ARCHITEKTURA SYSTÉMU PRO DYNAMICKY REKONFIGUROVATELNÝ KOMUNIKAČNÍ TERMINÁL

Jan Kloub
Informatika a výpočetní technika, 2 ročník, distanční
Školitel: doc. Ing. Hana Kubátová, CSc.
Školitel specialista: Ing. Martin Daněk, PhD.
Ústav teorie informace a automatizace AV ČR, v.v.i.
Pod Vodárenskou věží 1143/4, 182 08 Praha8

kloub@utia.cas.cz

Abstrakt. Tento článek popisuje možnost implementace architektury výpočetního systému, který by umožňoval dynamicky vytvářet datové cesty s výpočetními uzly, a realizovat tak aplikace pro proudové zpracovávání dat. Příkladem aplikace pro proudové zpracování dat je implementace vysílacího a přijímacího řetězce komunikačního terminálu, na kterou je zaměřen následující text. Architektura je realizována pomocí obvodu FPGA, který umožňuje dynamicky měnit svojí funkci za běhu – pomocí tzv. dynamické rekonfigurace. Funkce obvodu může být měněna úplně nebo jen částečně, kdy je část obvodu statická a část dynamická. Rekonfigurace je řízena softwarově pomocí programu řídícího mikroprocesoru, který lze implementovat ve statické části obvodu FPGA („Soft“ procesor) rovněž lze využít jakýkoliv jiný procesor připojený k obvodu.

Klíčová slova. DSP, FPGA, dynamická rekonfigurace, částečná rekonfigurace, SDR.

1 Úvod

V dnešní době jsou kladeny vysoké nároky na výkon, spotřebu, univerzálnost a cenu zařízení. Jedním ze způsobů jak využít efektivně zdroje, může být vícenásobné využití zdrojů číslicových obvodů. Vícenásobné využití zdrojů číslicových obvodů může znamenat například změnu programu u procesoru (CPU) nebo změnu konfigurace a propojení programovatelného hradlového pole (FPGA). Obvody FPGA umožňují implementovat paralelně vykonávané operace v závislosti na vnitřní konfiguraci. U některých obvodů FPGA je možné měnit konfiguraci během běhu aplikace a lze tak několikanásobně využít stejné zdroje obvodu. Změnu konfigurace za běhu aplikace nazýváme dynamickou rekonfigurací. Dynamická rekonfigurace je podpořena například v obvodech firmy Xilinx [14].

Tento text se zabývá možností implementace takového systému, který by mohl proudově zpracovávat data pomocí obecného procesoru a pomocí vytváření datových cest a výpočetních uzlů v hradlovém programovatelném poli. Architekturu sytému pro proudové zpracovávání dat lze například využít k implementaci řetězce komunikačního terminálu [11].

Modulace a jiné operace nad vysílanými nebo přijímanými daty lze provádět číslicově [10], a proto lze velkou část řetězce komunikačního terminálu implementovat v obvodu FPGA. Pro vlastní vysílání je třeba jen jednoduchá analogová část s A/D a D/A převodníky.

S využitím obvodu FPGA lze jednotlivé funkční části terminálu a jejich propojení za chodu měnit podle požadavků na parametry přenosového kanálu.

Cílem je navrhnout takovou architekturu, která by využívala programovatelné hradlové pole jako obecný prostředek pro řešení momentálních požadavků systému na proudové zpracování dat s využitím dynamické rekonfigurace. Sytém by měl být schopen udržovat informaci o obsazenosti prostředků a případně relokovat výpočetní jádra v rámci hradlového pole.

Struktura tohoto textu je následující: v kapitole 2 jsou nastíněny možnosti částečné dynamické rekonfigurace, v kapitole 3 je popsána možná realizace architektury pro proudové zpracování dat, kapitola 4 popisuje základní blokové schéma bezdrátového komunikačního kanálu, kapitola 5 popisuje implementaci konvolučního dekodéru vhodnou pro navrhovanou architekturu a v kapitole 6 jsou shrnuty výsledky práce a nastínění budoucí práce.

2 Dynamické rekonfigurace na obvodech FPGA

Funkce obvodu FPGA je dána obsahem jeho konfigurační paměti. Konfigurační paměť obsahuje informace o propojení a nastavení vnitřní struktury obvodu. Počáteční konfigurace obvodu je obvykle nahrána při zapnutí obvodu z externí paměti. Konfigurace obvodu lze provádět i za běhu, a mluvíme tak o dynamické rekonfiguraci. Rekonfigurace může být prováděna buď přes celý obvod FPGA nebo jen přes jeho část, a pak hovoříme o částečné dynamické rekonfiguraci. Část, někdy označována také jako oblast, která se nemění během běhu aplikace se nazývá statická část a část, která je měněna se nazývá dynamická. Ve statické části bývá implementována řídící logika, jejíž součástí může být i procesor (tzv. „Soft“ procesor). Do dynamické části jsou umisťovány moduly implementující konkrétní funkce.

Rekonfigurace je prováděna pomocí konfiguračního rozhraní obvodu FPGA, které je přístupné pomocí vývodů obvodu (JTAG) a nebo obvod obsahuje vnitřní konfigurační rozhraní (ICAP). Vnitřní konfigurační rozhraní je vhodné použít, pokud konfiguraci provádí řídící logika ve statické části obvodu. Konfigurace obvodu je dána obsahem jeho konfigurační paměti. Doba rekonfigurace je dána velikostí konfigurační bitové posloupnosti (bitstream) a přenosovou rychlostí mezi paměťovým systémem architektury, kde je bitstream uložen, a konfiguračním rozhraním. Z omezené přenosové rychlosti konfiguračního rozhraní vyplývá, že při rekonfiguraci modulů dochází k latencím, a proto není vhodné rekonfigurovat často, pokud nelze konfigurační proces překrývat využíváním jiné funkční části obvodu.

S použitím (částečné) dynamické rekonfigurace lze na obvodech FPGA realizovat takové funkce, které lze v průběhu aplikace měnit. Možnost měnit funkci obvodu za běhu aplikace přináší efektivnější využití zdrojů obvodu, protože pro různé funkce jsou použity stejné hardwarové zdroje.

V tomto textu je popisována pouze implementace dynamické rekonfigurace na obvodech firmy Xilinx [14].

2.1 Lokalita modulů pro dynamickou rekonfiguraci

V rámci částečné dynamické rekonfigurace se nabízí myšlenka mít několik dynamických částí obvodu FPGA určených pro moduly implementovaných funkcí a dle potřeby umisťovat moduly do aktuální volné části. Každá funkce je reprezentována jedním konfiguračním bitstreamem. Konfigurační bitstreamy pro obvody Virtex firmy Xilinx obsahují příkazy pro vnitřní automat konfiguračního rozhraní, jako je například adresa konfigurovatelné oblasti, informace o velikosti konfiguračních dat a zabezpečovací cyklický kód. Způsob konfigurace a reprezentace dat v bitstreamu je uvedena v [1] a [2].

S růstem počtu funkcí a s počtem oblastí, kam je třeba jednu funkci nahrát, roste i počet bitstreamů uložených v paměti, protože bitstream pro jednu funkci nelze kvůli zabezpečení nahrát na jiné místo v obvodu, než pro které byl určen. Pokud chceme bitstream nahrát do jiné oblasti obvodu (relokovat), musí být příslušným způsobem předem pozměněn. Dalším důvodem, proč v některých případech nelze relokovat bitstream, je také fakt, že celý obvod FPGA nemusí obsahovat homogenní regulární

strukturu (výskyt blokových pamětí, jádra procesorů atp.). Pokud chceme moduly v rámci FPGA přesouvat je nutné v konfiguračním bitstreamu pozměnit cílovou adresu konfiguračních dat tak, aby byla respektována struktura obvodu, a přepočítat zabezpečovací kód. Problematika relokace bitstreamu je popsána v [3], [4] a [5].

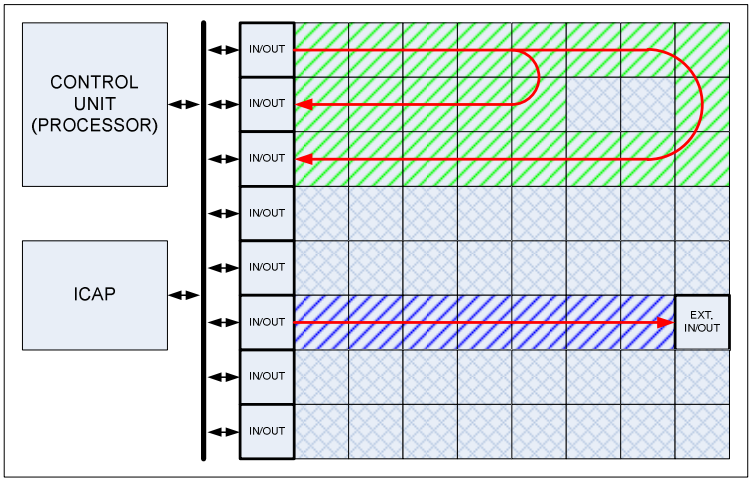
3 Architektura systému pro proudové zpracování dat

Cílem je navrhnout takovou architekturu, která bude schopna proudově zpracovávat příchozí data, za běhu měnit výpočetní cesty a tvořit tak dynamicky rekonfigurovatelný systém.

Některé rekonfigurovatelné systémy (jako například [6] a [7]) používají mřížku výpočetních jader a pro propojení jednotlivých jader používají konfigurovatelnou propojovací síť s přepínači (NoC – Network on Chip). Rekonfigurovatelné systémy lze rozdělit podle granularity rekonfigurovatelných výpočetních jader na „hrubozrnné“ (Coarse-grain, [6], [7]) nebo „jemnozrnné“ (Fine-grain [8]). Systémy s „hrubozrnnou“ architekturou obsahují komplexní výpočetní jádra jejichž rekonfigurace spočívá ve změně řídícího slova jádra. Systémy s „jemnozrnnou“ architekturou rekonfigurují na úrovni jednotlivých hradel a klopných obvodů, což je příhodné právě pro technologii FPGA obvodů. Obvody FPGA přinášejí výhodu při paralelním zpracování bitově orientovaných operací, jako například při digitálním zpracování signálů v komunikačním kanálu. Navrhovaný systém lze zařadit mezi systémy s „jemnozrnnou“ architekturou rekonfigurovatelných výpočetních jader.

Hlavním předpokladem pro návrh nového systému je datová nezávislost jednotlivých operací, a to že lze jednotlivé operace nad daty reprezentovat grafem přenosů dat mezi výpočetními uzly (dataflow graf - DFG). Snahou je mapovat DFG do dynamické oblasti obvodu FPGA. Možná architektura systému je znázorněna na obrázku 1, který znázorňuje dlaždicové rozdělení dynamické části obvodu, řídící logiku celé aplikace (např. procesor) a řídící logiku pro konfigurační rozhraní. Vstupní a výstupní body grafu budou připojeny ke statické části obvodu FPGA, a to buď zpět k rozhraní řídící logiky nebo k rozhraní pro propojení s externí obvody (například rozhraní A/D a D/A převodníků). Výpočetní uzly jsou propojeny přímo mezi sebou a jejich vzájemné propojení je omezeno propojovacími možnostmi konkrétního obvodu FPGA.

Dlaždicová topologie může být nahrazena topologií popsanou v [9], která simuluje propojení pravidelných šestiúhelníků, kde s každou dlaždicí může sousedit šest dlaždic, a tak umožnit lepší větvení DFG v dvojrozměrném prostoru.



Obrázek 1: Architektura pro proudové zpracování dat pomocí programovatelného hradlového pole

4 Dynamicky rekonfigurovatelný komunikační terminál

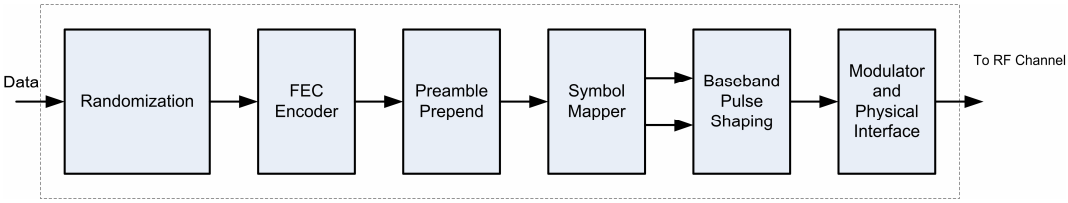
Snahou mnoha aplikací je vytvořit takový komunikační systém, který není závislý na zvoleném přenosovém režimu. Jedním ze způsobů jak vytvořit „univerzální“ komunikační terminál je využívat takové zdroje prostředků, které lze využít pro všechny zvolené režimy přenosu. Nabízí se tedy možnost dynamické rekonfigurace pro využití stejných zdrojů číslicových obvodů pro jiný účel v průběhu aplikace. Podobný koncept je použit v projektu softwarově definovaného rádia (SDR) [10], který pro zpracování signálů přenosového kanálů používá obecné procesory nebo vestavěné systémy a operace nad daty jsou řešeny programově.
Výkon současných procesorů nedovoluje v některých případech zpracovávat data v reálném čase, a proto se pro lepší výsledky implementují k procesorům koprocesory.

Navrhovaný systém může být řízen programově, pokud bude použit k řízení procesor. Řídicím procesorem může být například procesor MicroBlaze, který lze implementovat ve statické části obvodu FPGA. Pokud jsou kladeny vysoké nároky na výkon nebo na specifické vlastnosti samotného procesoru v dané aplikaci nebo je třeba pro dynamickou část obvodu využít téměř celý obvod FPGA, pak lze připojit vhodný procesor k obvodu FPGA. Například vzhledem k charakteru implementace komunikačního kanálu je vhodné použít signálový procesor (DSP), který může předzpracovávat data pro obvod FPGA.

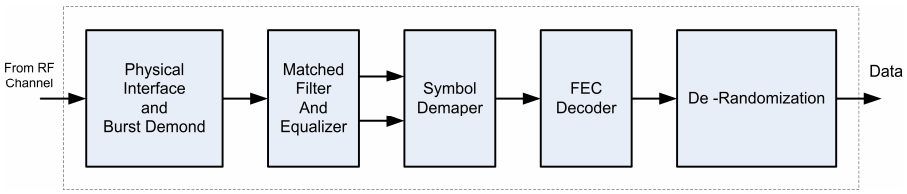
Program řídícího procesoru bude volat funkce pro rekonfiguraci jednotlivých modulů nebo změnu jejich propojení a posílat data ke zpracování. Přímé propojení jednotlivých operací v DFG umožní řadit výpočetní bloky za sebe a není tak třeba data po každé dílčí operaci předávat po sběrnici procesoru.

4.1 Architektura komunikačního řetězce

Základní architektura fyzické vrstvy bezdrátového komunikačního kanálu podle standardu 802.16 [11] pro vysílání a příjem dat je znázorněna na obrázku 2 a 3. Jednotlivé bloky a jejich funkce je popsána v [11]. Blokový diagram pro jiné komunikační médium nebo přenosový režim se nemusí příliš lišit, ale liší se funkce jednotlivých bloků. Například blok pro zabezpečení dat opravnými kódy (FEC – z angl. Forvard Error Correction) může využívat Reed-Solomonův (RS) kód, RS v kombinaci s konvolučním kódem nebo turbokódy. Parametry zabezpečovacích kódů jsou také parametrizovány podle typu přenosu. Implementace takovéhoto univerzálního bloku by byla velmi hardwarově náročná. Architekturu komunikačního terminálu lze reprezentovat dvěma grafy toku dat, které odpovídají blokovým schématům na obrázcích 2 a 3.



Obrázek 2: Blokový diagram fyzické vrstvy komunikačního kanálu pro vysílání dat (převzato z [11])

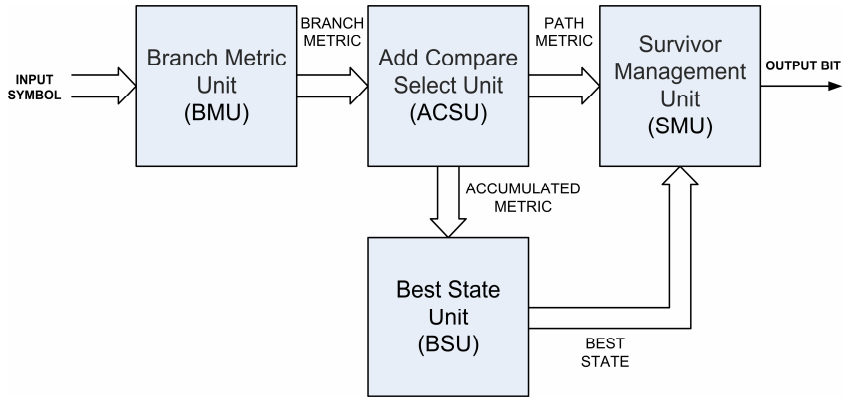


Obrázek 3: Blokový diagram fyzické vrstvy komunikačního kanálu pro příjem dat (převzato z [11])

5 Implementace modulu konvolučního dekodéru a jeho dekompozice

Jednou ze součástí funkčních bloků přijímače bývá dekodér konvolučního kódu, realizovaný nejčastěji Viterbiho dekodérem [12] (viz podkapitola 4.1 - FEC dekodér). Vlastní strukturu Viterbi dekodéru lze opět dekomponovat na dílčí funkční bloky a vyjádřit jí pomocí grafu toku dat, jak je znázorněno na obrázku 4. Viterbi dekodér přijímá data kódovaná konvolučním kódem [13] a vytváří rozhodovací graf („trellis“) pro určení nejpravděpodobnější přijaté bitové posloupnosti. Jednotka BMU určuje váhy jednotlivých hran grafu, ACSU vybírá dílčí nejlepší cesty grafem, BSU hledá koncový uzel grafu s nejlépe ohodnocenou cestou a SMU prochází graf od tohoto uzlu a dekóduje přijatou bitovou posloupnost. Parametry dekodéru jsou dány funkcionalitou jeho bloků, a proto je vhodné s ohledem na využití hardwarových zdrojů, bloky rekonfigurovat, a tak měnit parametry dekodéru. Jednotlivé bloky dekodéru můžeme považovat za součásti terminálu a mapovat je na jednotlivé dlaždice architektury.

Součástí práce na návrhu architektury dynamicky rekonfigurovatelného systému je funkční implementace Viterbi dekodéru v jazyce Handel-C, kterou lze popisovaným způsobem dekomponovat na jednotlivé funkční bloky. Současná implementace Viterbi dekodéru je plně parametrizovatelná zvoleným konvolučním kódem a stupněm paralelismu, a tedy i množstvím použitých hardwarových prostředků při zpracovávání příchozí bitové posloupnosti. Jedná se o takzvanou hybridní implementaci, která vzhledem k náročnosti na hardwarové zdroje obvodu FPGA kombinuje paralelní a sekvenční zpracování příchozích symbolů. Kombinace paralelního a sekvenčního zpracování umožňuje balancovat mezi výkonem dekodéru a množstvím použitých hardwarových prostředků. Výkon implementace dekodéru na obvodu FPGA Virtex-4 (xc4vsx35-10-ff668) byl porovnáván s implementací na procesoru firmy Texas Instruments TMS320C6416 (1GHz), který obsahuje koprocesor podporující Viterbi algoritmus [13]. Implementace na obvodu FPGA může pomocí vhodné volby množství použitých hardwarových zdrojů dosáhnou srovnatelných parametrů jako implementace pomocí signálového procesoru.



Obrázek 4: Blokové schéma Viterbi dekodéru

6 Závěr

V tomto textu byla popsána architektura, která bude předmětem další práce a vývoje. V současné době jsou dílčími výsledky mé práce seznámení s možnostmi částečné dynamické rekonfigurace a implementace Viterbi dekodéru. Dekodér je implementován bez datových závislostí mezi jeho funkčními bloky, a proto je velmi vhodný jako příklad pro referenční implementaci pomocí navrhované architektury.

Součástí mé budoucí práce bude studium možností, jak efektivně implementovat dynamicky rekonfigurovatelnou mřížku na stávajících obvodech FPGA s minimální režií zdroje obvodů,

implementovat mřížku, navrhnout metodu pro propojování dílčích funkčních modulů a implementovat pomocí navrhované architektury komunikační řetězec, který bude schopen pomocí částečné dynamické rekonfigurace měnit své přenosové vlastnosti.

Poděkování

Tato práce vznikla za podpory projektů číslo 1ET100750408, 1ET300750402 a 1ET400750408 Grantové agentury AV ČR.

Reference

[1] Xilinx Application Notes 151: "Virtex Series Configuration Architecture User Guide", 2004, URL: http://www.xilinx.com.

[2] Xilinx Application Notes 138: "Virtex FPGA Series Configuration and Readback", 2005, URL: http://www.xilinx.com.

[3] Kalte H., Lee G., Porrmann M., Ruckert U., "REPLICA: A Bitstream Manipulation Filter for Module Relocation in Partial Reconfigurable Systems", Parallel and Distributed Processing Symposium, 2005, Proceedings. 19th IEEE International 04-08 April 2005 Page(s):151b - 151b

[4] Krasteva Y.E., Jimeno A.B., de la Torre E., Riesgo T., "Straight method for reallocation of complex cores by dynamic reconfiguration in Virtex II FPGAs", Rapid System Prototyping, 2005. (RSP 2005). The 16th IEEE International Workshop on 8-10 June 2005 Page(s):77 – 83

[5] Corbetta S.Ferrandi., Morandi M., Novati M., Santambrogio M.D., Sciuto D., "Two Novel Approaches to Online Partial Bitstream Relocation in a Dynamically Reconfigurable System", VLSI, 2007. ISVLSI '07. IEEE Computer Society Annual Symposium on 9-11 March 2007 Page(s):457 – 458

[6] Singh H., Ming-Hau Lee, Guangming Lu, Kurdahi F.J., Bagherzadeh N., Chaves Filho, E.M., "MorphoSys: An Integrated Reconfigurable System for Data-Parallel Computation-Intensive Applications", 2000, URL: http://www.eng.uci.edu/morphosys/docs/ieee.pdf

[7] Mirsky E., DeHon A., "MATRIX: a reconfigurable computing architecture with configurable instruction distribution and deployable resources**",** FPGAs for Custom Computing Machines, 1996. Proceedings. IEEE Symposium on 17-19 April 1996 Page(s):157 – 166

[8] Tau E., Eslick I., Chen D., Brown J., DeHon A., "A First Generation DPGA Implementation", Proceedings of the Third Canadian Workshop on Field-Programmable Devices, pages 138-143, May 1995.

[9] Thomas A., "Design of a dynamic reconfigurable multi-grained hardware architecture with adaptive runtime routing", Field Programmable Logic and Applications, 2005. International Conference on 24-26 Aug. 2005 Page(s):745 – 746

[10] Huie J., D'Antonio P., Pelt R., Jentz B., „Synthesiszing FPGA Cores for Software-Defined Radio", URL: http://www.altera.com.cn/literature/cp/fpga-cores-for-sdr.pdf

[11] IEEE Std. 802.16-2004, IEEE Standard for Local and metropolitan area network – Part 16: Air Interface for Fixed Broadband Wireless Access Systems

[12] Rizwan Rasheed. „Reconfigurable Viterbi Decoder for Mobile Platform", URL: http://www.ctr.kcl.ac.uk/mwcn2005/Paper/C200540.pdf

[13] Kloub J., Heřmánek A., „Akcelerátor pro dekódování konvolučního a Reed-Solomonova zabezpečovacího kódu", 2007, Technical Computing Prague 2007

[14] Xilinx Application Notes 290: " Difference-Based Partial Reconfiguration", 2007, URL: http://www.xilinx.com.