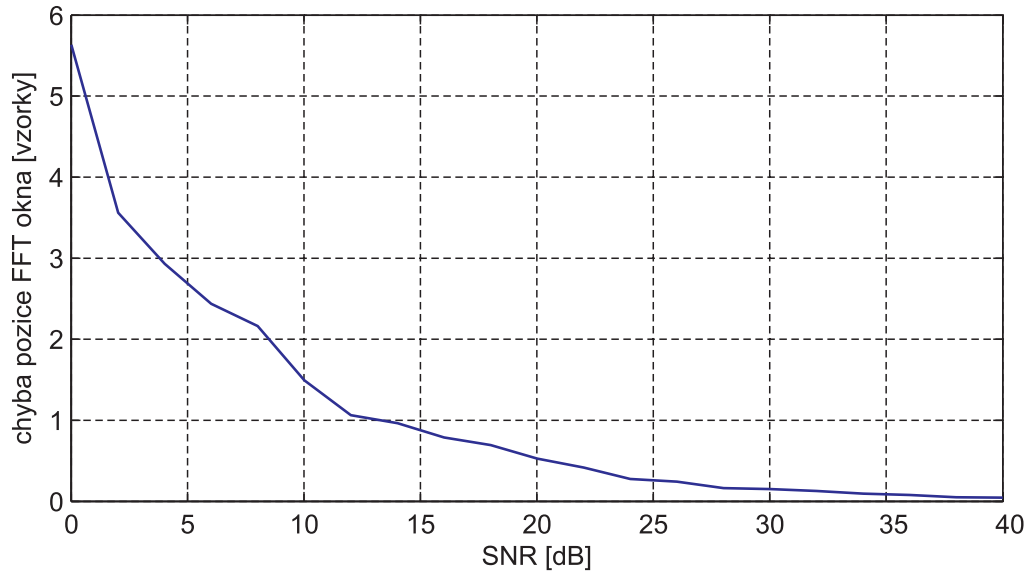
$$y_3 = \sum_{k \in \{6,18,30,\ldots,1698\}} g_k g_{k+3} Y_{l,k} Y^*_{l+1,k+3} \,, \tag{13}$$

$$y_4 = \sum_{k\in\{9,21,33,\ldots,1701\}} g_k g_{k+3} Y_{l,k} Y^*_{l+1,k+3}\,, \tag{14}$$

kde g_k je znaménko pilotu na k-té nosné (dáno PRBS generátorem). Hodnota časového ofsetu Δn je

$$\Delta n = -\frac{N}{2\pi \cdot D}\arg\left(y_{\max}\right), \tag{15}$$

kde $N = 2048$ je šířka FFT, faktor $D = 3$ udává rozestup SP (v norm. frekvenční ose), ze kterých je korelace y_{max} počítána. Vliv šumu na přesnost uvedené metody je patrný ze simulace na Obr. 18.



Obr. 18 Průměrná chyba určení Δ*n* v závislosti na SNR (průměr 50 běhů, neuvažována chyba zaokrouhlením na celé číslo)

4.3 Sledování vysílání (tracking)

Po dokončení zachycení vysílání přejde přijímač do módu sledování, ve kterém setrvává až do vypnutí, případně rozpoznání degradace přijímaného signálu pod stanovenou úroveň. Vyhodnocení ztráty signálu je možné provádět více způsoby, např. z výstupních dat (nárůst chybovosti detekovaný dekodérem FEC), nebo ověřováním přítomnosti pilotních signálů. V navrženém systému není tato funkce implementována.

4.3.1 Sledování zlomkového frekvenčního posunu

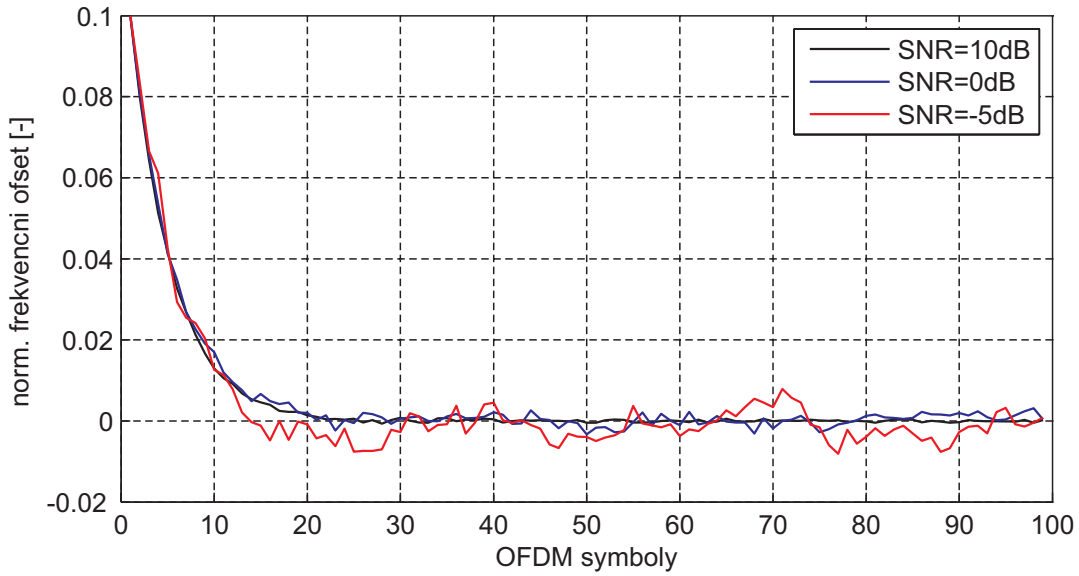
S použitím metody popsané v [13] je kontinuálně vyhodnocován fázový ofset přijímaných symbolů. Zjišťuje se globální fázový posun dvou po sobě následujících symbolů pomocí korelace continual pilotů. Hledaný posun je

$$\Delta f_{norm} = -\frac{1}{2\pi\left(1 + N_G / N\right)}\arg\left(\sum_{k\in CP} Y_{l,k} Y^*_{l+1,k}\right). \tag{16}$$

Výsledná hodnota je zkreslena vlivem šumu, proto jsou počítány průměrné hodnoty $\overline{\Delta f_{norm}}$, které se získávají exponenciální kumulací Δf_{norm}

$$\overline{\Delta f_{norm\,(i)}} = 0{,}9 \cdot \overline{\Delta f_{norm\,(i-1)}} + 0{,}1 \cdot \Delta f_{norm(i)}, \tag{17}$$

tedy s každým novým vzorkem Δf_{norm} dojde k mírné úpravě průměru, který tak postupně konverguje ke skutečné hodnotě frekvenčního ofsetu, který může být následně kompenzován fázovým derotátorem. Váhovací koeficient 0,9 byl zvolen experimentálně sledováním výsledků simulací tak, aby při očekávatelných hodnotách SNR došlo k dostatečnému ustálení v průběhu nízkých desítek symbolů i v případě tak velkého počátečního ofsetu, jak je vidět na Obr. 19. V realizovaném programu je z důvodu stability výsledná hodnota v každém běhu násobena tlumící konstantou 0,6. Vliv případného zbytkového ofsetu se vykompenzuje v bloku ekvalizace (viz dále).



Obr. 19 Časový průběh korekce počátečního norm. frekvenčního ofsetu 0,1

4.3.2 Sledování časového posunu

Vztah pro zjištění časového ofsetu Δn odpovídá rovnici (15) uvedené v kapitole 4.2.5, hodnota korelace y_{max} se však určí

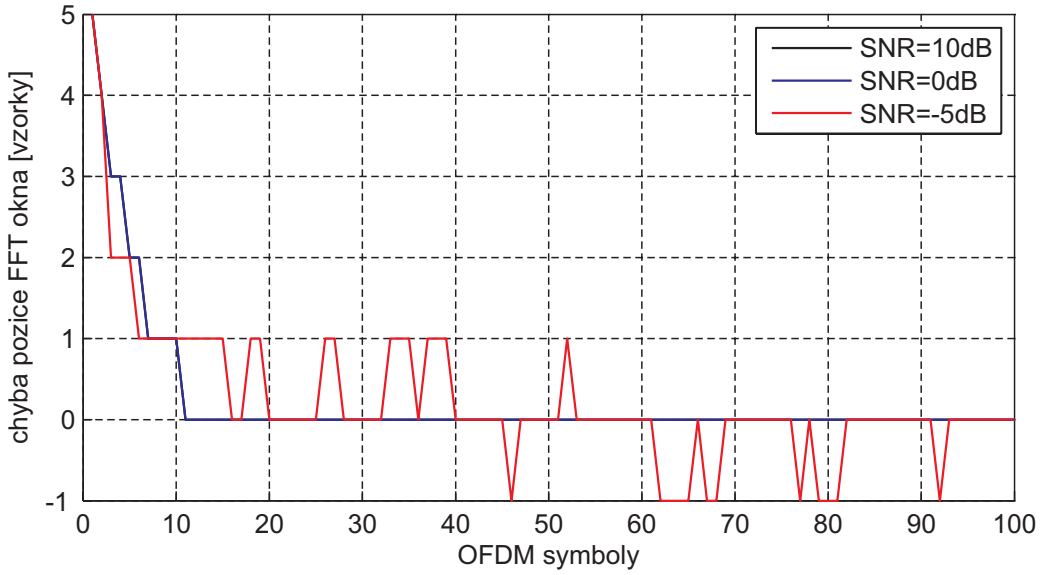
$$y_{\max} = \sum_{k \in SP} g_k g_{k+12} Y_{l,k} Y^*_{l,k+12}, \tag{18}$$

neboť Δn již nenabývá tak velkých hodnot a stačí větší rozestup scattered pilotů (pozice SP jsou již známy). Využívá se pouze jednoho OFDM symbolu a adekvátně se upraví parametr D v rovnici (15) na hodnotu 12. Analogicky výpočtu frekvenčního ofsetu se i zde provádí výpočet průměrné hodnoty exponenciální kumulací

$$\overline{\Delta n_{(i)}} = 0{,}7 \cdot \overline{\Delta n_{(i-1)}} + 0{,}3 \cdot \Delta n_{(i)}. \tag{19}$$

Váhovací koeficient byl opět zvolen na základě experimentů (v realizovaném programu se výsledek násobí 0,65 pro vyšší stabilitu). Jak již bylo zmíněno, časovou korekci je teoreticky možné provést s přesností na zlomky vzorkovacího intervalu. V navrženém systému se však

korekce provádí pouze v násobcích T, takže není nutné provádět interpolaci a decimaci vstupního signálu, čímž se sníží výpočetní a paměťové nároky. Vzniklou chybu max ± T / 2, která se projeví fázovým posunem nosných [13], vykompenzuje ekvalizér (viz dále). Tímto zaokrouhlením nemůže dojít k narušení ortogonality nosných, protože do stanoveného FFT okna nezasáhne sousední OFDM symbol [10], dojde však ke zhoršení odstupu signál/šum. Simulované průběhy jsou na Obr. 20.



Obr. 20 Časový průběh korekce pozice FFT okna z počátečního ofsetu 5 vzorků

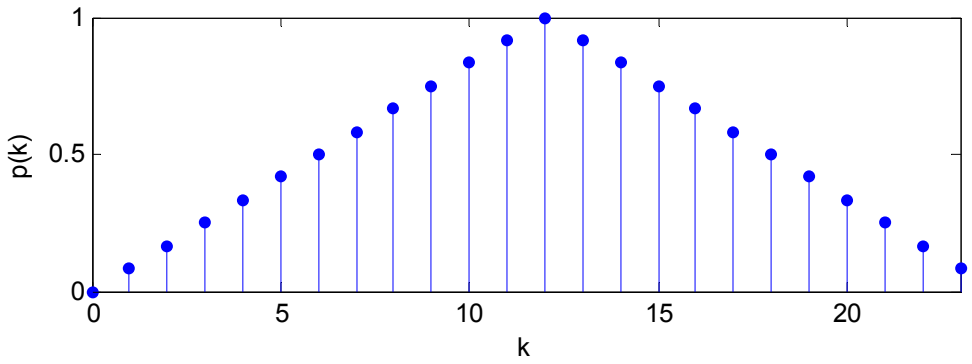
4.3.3 Odhad přenosové funkce kanálu, ekvalizace

Pro výpočet odhadu přenosové charakteristiky kanálu byl zvolen jednoduchý algoritmus uvedený v [14], který počítá odhad CTF (channel transfer function) pouze z pilotů v aktuálním OFDM symbolu a aplikuje ji na tento celý symbol. Výhodou této metody je rychlá odezva na časově proměnný kanál – je dosaženo nejkratší možné koherentní doby T_{ct}, omezené zespod délkou trvání jednoho OFDM symbolu T_S. Nevýhodou je horší rozlišení ve frekvenční oblasti dané vzdáleností scattered pilotů $12 \cdot (1 / T_U)$, kvůli čemuž je požadovaná koherentní šířka pásma kanálu $B_{ch} > 12 / T_U$. Navíc použitá metoda neošetřuje přítomnost šumu, což je daň za minimální výpočetní nároky. Pro snížení vlivu AWGN by bylo možné hodnoty pilotů filtrovat dolní propustí, existují i další sofistikované techniky pro optimalizaci odhadu CTF (interpolace v časové oblasti [7], 2-D filtrace, Wienerův filtr, ...), jak však uvádí [14], jejich přínos je vykoupen enormními nároky na výpočetní výkon.

CTF je známa na frekvencích, na kterých jsou umístěny piloty, jejichž demodulovaná hodnota udává přímo vzorky komplexní přenosové funkce kanálu $H_l(k)$ na nosné k pro symbol l. Lineární interpolací se pak určí hodnota přenosu $H_l(k)$ pro všechny ostatní nosné. Pro okrajové části spektra, kde nelze interpolaci uplatnit, jsou použity hodnoty CTF z pilotů nejbližších okrajům. Výpočet interpolované CTF $H_l(k)$ odpovídá konvoluci $H'_l(k)$ tvořené demodulovanými piloty proloženými nulami s trojúhelníkovou impulsní odezvou $p(k)$, tj.

$$H_l(k) = \sum_{i=0}^{\min(k+12,23)} H'_l(k+12-i)\,p(i), \qquad (20)$$

kde $SP_{min} \leq k \leq SP_{max}$ (tj. od první pozice scattered pilotu po poslední pozici dle aktuálního vzoru rozmístění SP v daném symbolu), hodnoty $p(k)$ jsou znázorněny na Obr. 21.



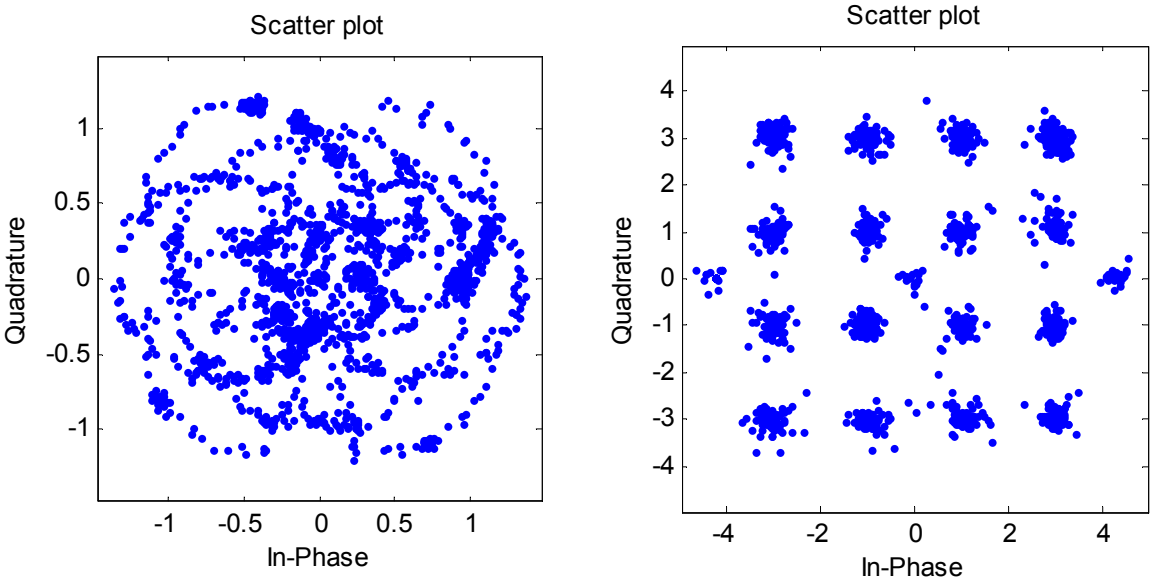
Obr. 21 Hodnoty impulsní odezvy p(k) interpolačního filtru

Po výpočtu CTF jsou metodou „Zero forcing“ upraveny (komplexní) hodnoty nosných Y_k na nové ekvalizované hodnoty Y_k' podle vztahu

$$Y_k' = Y_k \cdot \frac{1}{H_l(k)}. \tag{21}$$

Je-li v kanále přítomen šum, dojde pro velmi malé $H_l(k)$ k velkému zesílení šumu na dané nosné. V přijímačích vybavených Viterbiho dekodérem se využívá velikosti $H_l(k)$ pro stanovení „důvěryhodnosti“ informace obsažené na dané nosné, čímž lze zlepšit BER (bit error ratio) na výstupu přijímače [14].

Na Obr. 22 je patrný vliv frekvenčního zkreslení (OFDM signál prošel dolní propustí) na uspořádání konstelačního diagramu a jeho korekce ekvalizérem.

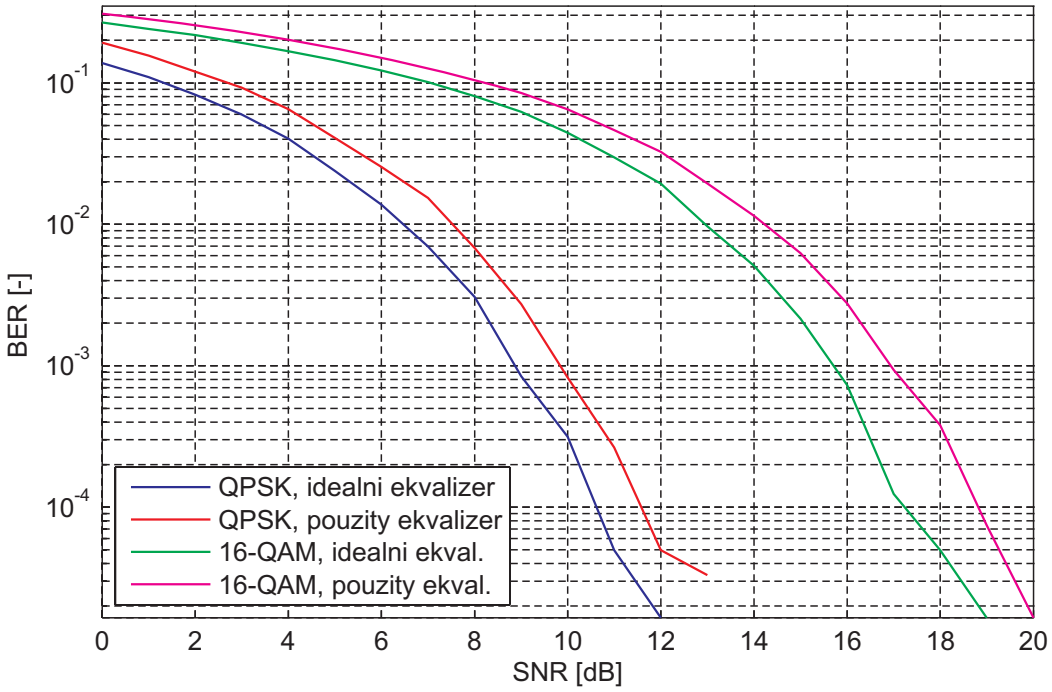


a) **b)**

Obr. 22 Konstelační diagram 16-QAM a) před ekvalizérema b) po ekvalizaci

Ekvalizér vykompenzoval chybu fáze nosných, zároveň došlo k obnovení amplitud, takže konstelační body jsou umístěny na korektních pozicích (±1, ±3 po pronásobení normovací konstantou $\sqrt{2}$ pro QPSK resp. $\sqrt{10}$ pro 16-QAM). K simulovanému signálu byl navíc přidán AWGN šum (SNR = 25 dB), který se ve výsledném konstelačním diagramu projevuje „rozmazáním“ konstelačních bodů. Body umístěné na reálné ose odpovídají pilotním signálům (CP+SP v levé a pravé krajní pozici, TPS v nule).

Vzhledem k tomu, že použitý princip ekvalizace nebere v úvahu přítomnost šumu, budou vypočtené hodnoty CTF zkresleny náhodným příspěvkem. To v důsledku vede k určitému zhoršení chybovosti na výstupu dekodéru oproti ideálnímu ekvalizéru, jehož odhad CTF by přesně kopíroval skutečnou CTF, bez ohledu na šum. Při simulaci tohoto efektu byl použit AWGN kanál bez změny frekvenční charakteristiky signálu, takže ideální ekvalizér má CTF = 1 pro všechny nosné. Na Obr. 23 je patrné, jak se použití reálného ekvalizéru projeví na chybovosti výstupních dat (po demodulaci nosných – viz dále).



Obr. 23 Zhoršení BER vlivem použitého ekvalizéru

4.3.4 Demodulace nosných

Demodulace nosných je inverzní proces k modulaci, popsané v kapitole 3.1. Na základě obsahu nosné Y_k' se hledá v konstelačním diagramu bod s minimální Euklidovskou vzdáleností a jemu odpovídající dibit (4-bit) je výstupem demodulace. Takto je minimalizována pravděpodobnost chyby ve smyslu Maximum Likelihood kritéria [15].

Hledání minimální vzdálenosti je v signálovém procesoru realizováno porovnáním reálné a imaginární složky Y_k' s hranicemi umístěnými v polovině vzdáleností mezi nominálními pozicemi v konstelačním diagramu.

4.4 Další nežádoucí jevy

Kromě časového a frekvenčního ofsetu, lineárního zkreslení a šumu existují mnohé další negativní vlivy, které mohou narušit přenos OFDM signálu. V navrženém dekodéru však nejsou speciálně ošetřovány, neboť jejich vliv je dostatečně omezen existujícími bloky, případně je tento vliv neodstranitelný s rozumnými výpočetními nároky.

K nejvýznamnějším nežádoucím jevům patří náhodná změna fáze oscilátorů (phase noise), která může značně ovlivnit kvalitu přenosu signálu [9], [11]. Neexistuje však schůdná cesta jak jej výpočetně odstranit. Záleží tedy na jakosti příslušných (analogových) komponent.

Dalším typickým jevem je například ofset vzorkovací frekvence, který se projevuje pomalým časovým „klouzáním“ přijímaného signálu [10]. Pokud přesáhne hranici citlivosti navrženého systému, dojde k jeho jednorázovému vykompenzování blokem jemné časové korekce. Toto se pak opakuje ve víceméně pravidelných intervalech.

Značným problémem bývá velký rozdíl mezi špičkovým a středním výkonem a s tím spojená otázka nelineárního zkreslení. Tento jev se v realizovaném systému díky přenosu v základním pásmu (bez zesilovačů) neprojeví.

Někdy může připadat v úvahu rychle se měnící kanál, kdy přestává platit předpoklad, že po dobu jednoho OFDM symbolu je CTF statická [11]. Důsledkem je zvýšení ICI a tím zhoršení BER.

Při skutečném vysílání DVB-T se projevují silné zpožděné příjmy dané použitím SFN (single frequency network). Tyto sítě se již projektují s ohledem na fakt, že pokud zpoždění echa nepřesáhne T_G, nedojde ke ztrátě ortogonality a lze provést korekci kanálovým ekvalizérem. Tekto ekvalizér pak musí být navržen s ohledem na odpovídající koherentní šířku pásma B_{ch} [14].

5 Struktura DSP TMS320C6711, možnosti využití výkonu

5.1 Vývojová deska C6711 DSK

Navržený kodér a dekodér OFDM byl realizován ve vývojovém prostředí Code Composer Studio v. 3.1 s využitím dvojice vývojových desek Texas Instruments C6711 DSK, jejichž jádrem je DSP (digital signal processor) TMS320C6711 [16] taktovaný na 150 MHz. Výrobce udává při této frekvenci výpočetní výkon 900 MFLOP (million floating-point operations per second), skutečně dosažitelná hodnota je však podstatně nižší. Vývojové desky dále obsahují 16 MB paměti SDRAM (synchronous dynamic random access memory), která komunikuje prostřednictvím 100 MHz sběrnice EMIF (external memory interface). Pro zavedení programu je k dispozici 128 kB programovatelné paměti flash ROM. Spojení s počítačem je realizováno paralelním portem s emulátorem protokolu JTAG a rozhranním HPI (host port interface). Pro převod signálu mezi digitální a analogovou podobou byla využita rozšiřující deska DSP STAR AD/DA od firmy ND-Tech Co., Ltd.

Tento procesor s danou externí pamětí neposkytuje dostatečný výpočetní výkon, aby rychlost zpracování realizovaného kodéru resp. dekodéru OFDM vyhovovala požadavkům standardu [1].

5.2 Výkonné jádro procesoru

TMS320C6711 je svým výkonem a vlastnostmi předurčen spíše pro experimentální a speciální aplikace, kde postačí jeho rychlost a využije se schopnosti pracovat s plovoucí desetinnou čárkou. V dnešní době je již tento typ zastaralý, existují novější řady. Jedná se o tzv. floating-point procesor založený na architektuře VLIW (very long instruction word) uzpůsobený k práci s 32-bitovým datovým slovem podle standardu IEEE-754. Jádro procesoru tvoří 8 paralelně pracujících jednotek, každá jednotka obsluhuje určitou podmnožinu instrukčního souboru. Podstatou VLIW u tohoto procesoru je sdružení 8 instrukcí pro všechny funkční jednotky do jediného tzv. instrukčního paketu.

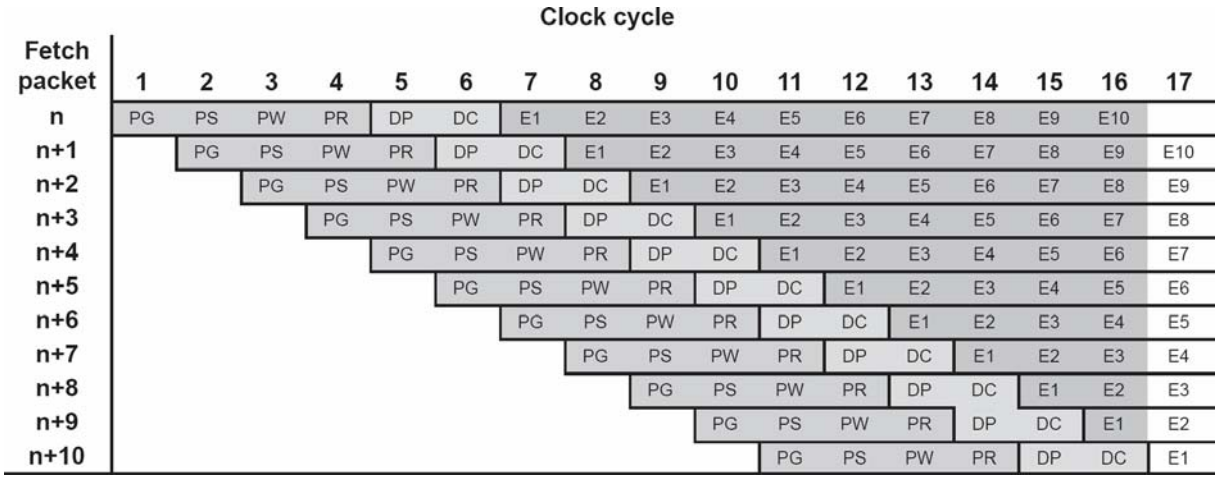
V závislosti na variantě procesoru může být maximální hodinová frekvence jádra 100 až 250 MHz. Každá instrukce je vykonána za 7 až 16 hodinových cyklů (podle typu instrukce). Jednotlivé fáze výkonu jsou:

- PG (program address generate)
- PS (program address send)
- PW (program access ready wait)
- PR (program fetch packet receive)
- DP (instruction dispatch)
- DC (instruction decode)
- E1-E10 (execute)

Díky tzv. instrukční pipeline může být v každém hodinovém cyklu spuštěno zpracování další instrukce (viz Obr. 24). Pokud tedy nedojde k větvení programu, je teoreticky možné každý hodinový cyklus dokončit jednu instrukci. Všechny instrukce v daném instrukčním paketu prochází pipeline společně, díky čemuž může počet vykonaných

instrukcí za vteřinu přesáhnout hodinovou frekvenci jádra. Omezený instrukční soubor jednotlivých funkčních jednotek však obvykle neumožňuje výraznější stupeň paralelismu.

Někdy v programu vstupní data jedné instrukce závisí na výsledcích předchozích instrukcí. Pokud není možné využít zbylé funkční jednotky jinými instrukcemi, musí se vložit prázdné instrukce NOP do doby, než je k dispozici potřebný výsledek. Toto do značné míry smazává výhodu pipeline, neboť se nejen ztrácí možnost paralelního výkonu ve funkčních jednotkách, ale navíc vykonání dané instrukce zabere velký počet hodinových cyklů. Kromě toho má každá instrukce definovanou tzv. latenci funkčních jednotek. Tím je řečeno, jak dlouho nesmí jiná instrukce dané funkční jednotky využít, a to i v případě, že výkon instrukce byl předčasně ukončen jejím podmínečným zpracováním. Existují i další omezení snižující dosažitelný výpočetní výkon a se všemi musí být předem počítáno při psaní programu. Psaní efektivního kódu v assembleru je pro člověka extrémně náročné. Je proto vhodné využít vyšší programovací jazyk (obvykle C) a na kritické pasáže aplikovat knihovnové funkce (bývají psány profesionály a ručně optimalizované). Velkou roli hrají též optimalizační schopnosti překladače.



Obr. 24 Řazení instrukcí do pipeline (převzato z [17])

Celkově lze říci, že k nejlepšímu využití možností procesoru dochází při cyklickém zpracování dat (např. při výpočtu odezvy filtru). Díky zachování pipeline při skoku na začátek cyklu je možné v *i*-té iteraci spustit výkon instrukcí pro *i+1* iteraci, popř. i několik následujících. Cyklus se rozdělí na tzv. prolog (spuštění cyklu), opakují se část a epilog (doběhnutí rozpracovaných instrukcí po skončení). Toto umožňuje podstatně efektivnější využití funkčních jednotek a pipeline. Překladač jazyka C ve vývojovém nástroji Code Composer Studio umí takto optimalizovat smyčky typu *for* a *while*. V těle smyčky ovšem nesmí být volány rozsáhlejší podprogramy; kratší funkce překladač automaticky vloží do těla cyklu (v C zdroji není vyžadováno uvedení klíčového slova inline). Pomocí direktivy *#pragma MUST_ITERATE* je možné dát překladači dodatečnou informaci o garantovaném minimálním a maximálním počtu opakování, což napomáhá optimální kompilaci. Použití této direktivy umožňuje překladači i další transformace smyček zefektivňující jejich běh i velikost programu (unrolling, software pipelining, nested loop transformations).

Společným znakem všech floating-point procesorů je složitá a rozměrná výpočetní jednotka. Ve srovnání s fixed-point procesory je proto typická nižší hodinová frekvence, menší velikost pamětí na čipu procesoru a vyšší cena. Výhodou je naopak velký dynamický rozsah použitého datového typu a snazší vývoj programů. Jazyk C navíc není přímo slučitelný

s fixed-point procesory, protože neobsahuje zlomkový datový typ. Proto se často v rámci vývoje vytvoří program využívající plovoucí řádovou čárku, který se pak přenese a zoptimalizuje pro procesory s pevnou řádovou čárkou. Tyto procesory pak ve stejné cenové hladině běžně dosahují řádově vyššího výkonu.

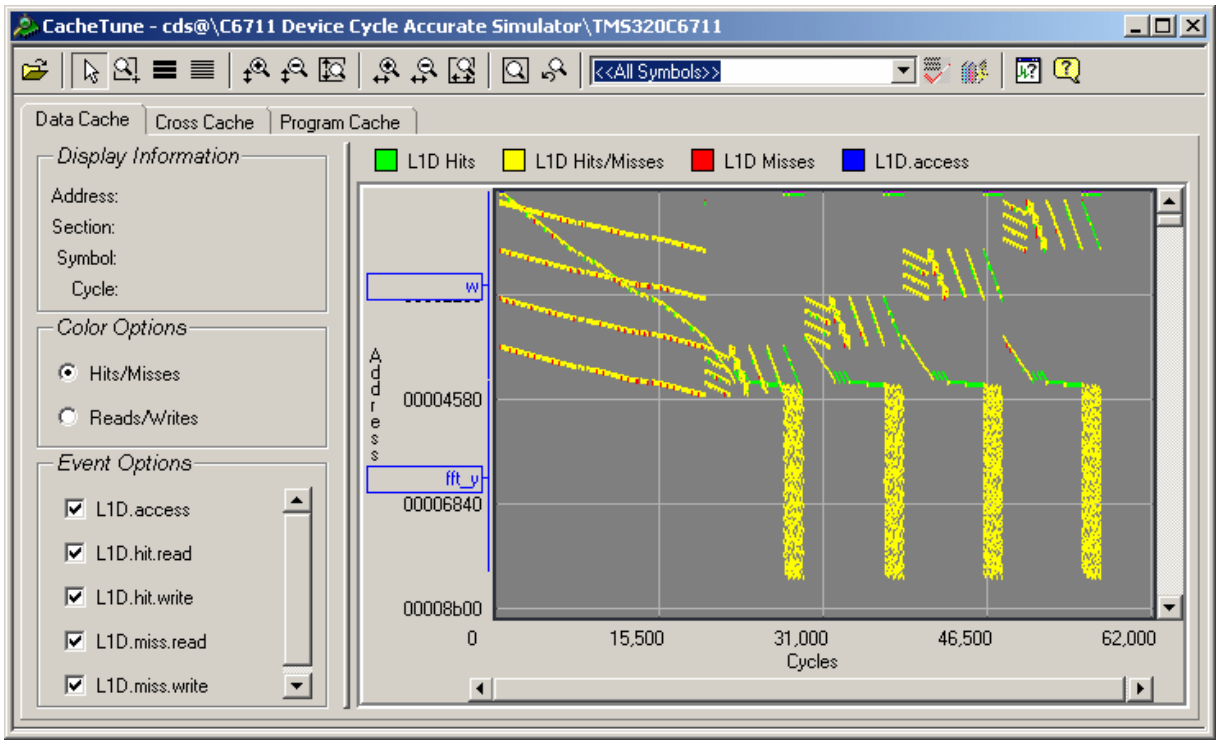
5.3 Paměť, CACHE

Slabinou signálových procesorů je nízká datová propustnost externích pamětí. Proto je snaha co nejvíce využívat vnitřních pamětí. U procesorů řady 6000 existuje dvoustupňová struktura vyrovnávacích pamětí CACHE [18], které snižují prostoje procesoru při přístupu k datům a mají tedy zásadní vliv na celkový výpočetní výkon. Pokud je nutné zpracovávat příliš objemná data (případ řešeného kodéru a dekodéru), velikost vyrovnávacích pamětí přestává stačit a dochází k nevyhnutelnému poklesu rychlosti zpracování danému přístupovou dobou a propustností externí paměti komunikující po 100 MHz sběrnici. Pro ilustraci lze uvést, že použitý algoritmus FFT je v simulátoru s plochým paměťovým modelem (ideální paměť) cca 4x rychlejší než při reálné paměťové struktuře.

Jádro C6711 provádí všechny výpočtové instrukce se souborem 32 registrů, jejichž obsah je vyměňován s vnitřní datovou pamětí L1D CACHE (programová L1P CACHE je oddělená). Tato paměť pojme 4 kB a představuje nejrychlejší úložiště pro data, která jsou načítána automaticky po blocích na principu asociativity (2-way set-associative). L1D je rozdělena do podskupin (sets), přičemž nejmenší jednotku tvoří skupina 32 bajtů, tzv. line. Při každé výměně dat mezi L1D a L2 CACHE je přenesena jedna line, v programu je proto vhodné všechny proměnné typu pole (array) zarovnat v paměti na 32-bajtové násobky. Toto lze zajistit direktivou *#pragma DATA_ALIGN*. Vzhledem k prostorové asociativitě CACHE je dále vhodné zpracovávat prvky polí sekvenčně v souladu s jejich rostoucí adresou.

Paměť druhé úrovně L2 CACHE má velikost 64 kB a velikost line je 128 bajtů. Tato paměť je společná pro data i program. Lze ji rozdělit na 1–4 bloky a část vyhradit jako statickou RAM v adresním prostoru od 0x00000000. Do takto vzniklého prostoru je výhodné umístit nejfrekventovaněji používané proměnné, neboť je zde zaručen nejrychlejší přístup. V programu se pro ně použije direktiva *#pragma DATA_SECTION*, přičemž musí být příslušný datový prostor specifikován v kompilačním souboru (*.cmd). Při použití operačního systému musí být též provedeno příslušné nastavení v jeho konfiguraci. Pokud jsou velká pole umístěna v externí paměti, je vhodné direktivou *#pragma DATA_ALIGN* provést zarovnání na 128 bajtů. Stejně jako u L1D zde existuje výhoda v sekvenčním zpracování polí.

V prostředí Code Composer Studio existuje nástroj CacheTune pro ladění programu z hlediska efektivního využití CACHE na základě simulovaného běhu procesoru, viz Obr. 25. Umožňuje zkoumat časové sekvence čtení/zápisů do L1/L2 CACHE pro všechny proměnné a vyhodnocovat konflikty dat. Existuje více různých typů konfliktů [18], některé jsou odstranitelné změnou programu, jiné nikoli (např. kvůli velkým objemům zpracovávaných dat). Konflikt způsobuje vzájemné vytlačování (eviction) různých dat z paměti, což při jejich opětovném načtení způsobuje časové ztráty. V realizovaném kodéru a dekodéru byla využita knihovnová funkce pro výpočet FFT, která je optimalizovaná z hlediska využití CACHE.



Obr. 25 Nástroj CacheTune

5.4 Periferie, řadič EDMA

Procesor TMS320C6711 může komunikovat s okolím dvěma prostředky: sériovým kanálem (multichannel buffered serial port) nebo přes rozhranní EMIF. Použitá deska s AD/DA převodníky využívá rozhranní EMIF. Vstup resp. výstup převodníků tak představuje paměťově mapovaný prostor. Pro generování vzorkovací frekvence je u kodéru i dekodéru využit jeden ze dvou univerzálních časovačů, které jsou v tomto DSP k dispozici.

V technice DSP se často signály zpracovávají tak, že se vyvolá přerušení, načte se vstupní vzorek, provede se jeho zpracování a vygeneruje se příslušný výstup. S dalším vzorkem následuje další přerušení atd. Algoritmus musí stihnout veškeré zpracování v intervalu mezi dvěma vzorky, navíc se do potřebného času přičítá režie na obsloužení přerušení. Tento přístup nelze při výpočtu kodéru/dekodéru OFDM aplikovat, neboť jeden OFDM symbol se skládá z velkého množství signálových vzorků a algoritmus výpočtu pracuje se všemi těmito vzorky naráz. Kodér by ovšem mohl provádět výpočet daného symbolu a v přerušeních odesílat na výstup vzorky z předchozího symbolu, které byly uchovány v paměti. Rychle se opakující přerušení by však měla drastický dopad na výpočetní výkon procesoru. V realizovaném systému jeden OFDM symbol obsahuje 2560 signálových vzorků, během výpočtu by tedy 2560-krát došlo ke skartaci pipeline při přechodu do obslužného podprogramu a stejně i při návratu, byla by narušována CACHE, vznikla by velká režie obsluhy přerušení a procesor by musel věnovat svůj čas vlastnímu přesunu dat. TMS320C6711 nabízí vysoce efektivní alternativu a tou je řadič přímého přístupu do paměti.

Řadič EDMA (enhanced direct-memory-access) [19] umožňuje přesuny bloků dat mezi různými místy v paměti s širokou variabilitou funkce a parametrů. Zásadní výhodou je, že veškeré přesuny se dějí na pozadí, tj. není jimi zatěžováno jádro procesoru. Při přesunu vzorků z A/D převodníku je tak např. možné naplnit předem určené pole v paměti, které posléze procesor přímo zpracuje jako jakoukoli jinou proměnnou. EDMA přenos se skládá

z prvků (elements), které se sdružují do rámců (frames); rámce pak tvoří blok. Celý systém pracuje na základě spouštěcích událostí, tzv. events. V realizovaném kodéru je touto událostí každé dokončení periody časovače, v dekodéru je jí signál na pinu procesoru EXT_INT7 vyvolaný A/D převodníkem. Tato událost může vyvolat přenesení jednoho prvku, všech prvků v jednom rámci, nebo celého bloku. EDMA přenos má předem určené parametry, mezi ty hlavní patří:

- zdrojová a cílová adresa,
- způsob změny adres po přenesení prvku a rámce,
- šířka přesouvaných dat (8, 16 nebo 32 bitů),
- způsob ukončení přenosu.

Řadič provádí aktualizaci všech parametrů v průběhu přenosu. Po vyčerpání celého bloku se může přenos zastavit, nebo se aktualizují parametry z předem připravené předvolby a při příští události se tak spustí nový přenos. V realizovaném kodéru a dekodéru je použita automatická změna parametrů na konci přenosu pro vytvoření systému typu double-buffering, kdy v průběhu zpracování jednoho OFDM symbolu je na pozadí přenášen předchozí resp. následující symbol.

6 Popis programu realizovaného kodéru

6.1 Operační systém DSP BIOS

Kodér i dekodér využívají preemptivní multi-threadingový operační systém DSP BIOS [20] s podporou hardwarové abstrakce. Jedná se o specializovaný systém zaměřený na běh procesů v reálném čase, jejich synchronizaci, komunikaci s hostitelským počítačem a též logickou analýzu za běhu programu. Jádro BIOSu je modulární a škálovatelné, takže daná aplikace použije jen rozsah, který je zapotřebí. Všechny rutiny jádra jsou ručně optimalizované a v typické aplikaci spotřebují řádově procento výkonu procesoru. Použití BIOSu umožňuje přímočařejší postup při vývoji programů, usnadňuje psaní, ladění i rozšiřování stávajících aplikací a zajišťuje jejich přenositelnost mezi procesory řady TMS320C5000 a TMS320C6000 (v závislosti na stupni abstrakce). Na rozdíl od univerzálních operačních systémů je v BIOSu potlačeno na minimum testování chybových stavů, aby se předešlo plýtvání procesorovým časem. U všech API (application programming interface) funkcí jsou přesně specifikovány možnosti jejich volání a zodpovědnost za správné použití je na programátorovi.

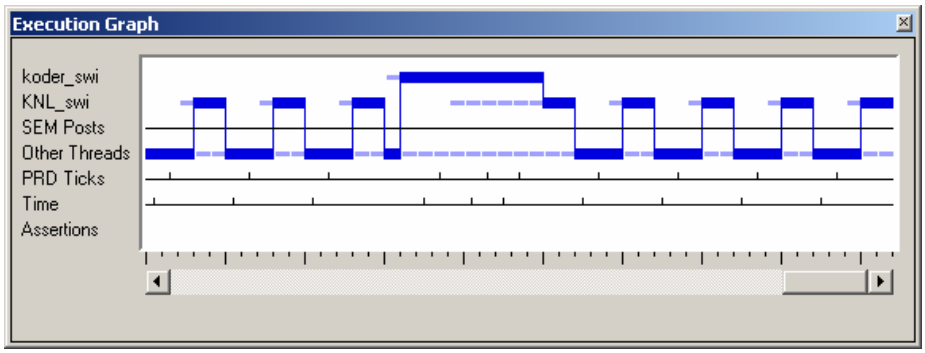
BIOS podporuje více specifických typů úloh (threads), mezi něž patří:

- HWI (hardware interrupt), tj. obsluha hardwarových přerušení,
- SWI (software interrupt); tyto procesy zpracovávají časově kritické úlohy,
- TSK (task) – proces, který má široké spektrum použití, může přenechávat procesorový čas a synchronizovat se s dalšími procesy,
- IDL (idle loop) – proces běžící v pozadí, zpracovává úlohy s nejnižší prioritou, mj. komunikaci s hostitelským počítačem.

V procesoru může běžet větší množství různých procesů zpracovávajících různá data s různou rychlostí. Toto umožňuje vedle použitých typů procesů také jejich nastavitelná priorita.

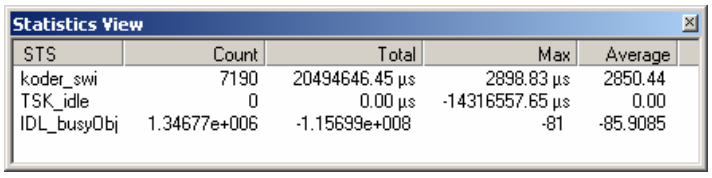
Součástí BIOSu je sada nástrojů pro analýzu procesů v reálném čase za běhu programu ve spolupráci s prostředím Code Composer Studio. Patří mezi ně:

a) Execution graph – analýza sekvencí spouštění procesů, viz Obr. 26.



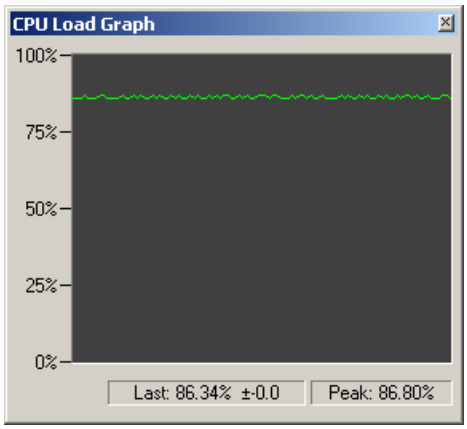
Obr. 26 Execution graph

b) Statistics view – statistika časových parametrů procesů, viz Obr. 27.



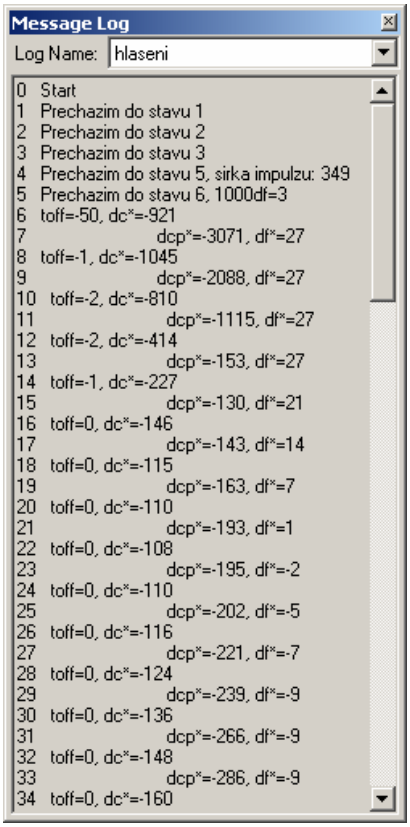
Obr. 27 Statistics view

c) CPU load graph – graf vytížení procesoru, viz Obr. 28.



Obr. 28 CPU load graph

d) Message log – nástroj pro rychlý výpis informací z běžícího programu do připojeného počítače, viz Obr. 29.



Obr. 29 Message log

6.2 Použitá konfigurace, proměnné a uspořádání paměti

Pomocí konfiguračního nástroje BIOSu byla pro kodér provedena následující nastavení:

- interní SRAM byla rozdělena na poloviny, 32 kB tvoří dvoucestnou L2 CACHE, dalších 32 kB slouží jako adresovatelná datová paměť,
- dva ze tří datových vektorů, které jsou použity při výpočtu FFT, byly alokovány v interní SRAM, ostatní data i program v externí SDRAM,
- jako systémový časovač byl nastaven TIMER0, druhý časovač řídí D/A převod a EDMA,
- vytvořen proces EDMA_HWI typu hardwarové přerušení, který je spouštěn po dokončení EDMA přenosu předchozího OFDM symbolu,
- vytvořen proces KODER_SWI typu softwarové přerušení pro generování nového OFDM symbolu,
- pro shromažďování statistických dat o běhu programu byl vyhrazen systémový buffer o velikosti 512 bajtů,
- byla nastavena velikost systémového zásobníku (stack) na 1024 bajtů,
- byla vypnuta podpora dynamicky alokovaných struktur pro úsporu systémových prostředků, tj. všechny objekty jsou deklarovány staticky.

Program používá několik globálně deklarovaných datových polí, které slouží jako vektory komplexních čísel. Výpočetně nejintenzivnější operací je inverzní FFT, proto byly dva ze tří použitých vektorů alokovány do interní paměti, ke které je nejrychlejší přístup. Vzhledem k omezené velikosti této paměti bylo nutné volit kompromis, protože zabrání více než poloviny SRAM by výrazně zpomalilo všechny ostatní výpočty. Všechny vektory byly zarovnány na délku CACHE line, aby byly minimalizovány ztráty času při jejich načítání do CACHE. Část globálních deklarací je uvedena v následujícím výpisu.

```
//promenne:
#pragma DATA_ALIGN(vstup,CACHE_L2_LINESIZE);
unsigned char vstup[kolikbajtu];    //vektor vstupnich bajtu koderu
#pragma DATA_SECTION(w, ".vnitrni");
#pragma DATA_ALIGN(w,CACHE_L1D_LINESIZE);
float w[sirkafft*2];                //twiddle factors pro FFT
#pragma DATA_SECTION(fft_y, ".vnitrni");
#pragma DATA_ALIGN(fft_y, CACHE_L1D_LINESIZE);
float fft_y[sirkafft*2];            //vytupni vektor z IFFT
#pragma DATA_ALIGN(fft_x, CACHE_L2_LINESIZE);
float fft_x[sirkafft*2+16];         //vstupni vektor do IFFT + povinny padding
unsigned char poradiSPkod=0;        //0..3 pro scattered piloty
#pragma DATA_ALIGN(okno1, CACHE_L2_LINESIZE);
#pragma DATA_ALIGN(okno2, CACHE_L2_LINESIZE);
float okno1[nramp],okno2[nramp];    //okenkovaci vektory
#pragma DATA_ALIGN(outbuf1, CACHE_L2_LINESIZE); //zarovnano pro L2 CACHE writeback
unsigned short outbuf1[CACHE_ROUND_TO_LINESIZE(L2,((sirkafft+ng)*2),sizeof(unsigned
short))];                           //dvojity buffer pro DA
#pragma DATA_ALIGN(outbuf2, CACHE_L2_LINESIZE); // zarovnano pro L2 CACHE writeback
unsigned short outbuf2[CACHE_ROUND_TO_LINESIZE(L2,((sirkafft+ng)*2),sizeof(unsigned
short))];                           //dvojity buffer pro DA
```

V deklaracích je využito maker z knihovny Chip Support Library [22], jejíž API tvoří podstatnou část veškerých použitých operací s hardwarem (viz dále). V nastavení překladače musí být vložena konstanta označující použitý procesor, neboť API jsou univerzální pro všechny podporované DSP.

Pro vyhrazení části vnitřní paměti jako datové SRAM je nutné kromě vytvoření sekce IRAM v konfiguraci BIOSu použít ještě konfigurační soubor pro překladač, jehož výpis následuje.

```
SECTIONS { .vnitrni: {} > IRAM
}
```

Pro finální program byl ve vlastnostech překladače nastaven nejvyšší stupeň optimalizace specifikovaný parametry *–o3 –pm*.

6.3 Inicializace programu, D/A převodníky, EDMA

Prvotní inicializaci provede na pozadí jádro operačního systému, které pak předá řízení funkci *main()*, ve které následují inicializace dané aplikace. Očekává se, že po provedení všech potřebných rutin je funkce *main()* opuštěna, čímž se předá řízení zpět jádru operačního systému. Pak jsou již v souladu s konfigurací automaticky povolena přerušení a spouštěny jednotlivé procesy. Výpis funkce *main()* následuje.

```
void main()
{
  //inicializace CSL library
  CSL_init();
  //vygenerovani simulovanych dat:
  genvstup();
  //vygenerovani pomocnych dat pro FFT:
  gen_twiddle();
  //vygenerovani okenkovacich vektoru:
  ok_vektory();
  //inicializace DA prevodniku a EDMA:
  da_init(f_prenos,outbuf1,outbuf2,(sirkafft+ng));
}
```

Podprogram *genvstup()* vygeneruje vektor bajtů, tvořících užitečný datový obsah jednoho OFDM symbolu. Vzhledem k extrémně pomalému spojení s hostitelským PC není možné pro vysílání použít skutečná data z připojeného počítače. Simulovaná data jsou tvořena jednou periodou navzorkované sinusoidy.

Podprogram *gen_twiddle()* naplní vektor *„w“* pomocnými daty pro výpočet IFFT (tzv. twiddle factors).

Podprogram *ok_vektory()* spočítá vektory *„okno1“* a *„okno2“* reprezentující okraje Hanningova okna použité pro úpravu spektra OFDM symbolů.

Podprogram *da_init()* je umístěn ve zdrojovém souboru *prev_init.c*, který spolu se souborem *prev_init.h* tvoří jednoduchý ovladač pro spuštění D/A převodníků a EDMA přenosů. Jednou z funkcí je též nakonfigurování parametrů EMIF rozhranní v použitém rozsahu adres. Vstupními parametry jsou:

- *int sps* – použitá vzorkovací frekvence výstupního analogového signálu,
- *unsigned short *sadr1*, **sadr2* – adresy dvojice bufferů nesoucí obsah celočíselných hodnot zapisovaných do D/A převodníků,

- *int N* – počet komplexních vzorků na jeden OFDM symbol.

Následuje výpis funkce *da_init()* ze souboru *prev_init.c*.

```
void da_init(int sps, unsigned short *sadr1, unsigned short *sadr2, int N)
{
  /*
  sps = samples per second
  sadr1 = prvni buffer
  sadr2 = druhy buffer
  N = pocet komplexnich cisel pro prenos
  */
  //nastaveni CE2: 32-bit asynchro.
  EMIF_RSET(CECTL2,0x31f3c723);
  //Nastaveni casovace:
  hTimer = TIMER_open(TIMER_DEV1, TIMER_OPEN_RESET);
  TIMER_configArgs(hTimer,
    TIMER_CTL_OF(0x3c1),
    TIMER_PRD_OF(  ((FCPU/4/2/sps+1)>>1)<<1  ),
    TIMER_CNT_OF(0)
  );
  //vypnuti+vymaz pripadnych preruseni
  IRQ_reset(IRQ_EVT_EDMAINT);
  EDMA_intDisable(TCCINTNUM);
  EDMA_intClear(TCCINTNUM);
  //Otevreni kanalu a alokace handle:
  hEdma=EDMA_open(EDMA_CHA_TINT1,EDMA_OPEN_RESET);
  if(hEdma==(EDMA_Handle)INV)
  {
    SYS_abort("Vracena obsazena handle k EDMA");
  }
  hEdmaPing=EDMA_allocTable(-1);
  if(hEdmaPing==(EDMA_Handle)INV)
  {
    SYS_abort("Vracena obsazena handle k EDMA");
  }
  hEdmaPong=EDMA_allocTable(-1);
  if(hEdmaPong==(EDMA_Handle)INV)
  {
    SYS_abort("Vracena obsazena handle k EDMA");
  }
  //vytvoreni pomocne reload struktury podle PING
  cfgEdma = cfgEdmaPing;
  //inicializace zdrojovych adres, pocitadel a linku v konfig.
  cfgEdma.src     = EDMA_SRC_RMK((int)sadr1);
  cfgEdmaPing.src = EDMA_SRC_RMK((int)sadr1);
  cfgEdmaPong.src = EDMA_SRC_RMK((int)sadr2);
  cfgEdma.cnt     = EDMA_CNT_RMK(N-1, 2);
  cfgEdmaPing.cnt = EDMA_CNT_RMK(N-1, 2);
  cfgEdmaPong.cnt = EDMA_CNT_RMK(N-1, 2);
  cfgEdma.rld     = EDMA_RLD_RMK(2,hEdmaPong);
  cfgEdmaPing.rld = EDMA_RLD_RMK(2,hEdmaPong);
  cfgEdmaPong.rld = EDMA_RLD_RMK(2,hEdmaPing);
  //naprogramovani EDMA kanalu konfiguracni strukturou
  EDMA_config(hEdma, &cfgEdma);
  //konfigurace PaRAM konfiguracnimi strukturami
  EDMA_config(hEdmaPing, &cfgEdmaPing);
  EDMA_config(hEdmaPong, &cfgEdmaPong);
  //zapnuti preruseni
  IRQ_enable(IRQ_EVT_EDMAINT);
  EDMA_intEnable(TCCINTNUM);
  //zapnuti EDMA kanalu
  EDMA_enableChannel(hEdma);
}
```

Funkce *da_init()* používá datové struktury a API z knihovny Chip Support Library, které umožňují jednoduchý přístup k hardwarovým prostředkům a zajišťují slučitelnost

s jádrem BIOSu. Většina funkcí je založena na použití tzv. handle, což je ukazatel identifikující daný hardwarový prostředek. Všechny operace vztažené např. k danému EDMA kanálu jsou pak definovány daným ukazatelem, čímž jsou vyloučeny potenciální konflikty (např. otevření už otevřeného kanálu). Ve struktuře PaRAM (parameter RAM) jsou vytvořeny dvě předvolby *„cfgEdmaPing"* a *„cfgEdmaPong"*, které zajišťují automatickou cyklickou záměnu bufferů po dokončení přenosu a vytváří tak systém double-buffering. Při zpracování pak procesor plní jeden buffer novými hodnotami, zatímco EDMA v pozadí odesílá data z druhého bufferu na výstupy D/A převodníků. Většina parametrů EDMA předvoleb je definována konstantami v inicializaci datových struktur, jak je zřejmé z následujícího výpisu.

```
//vytvoreni konfiguracni struktury EDMA pro PING prenos
EDMA_Config cfgEdmaPing = {
  EDMA_OPT_RMK(
    EDMA_OPT_PRI_LOW,
    EDMA_OPT_ESIZE_16BIT,
    EDMA_OPT_2DS_NO,
    EDMA_OPT_SUM_INC,
    EDMA_OPT_2DD_NO,
    EDMA_OPT_DUM_IDX,
    EDMA_OPT_TCINT_YES,
    EDMA_OPT_TCC_OF(TCCINTNUM),
    EDMA_OPT_LINK_YES,
    EDMA_OPT_FS_YES
  ),
  EDMA_SRC_OF(0),
  EDMA_CNT_OF(0),
  EDMA_DST_OF((int)DACS0),
  EDMA_IDX_OF((0<<16)|4),
  EDMA_RLD_OF(0)
};
```

6.4 Softwarové přerušení KODER_SWI

Tento proces je spouštěn z hardwarového přerušení EDMA_HWI vždy po dokončení přenosu daného bufferu na výstupní převodníky. Zároveň se začátkem přenosu z druhého bufferu je tak spuštěno generování nových dat, kterými se plní právě uvolněný buffer. Výkon tohoto procesu zabezpečuje funkce *koder()*, jejíž výpis následuje.

```
void koder(void)
{
  //vytvoreni nosnych:
  vytvor_nosne();
  //vlozeni nul:
  vlnuly();
  //IFFT:
    //prvni beh:
  DSPF_sp_ifftSPxSP(sirkafft, &fft_x[0], &w[0], fft_y, brev, sirkafft/4, 0,
                    sirkafft);
    //dekompozice:
  DSPF_sp_ifftSPxSP(sirkafft/4, &fft_x[2*3*sirkafft/4], &w[2*3*sirkafft/4], fft_y,
                    brev, 2, 3*sirkafft/4, sirkafft);
  DSPF_sp_ifftSPxSP(sirkafft/4, &fft_x[2*sirkafft/2], &w[2*3*sirkafft/4], fft_y,
                    brev, 2, sirkafft/2, sirkafft);
  DSPF_sp_ifftSPxSP(sirkafft/4, &fft_x[2*sirkafft/4], &w[2*3*sirkafft/4], fft_y,
                    brev, 2,sirkafft/4, sirkafft);
  DSPF_sp_ifftSPxSP(sirkafft/4, &fft_x[0], &w[2*3*sirkafft/4], fft_y, brev, 2, 0,
                    sirkafft);
  //naplneni vystupniho bufferu, uprava spektra:
  napln_buf();
  //zajisteni konzistence L2 CACHE s externi pameti:
  CACHE_wbL2(outbuf, ((sirkafft+ng)*2*sizeof(unsigned short)), CACHE_WAIT);
```

```
  //prohozeni ukazatele bufferu pro pristi periodu:
  if(ktery==prvni)
  {
    ktery=druhy;
    outbuf=outbuf2;
  }
  else
  {
    ktery=prvni;
    outbuf=outbuf1;
  }
}
```

Na začátku tohoto podprogramu jsou vstupní data namapována na jednotlivé nosné s použitou modulací (QPSK, 16-QAM) a jsou vloženy pilotní signály, což zajišťuje funkce *vytvor_nosne()*. Jednotlivé nosné jsou již generovány s ohledem na pozice nutné pro IFFT. Funkce *vlnuly()* provede menší úpravu vstupního vektoru a následuje IFFT. Pro výpočet zpětné Fourierovy transformace byla použita funkce *DSPF_sp_ifftSPxSP* z knihovny DSP Library [6]. Jedná se o speciální variantu algoritmu, která umožňuje rozdělení výpočtu na několik fází, čímž se snižuje objem současně zpracovávaných dat a zvyšuje se efektivita využití CACHE. Stejně jako všechny funkce z uvedené knihovny je psána v assembleru a ručně optimalizovaná pro dosažení nejvyšší možné rychlosti zpracování.

Na konci podprogramu jsou data z výstupního vektoru IFFT upravena okénkovou funkcí a převedena z formátu s plovoucí řádovou čárkou do celočíselného formátu. Následuje jejich zkopírování do výstupního bufferu, přičemž je provedena kompenzace ofsetových a multiplikativních chyb D/A převodníků.

Vzhledem k umístění bufferů použitých při EDMA přenosech v externí paměti je nutné zajistit tzv. koherenci L2 CACHE [18]. Řadič CACHE totiž nenabízí automatickou aktualizaci dat v externí paměti z odpovídajících lines v interní L2 CACHE před začátkem EDMA přenosu (koherence mezi L1D a L2 CACHE je zajištěna automaticky). Inkriminovaná data tak musí být z L2 CACHE zapsána zpět do externí paměti programově (tzv. write-back), odkud je pak čte řadič EDMA. Pro tuto operaci je použito makro z knihovny Chip Support Library *CACHE_wbL2()*.

Posledním úkonem je přehození ukazatelů na pracovní buffer pro CPU, se kterým bude pracovat při příštím spuštění KODER_SWI.

7 Popis programu realizovaného dekodéru

7.1 Použitá konfigurace, proměnné a uspořádání paměti

Podobně jako v kodéru byl použit konfigurační nástroj BIOSu s obdobnými nastaveními. Mezi odlišnosti patří:

- do interní SRAM byl umístěn vstupní vektor FFT; tento vektor sdílí paměťový prostor s dalšími 4 vektory, které jsou využívány v jiných částech programu, takže se mohou v paměti překrývat,
- ve zbývajícím prostoru interní SRAM byly umístěny statické proměnné (mimo velkých polí), stack o velikosti 2048 bajtů a většina interních proměnných BIOSu při zachování dvoucestné L2 CACHE (32 kB),
- byl nakonfigurován RTDX (real-time data exchange) kanál pro odesílání dat do hostitelského počítače s vyhrazeným bufferem o velikosti 32 kB,
- jako systémový časovač byl nastaven TIMER1, druhý časovač řídí A/D převod,
- vytvořen proces ACQUSITION_SWI typu softwarové přerušení pro obsluhu stavu po zapnutí dekodéru,
- vytvořen proces PRECHOD_SWI typu softwarové přerušení zajišťující přechod z režimu acquisition do režimu tracking,
- vytvořen proces TRACKING_SWI typu softwarové přerušení realizující zpracování vstupního signálu a výstup dat při běžném provozu,
- vedle bufferu pro statistická data byl vytvořen buffer o velikosti 512 bajtů pro objekt Message Log na posílání zpráv do hostitelského počítače.

Všechna pole byla opět zarovnána s ohledem na CACHE. Vzhledem k opačnému nastavení paměti pro proměnné (implicitně interní) bylo nutné u polí definovat umístění v externí paměti. Příslušný kompilační soubor pak obsahuje:

```
SECTIONS {
 .vnitrni: {} > IRAM
 .externi: {} > SDRAM
}
```

Součástí deklarací je v dekodéru též RTDX rozhranní pomocí volání makra *RTDX_CreateOutputChannel()* pro každý ze tří použitých kanálů.

7.2 Inicializace programu, A/D převodník, EDMA

V inicializaci programu je stejně jako u kodéru vygenerován vektor pomocných hodnot pro FFT (twiddle factors), dále je inicializováno RTDX rozhranní a proveden výmaz některých polí. Před spuštěním EDMA přenosů je provedena operace *CACHE_wbInvL2()* pro označení CACHE lines použitých buffery za neplatné (zároveň je proveden jejich přepis do externí SDRAM – tzv. write-back invalidate). Toto zajistí správné načtení bufferů jádrem po skončení EDMA přenosu. Před opuštěním funkce *main()* je inicializován časovač a

dvoukanálový A/D převodník funkcí *ad_init()* a poté je spuštěn EDMA přenos prvních vzorků signálu funkcí *zapni_EDMA()*.

Všechny funkce pro nastavení parametrů A/D převodníku a EDMA byly umístěny do zdrojového souboru *prev_init.c* resp. *prev_init.h*. Nastavení A/D převodníku je poněkud komplikovanější než u D/A, neboť je potřeba vedle periody vzorkujícího časovače specifikovat počet kanálů a provést reset výstupní FIFO. Data jsou do procesoru odesílána z jedné adresy postupným vyčítáním, dokončený A/D převod je signalizován externím přerušením EXT INT7. EDMA kanál proto využívá toto jako synchronizační událost. V průběhu zpracování signálu je potřeba dynamicky měnit délku přijímaných dat, proto je v uvedeném souboru ještě funkce *uprav_EDMA()*, kterou je možné změnit parametry následujícího přenosu.

7.3 Softwarové přerušení ACQUISITION_SWI

7.3.1 Přehled

Na rozdíl od kodéru jsou zde tři různá SWI, na konci EDMA přenosu se zavolá jedno z nich, podle aktuálního módu dekodéru, viz následující výpis.

```
Void EDMA_hwi()
{
  EDMA_intClear(TCCINTNUM); //vynulovani priznaku
  if(stav_dekoderu<6)
  {
    SWI_post(&acquisition_swi); //spusti SW preruseni acquisition_swi
  }
  else
  {
    if(acq_symboly<5)
    {
      SWI_post(&prechod_swi);   //spusti SW preruseni prechod_swi
    }
    else
    {
      SWI_post(&tracking_swi);  //spusti SW preruseni tracking_swi
    }
  }
}
```

Mezi činnosti prováděné v módu acquisition patří:

- konverze 512 komplexních vzorků (přenesených EDMA) do formátu s plovoucí řádovou čárkou a korekce chyby A/D převodníku,
- výpočet korelace přes délku ochranného intervalu (512 vzorků) se vzorky vzdálenými o šířku FFT (2048),
- výpočet průměru s klouzavým oknem přes 512 a následujících 1536 modulů korelace,
- vyhodnocování minima a maxima modulu korelace s pravidelnou aktualizací,
- porovnávání klouzavých průměrů a hledání korelační špičky,
- po nalezení začátku OFDM symbolu výpočet frekvenčního ofsetu z výsledku korelace a změna parametrů EDMA kanálu pro přechod do následujícího módu.

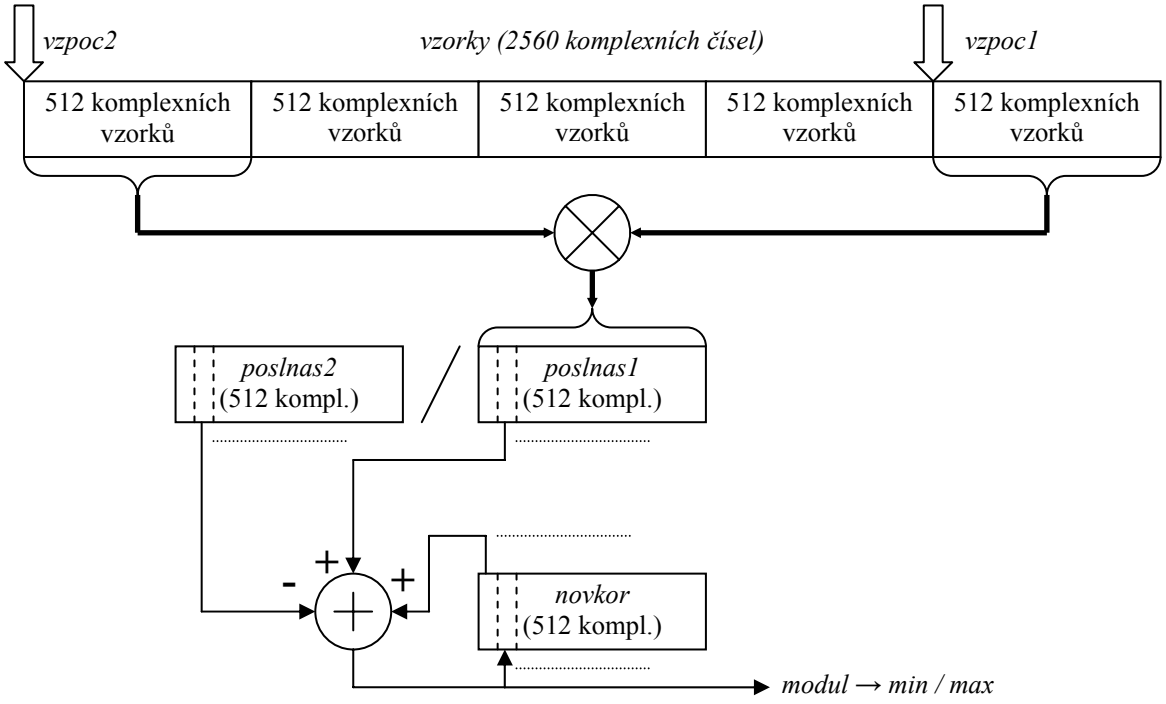
Celý podprogram *acquisition()* je značně rozsáhlý a zdrojový kód je možné najít na přiloženém CD. V následujícím textu jsou stručně popsány některé klíčové prvky použitého algoritmu. Všechny operace jsou založeny na blokovém zpracování, což je výhodné z hlediska využití pipeline, viz kap. 5.2.

7.3.2 Korelace vstupních vzorků

Po spuštění dekodéru se nejprve vektor *„vzorky“* naplní 2560-ti komplexními vzorky vstupního signálu, tj. proběhne 5 EDMA přenosů, každý o 512 vzorcích. Poté se pronásobí vzorek po vzorku první a poslední blok, jak je vidět na Obr. 30, a výsledky násobení pro každou dvojici vzorků se umístí do vektoru *„poslnas1“*. Na poslední pozici vektoru *„novkor“* se vloží součet všech hodnot vektoru *„poslnas1“*, čímž se získá první hodnota korelace. Na konci podprogramu se vždy inkrementují ukazatele *„vzpoc2“* a *„vzpoc1“* o 512 pozic (cyklicky) a prohodí se vektory *„poslnas1“* a *„poslnas2“*.

Při opakovaném spouštění je pak vždy naplněn vstupními vzorky následující blok ve vektoru *„vzorky“* (systém FIFO) a spočítá se nový vektor *„poslnas1“*. Následuje výpočet nových hodnot korelace, nyní již všech 512, a to podle systému na obrázku. Vychází se z faktu, že následující hodnota korelace se od předchozí liší pouze díky dvěma krajním hodnotám intervalu, ze kterého je počítána. Stačí tedy přičíst k minulému výsledku novou hodnotu pronásobení z *„poslnas1“* a odečíst odpovídající starou hodnotu z *„poslnas2“* (který nese hodnoty *„poslnas1“* z minulého běhu). Pro výpočet 512 komplexních hodnot korelace ve vektoru *„novkor“* tak stačí 512 komplexních násobení a součtů trojic čísel.

Na obrázku je též uvedeno zjišťování hodnot *„min“* / *„max“*, tj. minima a maxima modulu korelace. Tyto proměnné jsou duplikované, takže se vždy po dobu např. 5 symbolů jeden pár používá (viz dále) a druhý se mezitím aktualizuje.

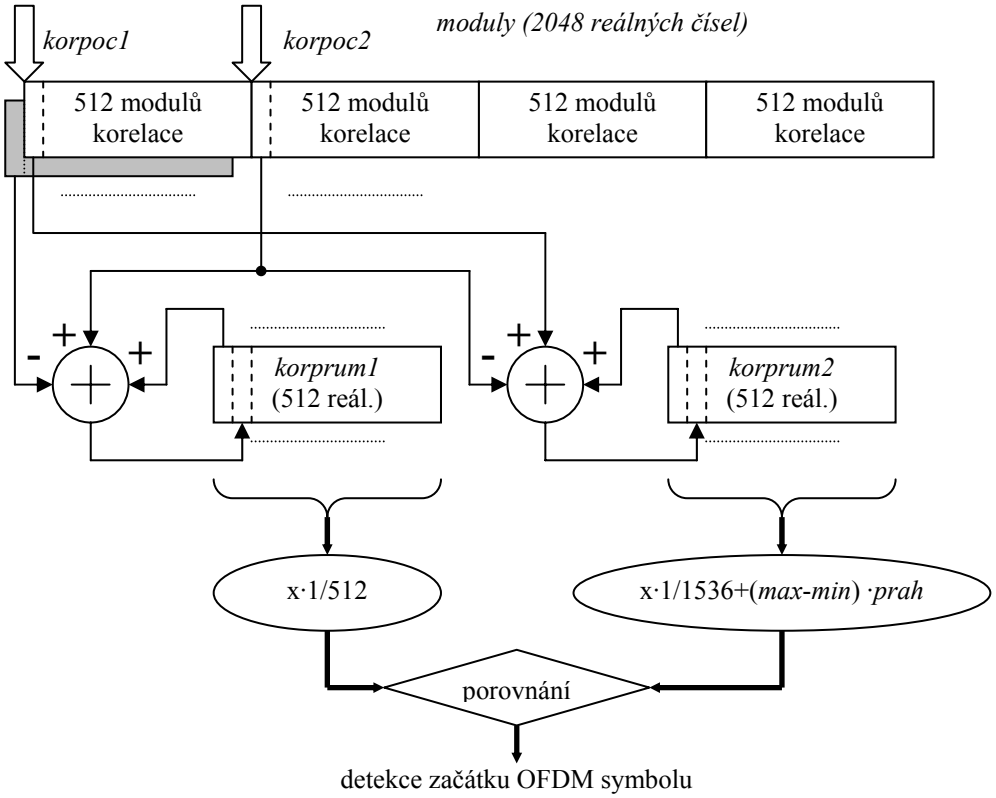


Obr. 30 Algoritmus korelace

7.3.3 Výpočet průměrů modulů korelace

Jak již bylo uvedeno, hledání korelační špičky je provedeno přes porovnávání klouzavých průměrů, viz Obr. 31. Nejprve je potřeba shromáždit 2048 výsledků korelace, jejichž moduly se zapisují do vektoru *„moduly“*. V průběhu je prováděn součet prvních 512 hodnot do posledního prvku vektoru *„korprum1“* a následujících 1536 do téhož prvku vektoru *„korprum2“*. Tím jsou získány první dvě hodnoty průměrů (před podělením počtem prvků). Na konci podprogramu jsou vždy cyklicky inkrementovány ukazatele *„korpoc1“* a *„korpoc2“* o 512.

Při dalších opakováních podprogramu jsou cyklicky zapisovány nové hodnoty modulů (opět systém FIFO), přičemž pro výpočet *„korprum1“* je použita stará hodnota bloku před jeho přepsáním, jak je v obrázku naznačeno. Pro výpočet 512 nových průměrů se využívá minulých hodnot, podobně jako při výpočtu korelace. Na jeden vzorek průměru tedy stačí součet tří čísel (viz sumátory v obrázku). Jednotlivé prvky *„korprum1“* a *„korprum2“* jsou pak škálovány, aby mohly být následně porovnány. Střed intervalu, kdy jsou upravené hodnoty *„korprum1“* větší než *„korprum2“* se využije pro stanovení začátku OFDM symbolu.



Obr. 31 Algoritmus výpočtu průměrů modulů korelace

7.3.4 Frekvenční korekce, změna parametrů EDMA

Po stanovení začátku OFDM symbolu se použije vzorek korelace přibližně z poloviny ochranného intervalu pro stanovení zlomkové hodnoty frekvenční korekce podle výše uvedeného vzorce.

V závislosti na pozici začátku OFDM symbolu v rámci vstupního bufferu je provedena změna předvoleb EDMA řadiče. Další načítané bloky mají pak délku 2560 vzorků (tedy celý OFDM symbol včetně ochranného intervalu) a jejich začátek je umístěn do poloviny ochranného intervalu, čímž je zaručeno, že při dalším příjmu nedojde k ICI (inter-carrier interferences). Pro další zpracování je po dokončení následujícího EDMA přenosu již volán proces PRECHOD_SWI.

7.4 Softwarové přerušení PRECHOD_SWI

Tento proces zabezpečuje provedení nezbytných operací na několika OFDM symbolech, než je možné přejít do módu tracking. Na začátku volaného podprogramu *prechod()* je nejprve převeden načtený vektor z celočíselné podoby do reprezentace v plovoucí řádové čárce, přičemž jsou též vykompenzovány chyby A/D převodníku. Součástí zpracování vstupního vektoru je též úprava fáze signálu v souladu s dříve zjištěnou hodnotou frekvenčního ofsetu. Podprogram *fazderot()* je uveden v následujícím výpisu.

```
void fazderot(float *data,int okolik)
{
  /*
  data = ukazatel na realnou cast komplexniho vektoru pro upravu
  okolik = kolikanasobek rotace se ma provest (pouzito po vynechani vzorku vst.
           signalu)
  */
  float refd,imfd,pomre,pomim;  //pomocne promenne
  static float fazaku=0;        //fazovy akumulator
  //aktualizace faze:
  fazaku=fazaku - (deltafrek+(float)intdeltafrek)*(2*pi/sirkafft)*okolik;
  if(fazaku>(100*2*pi))
  {
    fazaku=fazaku-100*2*pi;
  }
  else if(fazaku<(-100*2*pi))
  {
    fazaku=fazaku+100*2*pi;
  }
  refd=cosf(fazaku);
  imfd=sinf(fazaku);
  pomre=data[0];
  pomim=data[1];
  //uprava faze signalu:
  data[0]=pomre*refd-pomim*imfd;
  data[1]=pomre*imfd+pomim*refd;
}
```

V podprogramu *fazderot()* jsou využity funkce *cosf()* a *sinf()* z knihovny FastRTS67X [21], které jsou několikanásobně rychlejší než standardní implementace v jazyce C. Algoritmus úpravy fáze byl uveden v kap. 4.2.3. Proměnná realizující fázový akumulátor je držena v rozsahu $\pm 100 \cdot 2\pi$, aby nedošlo ke zhoršení přesnosti zaokrouhlováním.

Jakmile je připraven vstupní vektor, je možné provést Fourierovu transformaci. Podobně jako v kodéru je použita optimalizovaná funkce z knihovny DSP Library, viz následující výpis.

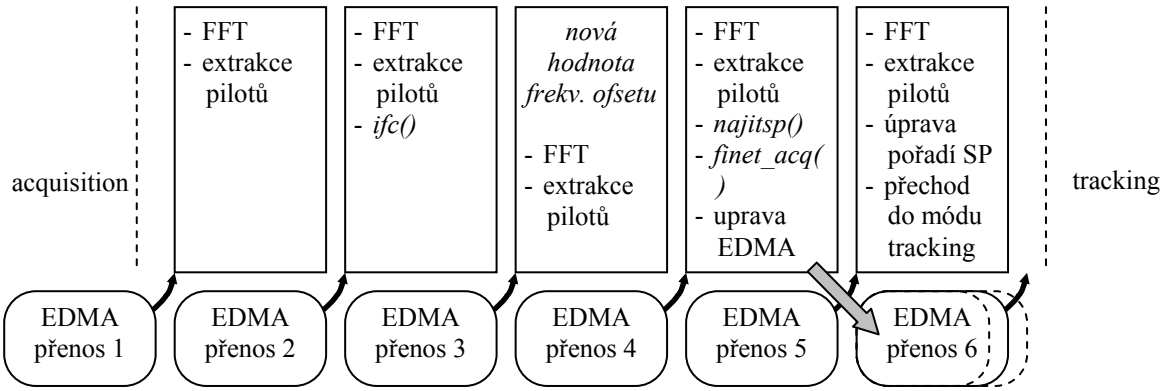
```
//FFT:
  //prvni beh:
DSPF_sp_fftSPxSP(sirkafft,&fft_x[0],&w[0],fft_y,brev,sirkafft/4,0,sirkafft);
  //dekompozice:
DSPF_sp_fftSPxSP(sirkafft/4,&fft_x[2*3*sirkafft/4],&w[2*3*sirkafft/4],fft_y,brev,2,
                 3*sirkafft/4,sirkafft);
```

```
DSPF_sp_fftSPxSP(sirkafft/4,&fft_x[2*sirkafft/2],&w[2*3*sirkafft/4],fft_y,brev,2,
                 sirkafft/2,sirkafft);
DSPF_sp_fftSPxSP(sirkafft/4,&fft_x[2*sirkafft/4],&w[2*3*sirkafft/4],fft_y,brev,2,
                 sirkafft/4,sirkafft);
DSPF_sp_fftSPxSP(sirkafft/4,&fft_x[0],&w[2*3*sirkafft/4],fft_y,brev,2,0,sirkafft);
```

V dalším zpracování se z výstupního vektoru FFT zjistí pilotní nosné (podprogram *odsnuly()*). Další průběh podprogramu *prechod()* je již závislý na počtu dosavadních volání (1 až 5). Zjištěné nosné jsou ukládány střídavě do dvou vektorů, aby byly k dispozici vždy dva po sobě následující symboly. Sekvence prováděných činností (s EDMA přenosy běžícími na pozadí) je ilustrována na Obr. 32. Patří mezi ně:

- 1. symbol – uložení do prvního vektoru,
- 2. symbol – uložení do druhého vektoru, využití obou vektorů ve funkci *ifc()* pro výpočet frekvenčního ofsetu v násobcích rozestupu nosných (celočíselný frekv. ofset),
- 3. symbol – uložení do prvního vektoru; v průběhu jeho načítání EDMA řadičem byla zjištěna nová hodnota frekvenčního ofsetu, takže jej nelze použít ve dvojici s předchozím symbolem pro další zpracování (musí se načíst další symbol),
- 4. symbol – uloží se do druhého vektoru, s použitím předchozího vektoru se najde aktuální vzor rozmístění scattered pilotů (podprogram *najitsp()*), zjistí se přesná hodnota časového ofsetu začátku užitečné části OFDM symbolu (podprogram *finet_acq()*) a v souladu s tím se změní konfigurace EDMA řadiče, aby došlo k odpovídajícímu vykompenzování;
- 5. symbol – upraví se indikátor pořadí scattered pilotů, provede se přechod do módu tracking.

Na obrázku je naznačeno, že v průběhu zpracování 4. symbolu je upravena předvolba pro EDMA přenos 6. symbolu (právě probíhající přenos změnit nelze), tato úprava spočívá ve změně počtu přenášených vzorků. Aby nedošlo ke ztrátě užitečných dat vlivem případného zkrácení přenosu, je začátek načítaných vzorků umístěn vždy do poloviny ochranného intervalu, takže užitečná část je situována v prostředku bufferu. Vzhledem k možnosti prodloužení přenosu nad 2560 vzorků jsou příslušné vstupní buffery v paměti alokovány s rezervou 64 míst.



Obr. 32 Průběh zpracování při přechodu z acquisition do tracking

7.5 Softwarové přerušení TRACKING_SWI

V režimu tracking dekodér přijímá OFDM signál a poskytuje dekódovaná data. Kromě toho musí být kontinuálně udržovaná synchronizace s přijímaným signálem. Mezi činnosti prováděné v tomto módu patří:

- konverze 2048 komplexních vzorků (z 2560±*úprava* přenesených EDMA) užitečné části OFDM symbolu do formátu s plovoucí řádovou čárkou a korekce chyby A/D převodníku,
- ošetření koherence L2 CACHE (*CACHE_wbInvL2()*),
- fázová úprava vzorků v souladu s aktuálně zjištěnou hodnotou frekvenčního ofsetu (podprogram *fazderot()*),
- Fourierova transformace vstupního vektoru (*DSPF_sp_fftSPxSP()*),
- extrakce užitečných nosných a pilotních signálů z výstupního vektoru FFT (*odsnuly()*),
- aktualizace časového ofsetu (*finet_tr()*) a příslušná změna parametrů EDMA (*uprav_EDMA()*),
- aktualizace zlomkového frekvenčního ofsetu (*finefrek_tr()*),
- provedení ekvalizace nosných (*ekvalizer()*),
- odstranění pilotů z vektoru užitečných nosných (*odspil()*),
- QPSK/16QAM demodulace subnosných (*demap()*),
- odeslání výstupních dat, informací o odhadnuté CTF (channel transfer function) a konstelaci subnosných do hostitelského počítače přes RTDX,
- aktualizace pořadí scattered pilotů, prohození ukazatelů bufferů pro příští volání.

Zdrojový kód většiny uvedených podprogramů je příliš rozsáhlý, než aby zde mohl být uveden (viz přiložené CD). Za zmínku stojí rutina *odsnuly()*, ve které byla použita další z funkcí knihovny DSP Library, a to *DSPF_sp_blk_move()* pro rychlý přesun bloku dat, viz následující výpis.

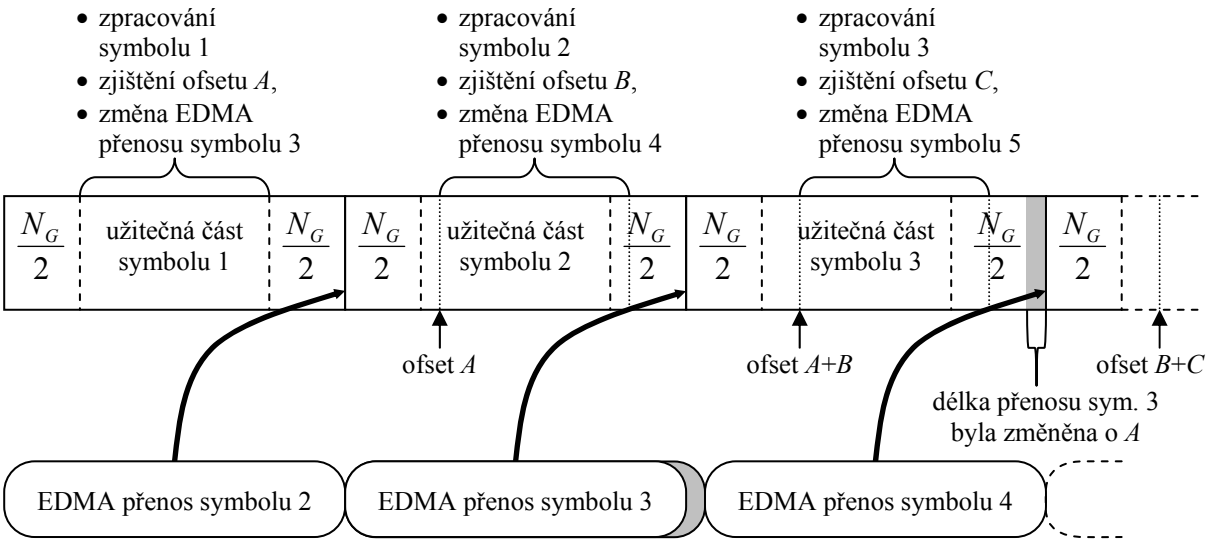
```
void odsnuly()
{
  float* p_prsym;
  if(ktery==prvni) p_prsym=prsym1; else p_prsym=prsym2;
  DSPF_sp_blk_move(&fft_y[re(sirkafft/2+(sirkafft-nosnych+1)/2)],&p_prsym[0],
                   (sirkafft/2-(sirkafft-nosnych+1)/2)*2);
  DSPF_sp_blk_move(&fft_y[0],&p_prsym[re(sirkafft/2-(sirkafft-nosnych+1)/2)],
                   (sirkafft/2-(sirkafft-nosnych-1)/2)*2);
}
```

Funkce *DSPF_sp_blk_move()* nahrazuje použití cyklu *for* a provádí dvojité rozvinutí cyklu (tzv. unrolling) a tzv. cache touching.

V podprogramech *finet_tr()* a *finefrek_tr()* je nutné počítat argument (úhel) komplexního čísla. Pro realizaci této operace byla použita funkce *atan2f()* z knihovny FastRTS67X. Tato funkce má dva vstupní argumenty (reálná a imaginární část) a provádí automatické ošetření speciálních stavů (nulovost jednoho z argumentů), takže nevznikají chyby typu dělení nulou. Stejně jako v předešlých případech se opět jedná o výpočetně vysoce efektivní implementaci.

V ekvalizéru se po zjištění komplexního vektoru CTF musí provést jeho reciproka, aby jím bylo možno pronásobit vektor nosných. V tomto případě se nelze vyhnout operaci dělení, která patří k nejnáročnějším výpočtům v technice DSP, neboť je pro ni jen omezená nebo žádná podpora v instrukčním souboru. Pro zajištění maximální efektivity bylo využito vlastností výpočtu usměrnění komplexního čísla. Celý výpočet je rozdělen na násobení vektorů reálné a imaginární části CTF vektorem $1/x$, kde x představuje vektor součtů čtverců reálných a imaginárních částí CTF. Vektorovou operaci $1/x$ zajišťuje funkce z knihovny DSP Library *DSPF_sp_vecrecip()*, přičemž vektor x je počítán s výpočtem CTF ve stejné smyčce (ukládá se do odděleného pole). Použité vektory v ekvalizéru sdílí paměťový prostor se vstupním vektorem FFT a jsou tedy umístěny v interní datové SRAM, takže je k nim rychlý přístup.

Kontinuální časová synchronizace pozice začátku OFDM symbolu vyžaduje po každém symbolu aktualizaci parametrů EDMA řadiče. Jak již bylo uvedeno výše, není možné ovlivnit parametry již probíhajícího přenosu ale až toho následujícího. Pro zajištění možnosti okamžité kompenzace je použito zvláštního ukazatele do vstupního bufferu, který vzhledem k redundanci délky tohoto bufferu umožňuje změnu začátku zpracovávaných vzorků (v mezích daných polovinou ochranného intervalu na obě strany). Celý proces je pro tři po sobě následující symboly ilustrován na Obr. 33. Při zpracování prvního symbolu je zjištěn ofset A, o který se změní nominální délka EDMA přenosu pro symbol 3. Aby při zpracování symbolu 2 byl již ofset vykompenzován, použijí se z načteného bufferu data posunutá o A oproti nominální pozici. U symbolu 2 je následně zjištěn ofset B oproti aktualizované pozici začátku. Protože kompenzace EDMA ze symbolu 1 se projeví až v symbolu 4, musí být v symbolu 3 použit ofset v bufferu o hodnotě $A+B$. V každém následujícím symbolu je pak použit ofset dvou předchozích hodnot, tj. $B+C$ v symbolu 4 atd.



Obr. 33 Průběh aktualizace časové synchronizace

8 Výsledky realizace kodéru a dekodéru

8.1 Výpočetní náročnost kodéru

V kodéru i dekodéru je maximální vzorkovací frekvence omezená dobou výkonu příslušného softwarového přerušení, které zpracovává jeden OFDM symbol (plus minimální režie BIOSu). Za tuto dobu je přeneseno 2560 vzorků signálu, takže maximální vzorkovací frekvence je

$$f_{vz\,\max} = \frac{2560}{\text{doba zpracování OFDM symbolu}}. \tag{22}$$

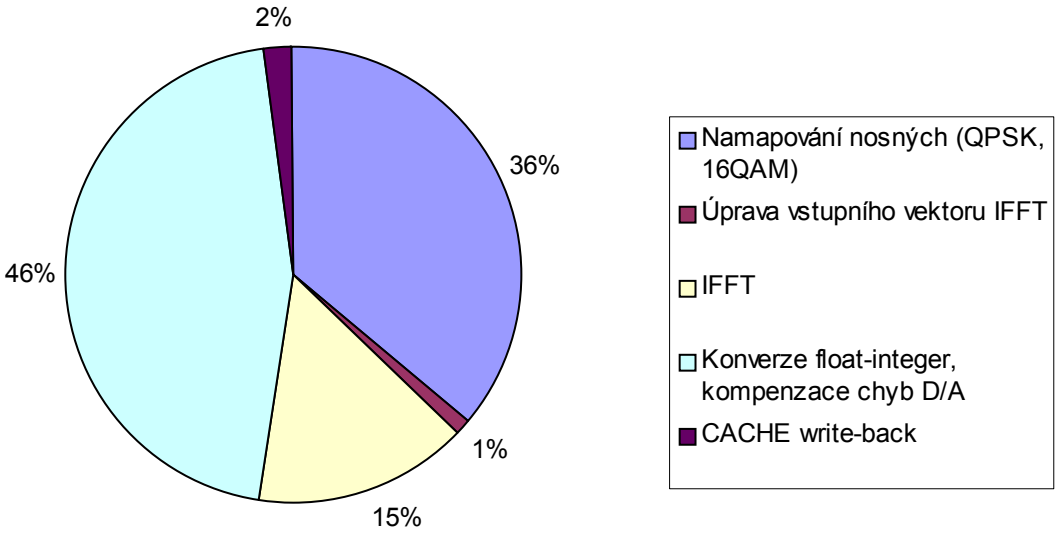
V případě kodéru jde o proces KODER_SWI. Při nastaveném maximálním stupni optimalizace překladače byla změřena doba zpracování cca 2900 µs, čemuž odpovídá maximální dosažitelná vzorkovací frekvence cca **880 kHz**, viz výsledek analýzy Statistics View na Obr. 34. Při této vzorkovací frekvenci je bitový tok přenášených dat cca 1,04 Mbit/s (QPSK), resp. 2,09 Mbit/s (16QAM). Normou [1] vyžadovaná vzorkovací frekvence je přibližně 9,1 MHz, kodér tedy nedokáže běžet v reálném čase. Kodér zabírá 92249 B paměti.

Statistics View

STS	Count	Total	Max	Average
koder_swi	7190	20494646.45 µs	2898.83 µs	2850.44
TSK_idle	0	0.00 µs	-14316557.65 µs	0.00
IDL_busyObj	1.34677e+006	-1.15699e+008	-81	-85.9085

Obr. 34 Výsledek analýzy kodéru

Měřením uplynulých hodinových cyklů jednotlivých částí uvedeného procesu bylo zjištěno rozložení celkového výpočetního času mezi části kodéru, viz Obr. 35.



Obr. 35 Rozložení zátěže v kodéru

8.2 Výpočetní náročnost dekodéru

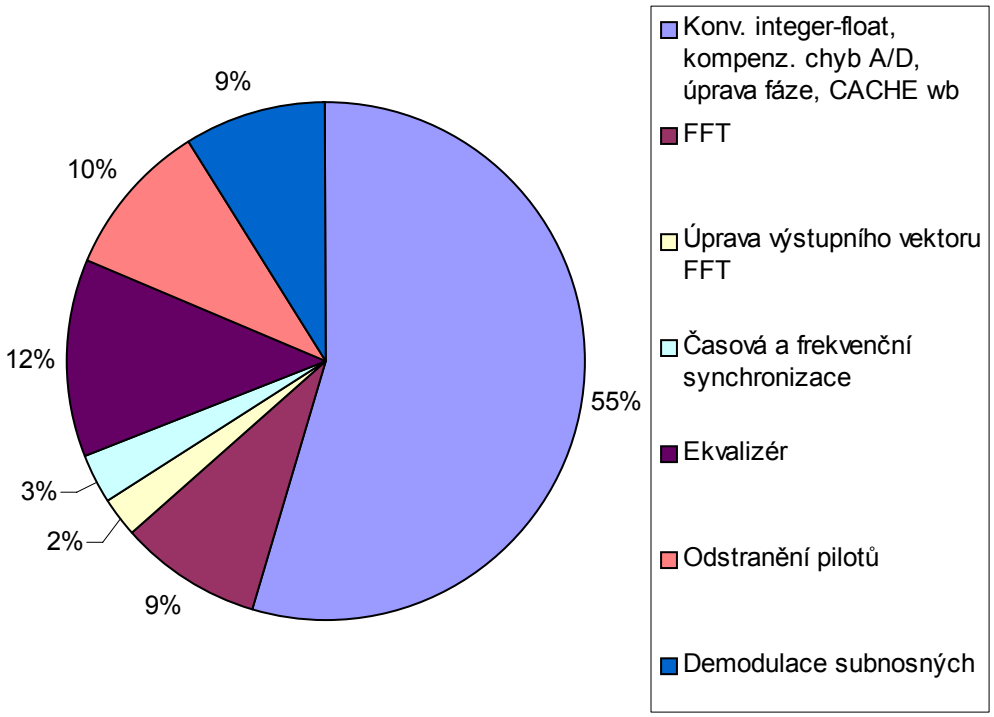
Dekodér používá tři různé procesy pro zpracování signálu podle aktuálního stavu, viz kap. 7. Jako nejnáročnější se ukázal být proces TRACKING_SWI trvající cca 7300 µs, který zabezpečuje všechny funkce dekodéru při normálním běhu (po zasynchronizování). Tomu odpovídá maximální dosažitelná vzorkovací frekvence cca **350 kHz**. Statistika výpočetní náročnosti je uvedena na Obr. 36. Při tomto měření byla vypnuta RTDX komunikace předávající velké objemy dat hostitelskému počítači, neboť slouží pouze k ladícím účelům a představuje enormní výpočetní zátěž. V případě procesu ACQUISITION_SWI je nutno brát v úvahu, že narozdíl od ostatních dvou zpracovává 512 vzorků (oproti 2560). Nicméně výpočetní náročnost všech tří procesů vztažená na jeden vzorek je řádově srovnatelná. Dekodér zabírá 214500 B v paměti.

Statistics View

STS	Count	Total	Max	Average
acquisition_swi	35	19246.21 µs	845.52 µs	549.89
tracking_swi	385	2747655.76 µs	7222.16 µs	7136.77
prechod_swi	5	24582.69 µs	5375.92 µs	4916.54
TSK_idle	0	0 inst	-2147483648 inst	0.00
IDL_busyObj	340661	-3.12974e+007	-81	-91.8726

Obr. 36 Výsledek analýzy dekodéru

Rozložení zátěže mezi jednotlivé části procesu TRACKING_SWI ukazuje Obr. 37.



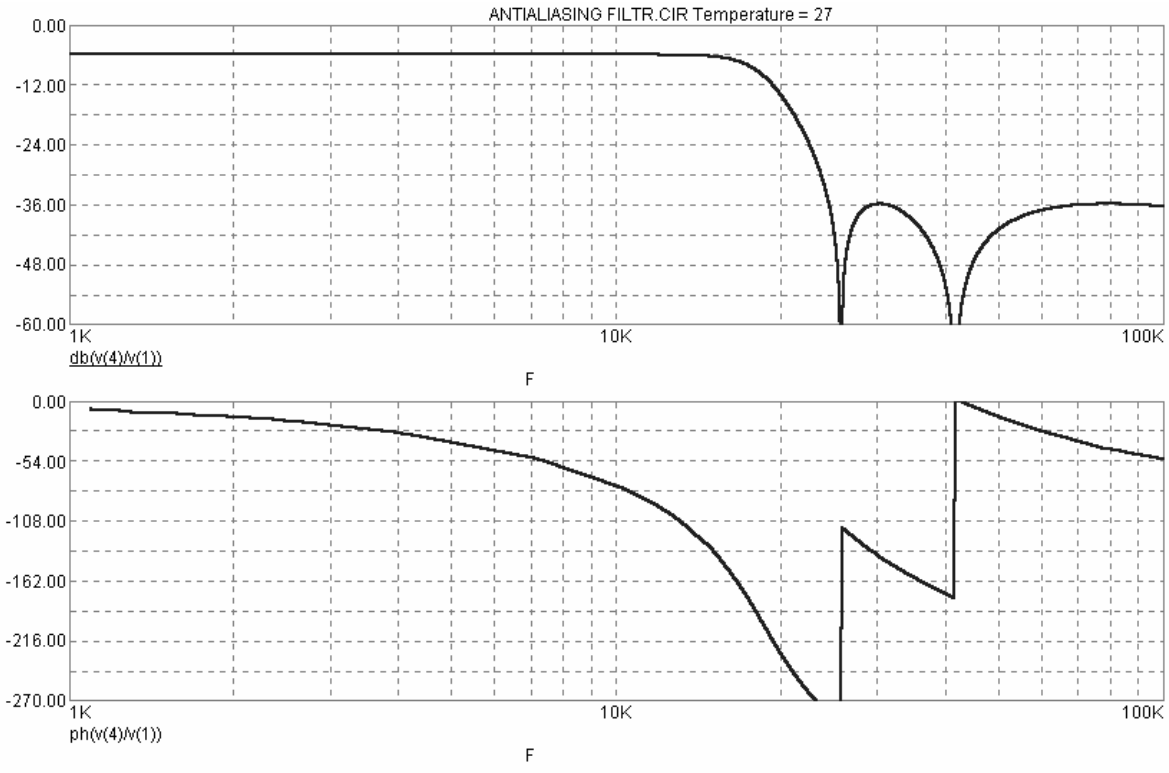
Obr. 37 Rozložení zátěže v dekodéru

U kodéru i dekodéru spotřebuje významnou část procesorového času převod formátu čísel mezi celočíselnou reprezentací (s kterou pracují AD/DA převodníky) a typu s plovoucí

řádovou čárkou (s kterou pracuje jádro procesoru). Je zřejmé, že při použití fixed-point zpracování všech výpočtů by došlo k výrazné úspoře času. Zároveň by se ušetřil jeden vektor 2560 komplexních čísel, protože FFT/IFFT by operovala přímo se vstupním resp. výstupním bufferem. Jako příklad vhodné řady fixed-point DSP pro možnost dosažení práce v reálném čase lze uvést TMS320C6414T/15T/16T případně TMS320C645x. Tyto procesory pracují na mnohonásobně vyšších frekvencích; v této aplikaci by velmi významným faktorem byla také podstatně větší velikost vnitřních pamětí, čímž by zcela odpadl problém přístupu do pomalé externí SDRAM.

8.3 Analogový filtr

Použité převodníky nejsou vybaveny antialiasingovými filtry. Systém sice funguje i bez nich, pokud však vlivem nepatrně rozdílné vzorkovací frekvence v kodéru a dekodéru dochází k úpravám pozice FFT okna, nastávají v přenosu nespojitosti. Proto byl vyroben dvojitý analogový filtr typu dolní propust 5. řádu s inverzní Čebyševovou aproximací, který je možné zapojit mezi kodér a dekodér. Filtr je navržen pro vzorkovací frekvenci 36 kHz, jeho mezní frekvence je tedy 18 kHz (při rychlejších přenosech je problém s pomalým RTDX rozhranním, viz dále). Průběh modulové a argumentové charakteristiky filtru je uveden na Obr. 38.

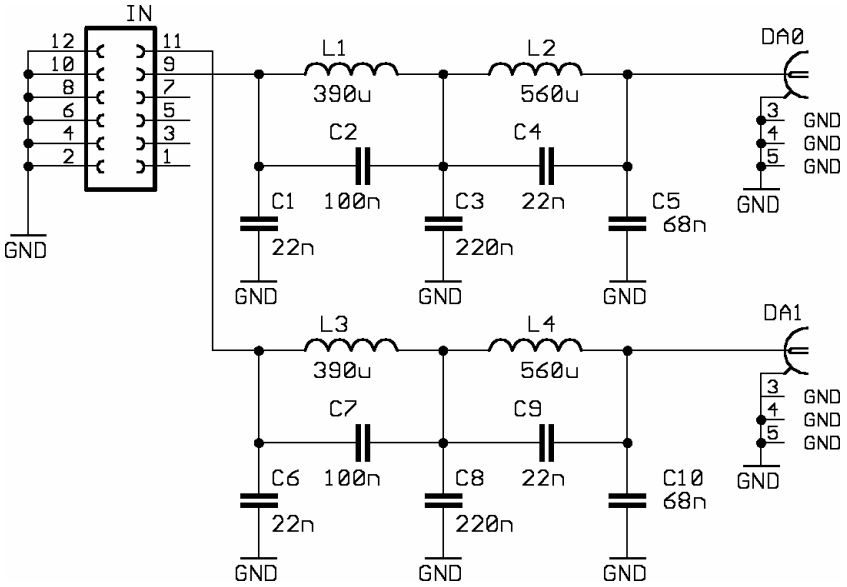


Obr. 38 Frekvenční charakteristiky analogového filtru

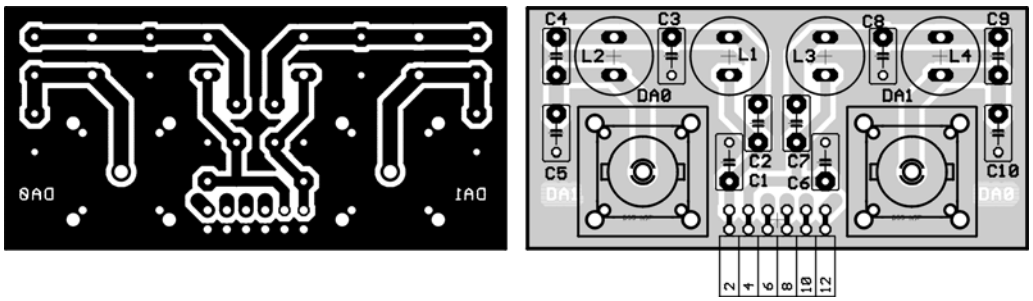
Tento filtr nemá v propustném pásmu ideálně plochou modulovou charakteristiku, což lze s výhodou využít pro demonstraci schopností použitého ekvalizéru.

Schéma zapojení filtru je na Obr. 39, motiv plošného spoje a osazení součástek na Obr. 40. Hodnoty součástek jsou vybrány z vyráběné řady, takže frekvenční charakteristiky

nejsou zcela shodné s teoreticky navrženými, v dané aplikaci to však nemá velký význam. Filtr se zasune vstupním konektorem do pinové lišty na desce s převodníky kodéru. Výstupní konektory jsou typu BNC pro připojení koaxiálních kabelů.



Obr. 39 Schéma analogového filtru



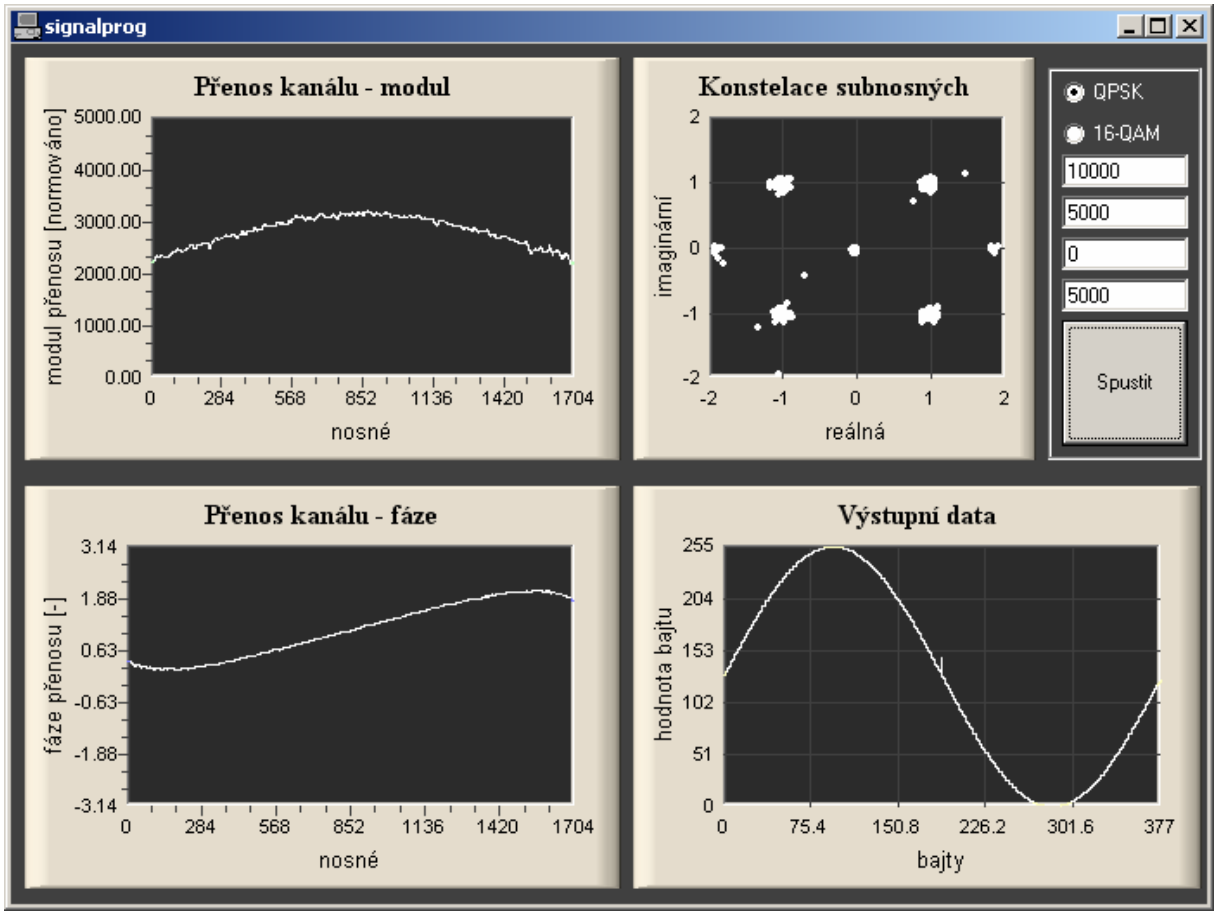
a) **b)**

Obr. 40 a) Deska plošného spoje analogového filtru (1:1) a b) osazení součástek

8.4 Demonstrační program

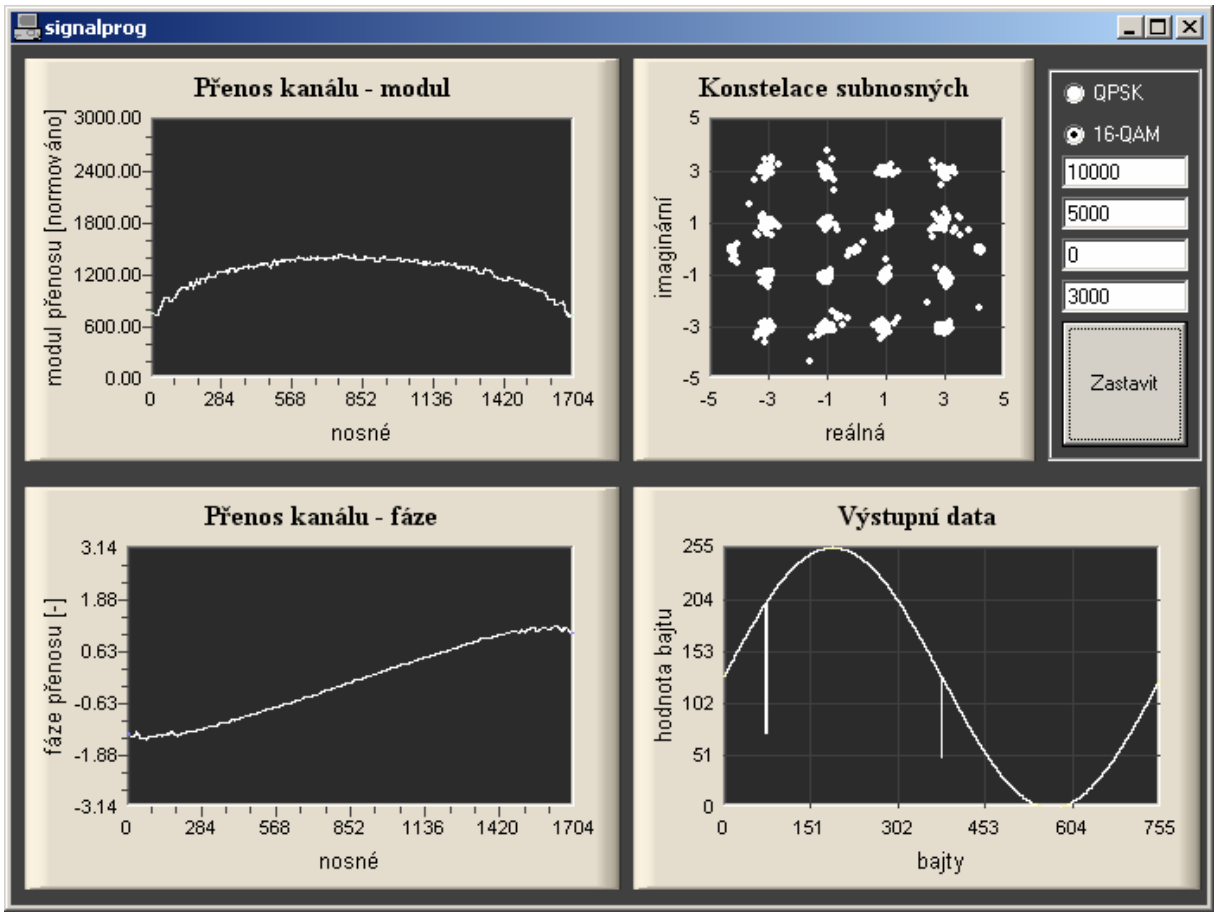
Pro demonstraci funkce dekodéru byla v jazyce Visual Basic napsána aplikace, která získává ze signálového procesoru výstupní data, vektor CTF a konstelační diagram subnosných. Komunikace probíhá přes rozhranní RTDX a je zprostředkována knihovnou RTDXINT.DLL dodávanou s nástrojem Code Composer Studio. Tato knihovna však obsahuje chyby, takže komunikace není příliš stabilní. Emulátor protokolu JTAG od firmy Texas Instruments navíc nabízí pouze velice nízkou přenosovou rychlost, která byla na použitém počítači dále omezena extrémními nároky tohoto emulátoru. Výsledkem je přenos řádu stovek bajtů za sekundu, přesto je možné s danými omezeními sledovat průběh uvedených dat.

Na Obr. 41 je vidět výstup demonstračního programu při použití modulace subnosných QPSK a vřazení analogového filtru. Z modulové a fázové charakteristiky kanálu (vypočtená CTF) je vliv filtru dobře patrný. Nulová frekvence je situována na prostřední nosné, takže modul přenosu na obě strany mírně klesá. K fázové charakteristice filtru se ještě přičítá určitý posuv způsobený korekcí zbytkové odchylky pozice FFT okna, kterou rovněž ekvalizér zajišťuje. Rozptyl bodů konstelačního diagramu je způsoben zejména kvantizačním šumem při DA/AD převodech a zeslabením signálu po průchodu filtrem. Výstupními daty jsou vzorky jedné periody sinusoidy, jak byly vygenerovány v kodéru.



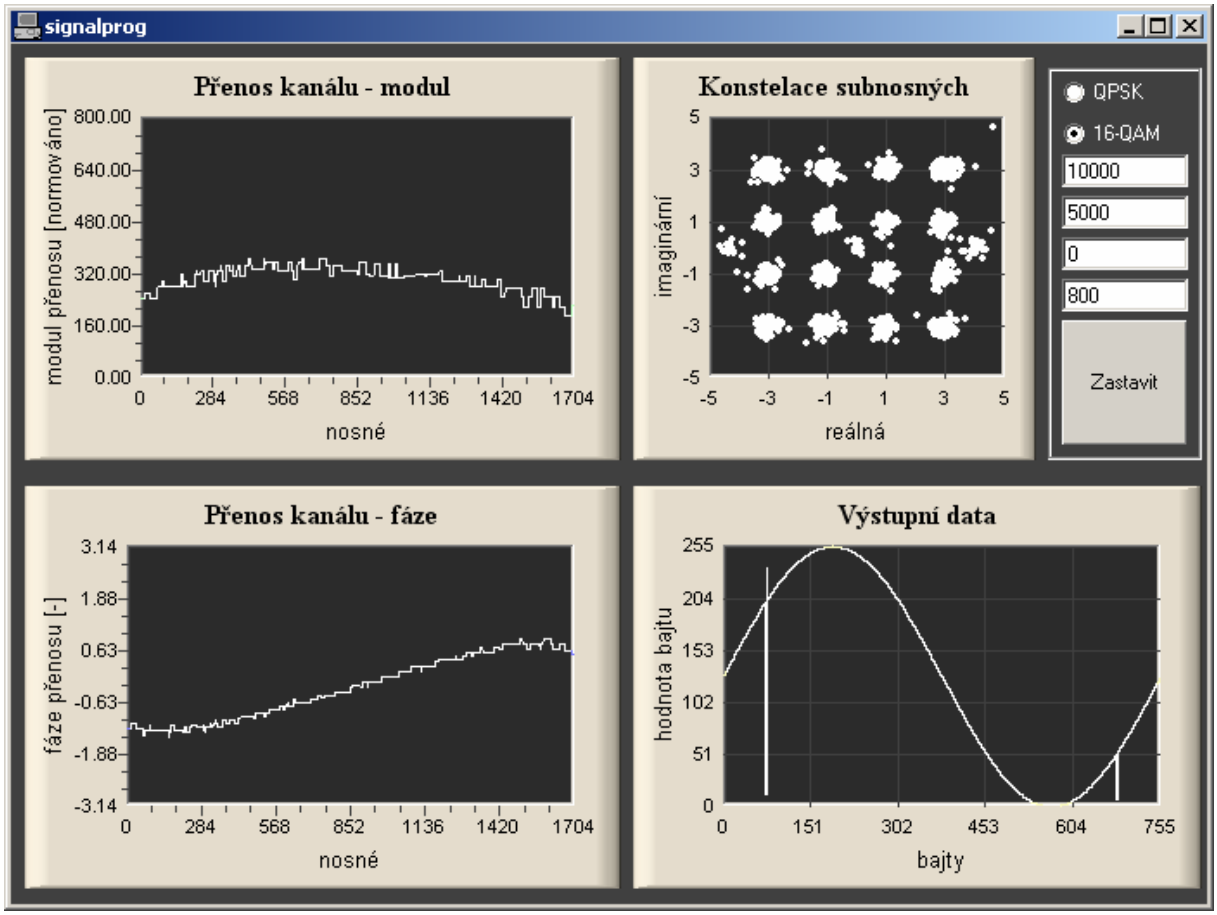
Obr. 41 Výstup dekodéru - QPSK, použit analogový filtr

Na následujícím Obr. 42 je zachycen tentýž stav, tentokrát s modulací subnosných 16QAM. Viditelný rozptyl konstelačních bodů je poněkud větší, což je dáno menší vzdáleností oproti QPSK. Ve výstupních datech jsou vidět chyby vzniklé při dekódování.



Obr. 42 Výstup dekodéru - 16QAM, použit analogový filtr

Poslední ukázkou je Obr. 43. Tentokrát byla v kodéru nastavena 4x menší amplituda výstupního signálu, což se zřetelně projevilo na rozptylu konstelace, ve výstupních datech je více chyb. Vypočtené charakteristiky kanálu jsou též poznamenané větším šumem. Dekodér však stále spolehlivě udržuje optimální synchronizaci.



Obr. 43 Výstup dekodéru - 16QAM, použit analogový filtr, 4x menší amplituda

Závěr

Byla uvedena struktura kodéru OFDM se všemi funkčními bloky potřebnými pro vytvoření signálu pro vysílač DVB-T v základním pásmu podle normy [1]. Vstupem zkoumaného systému je multiplex více audiovizuálních kanálů zakódovaných pomocí MPEG-2 (případně MPEG-4 AVC), výstupem je komplexní obálka, tj. vysílaný signál v základním pásmu.

Z výše uvedené struktury byly vybrány funkční bloky nezbytné pro vytvoření OFDM signálu ze sériového bitového toku bez zabezpečení dat proti chybám. Tyto bloky byly následně naprogramovány v jazyce C. Funkce těchto bloků byla ověřena simulacemi v programu Matlab, jejichž grafický výstup je uveden v příslušných kapitolách.

Na základě studia literatury bylo sestaveno blokové schéma dekodéru OFDM pro dekódování signálu z výše popsaného kodéru při uvažování možnosti použití reálného přenosového kanálu. Funkce všech bloků byla naprogramována v jazyce C a též odsimulována v Matlabu. Tento dekodér je vybaven kompletní synchronizační strukturou, takže toleruje určitý časový i frekvenční ofset přijímaného signálu. Pomocí ekvalizéru je schopen vykompenzovat frekvenční charakteristiku kanálu a provést demodulaci nosných. Výstupem je datový tok, který odpovídá vstupnímu datovému toku kodéru (s chybovostí závislou na neideálním přenosovém kanálu).

Algoritmus kodéru a dekodéru byl implementován do DSP TMS320C6711 firmy Texas Instruments s využitím dvojice vývojových desek C6711 DSK a rozšiřujících modulů s AD/DA převodníky DSP STAR AD/DA od firmy ND-Tech Co., Ltd. Jako vývojové prostředí byl použit program Code Composer Studio v. 3.1 od firmy Texas Instruments. Přenosové médium mezi realizovaným kodérem a dekodérem tvoří dvojice koaxiálních kabelů případně s vřazeným analogovým filtrem typu dolní propust.

Program pro kodér i dekodér byl zoptimalizován z hlediska rychlosti výpočtu. Na kritické části kódu bylo použito knihoven DSP Library a FastRTS67X, které obsahují ručně optimalizované assemblerovské funkce. Při psaní programu bylo počítáno se specifickými prvky struktury použitého signálového procesoru (instrukční pipeline, organizace paměti a CACHE, řadič přímého přístupu do paměti EDMA). Vzhledem k dosažitelnému výpočetnímu výkonu DSP však kodér a dekodér přenáší data pomaleji, než vyžaduje standard DVB-T.

Zdrojové kódy i matlabovské soubory lze nalézt na přiloženém CD.

Seznam použité literatury a jiných zdrojů

[1] *European Standard ETSI EN 300 744*: *Digital Video Broadcasting (DVB); Framing structure, channel coding and modulation for digital terrestrial television* [online]. Ver. 1.5.1. 11-2004. [cit. 25.4.2007]. Dostupné z <http://pda.etsi.org/pda/home.asp?wki_id=3lWRYYk9dl78GFGGBl0Ly>.

[2] ŘÍČNÝ, V.; KRATOCHVÍL, T. *Základy televizní techniky*. Brno: VUT, 2002. 178 s. Fakulta elektrotechniky a komunikačních technologií.

[3] LITWIN, L.; PUGEL, M. *The principles of OFDM* [online]. [cit. 25.4.2007]. Dostupné z <http://rfdesign.com/images/archive/0101Puegel30.pdf>.

[4] WITTROCK, C. *The status of Digital Terrestrial Television* [online]. [cit. 25.4.2007]. Dostupné z <http://www.protelevision.com/pdf/dvbtvs8.pdf>.

[5] *Hierarchical encoding – an interesting specialty within DVB-T and DVB-H* [online]. [cit. 25.4.2007]. Dostupné z <http://www.protelevision.com/pdf/hierarch.pdf>.

[6] *SPRC121: TMS320C67x DSP Library Programmer's Reference Guide* [online]. 3-2006. Texas Instruments. [cit. 25.4.2007]. Dostupné z <http://focus.ti.com/docs/toolsw/folders/print/sprc121.html>.

[7] HARA, S.; PRASAD, R. *Multicarrier techniques fo 4G mobile communications*. 3rd ed. Artech House, 2003. 231 s. ISBN 1-58053-482-1.

[8] SCHULZE, H.; LUDERS, Ch. *Theory and applications of OFDM and CDMA*. Chichester: John Wiley & Sons Ltd., 2005. 403 s. ISBN-13 978-0-470-85069-5, ISBN-10 0-470-85069-5.

[9] BAHAI, A. R. S.; SALTZBERG, B. R.; ERGEN, M. *Multi-Carrier Digital Communication: Theory and Applications of OFDM*. 2nd ed. New York: Springer Science+Business Media, Inc., 2004. 407 s. ISBN 0-387-22575-7.

[10] SPETH, M.; FECHTEL, S.; FOCK, G.; MEYR, H. Optimum Receiver Design for Wireless Broad-Band Systems Using OFDM-Part I. In *IEEE Transactions on Communications*. 1999. s. 1668 - 1677. (IEEE; vol. 47, issue 11, 10.1109/26.803501).

[11] ZOU, H.; McNAIR, B.; DANESHRAD, B. An Integrated OFDM Receiver for High-Speed Mobile Data Communications. In *Global Telecommunications Conference*. 2001. s. 3090 – 3094. (GLOBECOM; vol. 5, 10.1109/GLOCOM.2001.965995).

[12] INDUSTRIAL TECHNOLOGY RESEARCH INSTITUTE. Time-frequency correlation-based synchronization for coherent OFDM receiver. Původce vynálezu: CHEN Ching-Yung, WANG Yi-Ting, HUNG Yung-Hua. United States of America. Patent Application Publication No. US 2006/0088133 A1. 27. 4. 2006 [online]. [cit. 25.4.2007]. Dostupné z <http://www.freepatentsonline.com/20060088133.html>.

[13] SPETH, M.; FECHTEL, S.; FOCK, G.; MEYR, H. Optimum Receiver Design for OFDM-Based Broadband Transmission-Part II: A Case Study. In *IEEE Transactions on Communications*. 2001. s. 571 - 578. (IEEE; vol. 49, issue 4, 10.1109/26.917759).

[14] FRESCURA, F.; PIELMEIER, S.; REALI, G.; BARUFFA, G.; CACOPARDI, S. DSP based OFDM demodulator and equalizer for professional DVB-T receivers. In *IEEE Transactions on Broadcasting*. 1999. s. 323 – 332. (vol. 45, issue 3, 10.1109/11.796275)

[15] MARŠÁLEK, R. *Teorie rádiové komunikace: Přednášková prezentace č. 4*. Brno: VUT, 2007. Fakulta elektrotechniky a komunikačních technologií.

[16] *SPRS088o: TMS320C6711, TMS320C6711B, TMS320C6711C floating-point digital signal processors* [online]. 11-2005. Texas Instruments. [cit. 12.5.2008]. Dostupné z <http://focus.ti.com/lit/ds/sprs088o/sprs088o.pdf>.

[17] *SPRU733a: TMS320C67x/C67x+ DSP CPU and Instruction Set Reference Guide* [online]. 11-2006. Texas Instruments. [cit. 12.5.2008]. Dostupné z <http://focus.ti.com/lit/ug/spru733a/spru733a.pdf>.

[18] *SPRU656a: TMS320C6000 DSP Cache User's Guide* [online]. 5-2003. Texas Instruments. [cit. 12.5.2008]. Dostupné z <http://focus.ti.com/lit/ug/spru656a/spru656a.pdf>.

[19] *SPRU234c: TMS320C6000 DSP Enhanced Direct Memory Access (EDMA) Controller Reference Guide* [online]. 11-2006. Texas Instruments. [cit. 12.5.2008]. Dostupné z <http://focus.ti.com/lit/ug/spru234c/spru234c.pdf>.

[20] *SPRU423f: TMS320 DSP/BIOS User's Guide* [online]. 11-2004. Texas Instruments. [cit. 12.5.2008]. Dostupné z <http://focus.ti.com/lit/ug/spru423f/spru423f.pdf>.

[21] *SPRU100a: TMS320C67x FastRTS Library Programmer's Reference* [online]. 10-2002. Texas Instruments. [cit. 12.5.2008]. Dostupné z <http://focus.ti.com/lit/ug/spru100a/spru100a.pdf>.

[22] *SPRU401j: TMS320C6000 Chip Support Library API Reference Guide* [online]. 8-2004. Texas Instruments. [cit. 12.5.2008]. Dostupné z <http://focus.ti.com/lit/ug/spru401j/spru401j.pdf>.

Seznam zkratek, symbolů a příloh

AWGN	average white gaussian noise
BER	bit error ratio
BCH	Bose, Ray-Chaudhuri, Hocquenghem
CACHE	rychlá statická vyrovnávací paměť
COFDM	coded orthogonal frequency division multiplexing
CP	continual pilots
CPU	central processing unit
CTF	channel transfer function
D/A	digital to analog conversion
DFT	discrete fourier transform
DSP	digital signal processor
DVB-H	digital video broadcasting-handheld
DVB-T	digital video broadcasting-terrestrial
EDMA	enhanced direct-memory-access
EMIF	external memory interface
FEC	forward error correction
FFT	fast Fourier transform
FIFO	first-in-first-out
HPI	host port interface
HWI	proces typu hardware interrupt
ICI	inter carrier interferences
IDFT	inverse discrete fourier transform
IDL	proces typu idle loop
IFFT	inverse fast Fourier transform
ISI	inter symbol interferences
JTAG	joint test action group
MFLOP	million floating-point operations per second
MPEG	moving picture experts group
MSB	most significant bit
OFDM	orthogonal frequency division multiplexing
PaRAM	parameter RAM pro konfiguraci EDMA
PRBS	pseudo random binary sequence
QAM	quadrature amplitude modulation
QPSK	quadrature phase shift keying
RAM	random access memory
ROM	read only memory
RTDX	real-time data exchange
SDRAM	synchronous dynamic random access memory
SFN	single frequency network
SNR	signal to noise ratio
SP	scattered pilots
SWI	proces typu software interrupt
TPS	transmission parameters signaling
TSK	proces typu task
VLIW	very long instruction word

A, B	označení výstupních větví vnitřního kodéru
b	pozice v konstelačním diagramu
B_{ch}	koherentní šířka pásma [Hz]
D	rozestup SP v normalizované frekvenční ose
δ	délka boků okna pro omezení spektra signálu [s]
Δ	délka ochranného intervalu [s]
Δf_{norm}	normalizovaný frekvenční ofset
$\overline{\Delta f_{norm}}$	průměrná hodnota normalizovaného frekvenčního ofsetu
Δn	hodnota diskrétního časového ofsetu
$\overline{\Delta n}$	průměrná hodnota diskrétního časového ofsetu
ε	nalezený celočíselný normalizovaný frekvenční ofset
f_{norm}	normalizovaný kmitočet
f_{vz}	vzorkovací frekvence OFDM signálu [Hz]
f_{vzmax}	maximální dosažitelná vzorkovací frekvence OFDM signálu [Hz]
g_k	znaménko pilotu na k-té nosné
G_1, G_2	generující polynomy vnitřního kodéru
$H(q)$	permutační funkce prokládání datových slov
$H_l(k)$	vzorky komplexní přenosové funkce kanálu
$H_l'(k)$	demodulované piloty proložené nulami
I	hloubka prokládání vnějšího prokladače
j	indexace větví vnějšího prokladače
k	index nosných
K	počet nosných
κ	testovaný normalizovaný celočíselný frekvenční ofset
$konst$	konstanta při vyhodnocování výsledků korelace
m	počet bitů na nosnou
M	podíl počtu bajtů a hloubky prokládání vnějšího prokladače
max	maximum modulů výsledků korelace
min	minimum modulů výsledků korelace
n	diskrétní časový index
N	šířka použité IFFT
N_G	počet vzorků odpovídající délce ochranného intervalu
N_S	počet vzorků odpovídající délce jednoho symbolu
n_p	index vrcholu $y(n)$
$p(k)$	impulsní charakteristika interpolačního filtru
P_A	průměr modulů výsledků korelace počítaný přes N_G hodnot
P_B	průměr modulů výsledků korelace počítaný přes N_S - N_G hodnot s posunem o N_G
R	kódový poměr vnitřního kodéru
$s(t)$	vstupní signál dekodéru
$s(n)$	vstupní diskrétní signál dekodéru
$s(n)_{kompenz}$	vstupní diskrétní signál dekodéru po vykompenzování frek. ofsetu
SP_{min}	první pozice SP v daném symbolu
SP_{max}	poslední pozice SP v daném symbolu
t	čas [s]
T	vzorkovací perioda OFDM signálu [s]
T'	vzorkovací perioda OFDM signálu po převzorkování [s]
T_{ct}	koherentní doba [s]
T_S	délka trvání jednoho OFDM symbolu [s]
T_U	délka trvání užitečné části symbolu [s]

v	počet větví vnitřního prokladače
w	pořadí bitů ve větvích vnitřního prokladače
$y(n)$	výstup korelátoru
$Y_{l,k}$	výstup FFT na k-té nosné pro l-tý OFDM symbol
Y_k	komplexní hodnoty nosných (uvažován aktuální symbol)
Y_k'	komplexní hodnoty nosných po ekvalizaci (uvažován aktuální symbol)
z	komplexní hodnota získaná z konstelačního diagramu

Příloha č. 1: Fotografie realizovaného systému

Příloha č. 1: Fotografie realizovaného systému