Využití funkční verifikace pro ověřování metodik pro zajištění odolnosti proti poruchám

Jakub Podivínský
Informatika a výpočetní technika, druhý ročník, prezenční studium
Školitel: Zdeněk Kotásek

Fakulta informačních technologií, Vysoké učení technické v Brně
Božetěchova 2, 612 66 Brno
ipodivinsky@fit.vutbr.cz

Abstrakt. Náplní tohoto článku je představení práce zabývající se aplikací funkční verifikace pro testování metodik pro zajištění odolnosti proti poruchám v systémech založených na FPGA. Je zde představena problematika obvodů FPGA, vlivu poruch a možnosti jejich eliminace. Na základě těchto poznatků jsou formulovány cíle práce a návrh jejich dosažení. Uveden je také přehled dosažených výsledků při řešení této práce. Mezi ně patří experimentální elektromechanický systém, první verze platformy pro testování metodik pro zajištění odolnosti proti poruchám a zejména verifikační prostředí pro procesor běžící na FPGA, který bude tvořit jádro dalšího experimentálního systému. Dosavadní výzkumná práce je shrnuta v časopise Microprocessors and Microsystems [1].

1 Úvod a motivace

Číslicové systémy hrají stále větší roli v našem každodenním životě, setkáváme se s nimi v nejrůznějších aplikacích. Jsou hojně využívány v průmyslové výrobě, používají se jako řídicí systémy v dopravních prostředcích, medicíně, telekomunikacích a podobně. Současný trend je přesouvání stále větší zodpovědnosti právě na číslicové řídicí systémy, což obvykle vede ke snížení hmotnosti mechanické části a tím i snížení provozních nákladů, například v letectví nebo automobilismu [2]. Důsledkem je, že používané číslicové systémy jsou stále komplexnější, což vede k neustálému růstu míry jejich integrace. Tento jev má za následek větší náchylnost takových systémů k poruchám, a to jak k poruchám vzniklým při návrhu a implementaci systému, tak k poruchám vzniklým za provozu systému. Výskyt takové poruchy může mít nedozírné následky, a to nejen ve formě finančních ztrát, ale může dojít i k ohrožení lidských životů (např. porucha v řídicím systému letadla). Pro bezpečnostně kritické aplikace je tedy velmi důležité hledat cesty, jak zajistit, aby vznik poruch nijak neohrozil provoz těchto systémů. Tato práce se zabývá tvorbou platformy pro ověřování metodik pro zajištění odolnosti proti poruchám (*FT - Fault Tolerance*) v systémech založených na FPGA. Jako základní vyhodnocovací mechanismus poslouží funkční verifikace.

2 Současný stav poznání

Před formulací cílů práce je třeba krátce uvést do problematiky FPGA, odolnosti proti poruchám a funkční verifikace.

2.1 Obvody FPGA a výskyt poruch

Programovatelná hradlová pole (*FPGA*) jsou obvody, které je možné programovat jak před použitím, tak za běhu aplikace bez přerušení činnosti ostatních částí obvodu pomocí *částečné dynamické rekonfiguraci* (PDR – Partial Dynamic Reconfiguration) [3]. FPGA jsou stále populárnější a nacházejí uplatnění v řadě aplikací, především

díky zmíněné programovatelnosti, snadnému návrhu, flexibilitě, snižující se spotřebě ale také klesajícím cenám. FPGA se skládá z matice konfigurovatelných logických bloků (*CLB*), které jsou propojeny pomocí programovatelné propojovací sítě. Mimo CLB obsahují i řadu dalších prvků. Konfigurace jednotlivých bloků a propojovací sítě je uložena v paměti SRAM ve formě bitové posloupnosti (tzv. *bitstream*). V současné době jsou nejpoužívanější FPGA s konfigurací uloženou v paměti SRAM.

Poruchy v FPGA typicky vznikají vlivem vysoce energetických částic [4]. Dopad takové částice na FPGA může způsobit nežádoucí zákmit na přenášeném signálu, což je efekt označovaný jako *Single Event Transient* (SET). Při zásahu paměťové buňky může dojít ke snížení napětí, což vede ke změně uložené hodnoty. Tento efekt se nazývá *Single Even Upset* (SEU) a je to nejčastější porucha postihující FPGA obvody.

2.2 Implementace a ověřování odolnosti proti poruchám

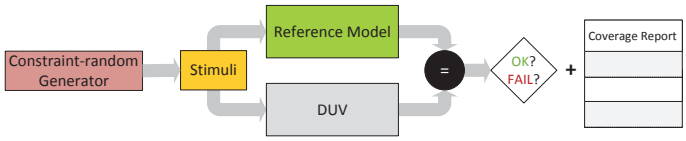
Při zajišťování odolnosti proti poruchám můžeme použít vhodnou modifikaci základních typů *redundance* (*hardwarová, časová, informační a programová redundance*). Základní technikou pro detekci a maskování poruchy je technika TMR (*Triple Modular Redundancy*), která je schopna díky ztrojení funkčních bloků maskovat jednu poruchu. Následná částečná dynamická rekonfigurace umožňuje opravit poruchou napadenou část konfigurační paměti FPGA a obnovit bezporuchový stav.

Čekat na přirozený výskyt poruch (SEU) je velmi neefektivní. Omezující jsou zde parametry MTTF (*Mean Time To Failure*) a MTBF (*Mean Time Between Failures*), které se mohou pohybovat i v řádech několika let, proto je nutné tyto poruchy vhodným způsobem simulovat. Simulační metoda pro emulaci vlivu SEU poruch v konfigurační paměti FPGA je představena v [5]. Autoři kombinují simulaci a topologickou analýzu systému, který je implementován v FPGA. Nástroj pro injekci poruch do FPGA je náplní článku [6]. Podporuje různé modely poruch použitelné v FPGA, které jsou implementovány ve VHDL, ale je třeba modifikovat původní návrh a doplnit další hradla a spoje pro injekci poruch. Techniky založené na injekci poruch do reálného FPGA bez nutnosti změny původního systému jsou představeny v [7]. Jsou založeny na PDR, která umožňuje přečíst část aktuální konfigurační paměti, invertovat zvolený bit a následně zapsat tuto část zpět do paměti. Výzkumem v oblasti injekce poruch se také zabývají někteří členové týmu doc. Kotáska. Ing. Jan Kaštil a další prezentují v [8] externí injektor poruch do FPGA. Tento injektor je založen na generování poruch mimo FPGA (generování probíhá v PC), tedy není omezen na konkrétní desku s FPGA čipem.

2.3 Funkční verifikace číslicových systémů

Funkční verifikace ověřuje, zda systém odpovídá specifikaci monitorováním jeho vstupů a výstupů v simulačním prostředí (např. *ModelSim*). Jedná se o rozšířenou sofistikovanější verzi běžně používaných *testbench*. Pro usnadnění tvorby verifikačních prostředí existuje standardizovaný jazyk *SystemVerilog* [9], metodika UVM (*Universal Verification Methodology*) [10] a knihovny metodiky UVM.

Verifikovaný systém je na Obrázku 1 označen jako DUV (*Device Under Verification*), výstupy tohoto systému jsou porovnávány s referenčním modelem (*Reference Model*). Pokud je zjištěna neshoda mezi výstupy DUV a referenčního modelu, znamená to, že jejich funkce nejsou ekvivalentní. Výstupem funkční verifikace je také zpráva o pokrytí klíčových funkcí verifikovaného systému (*Coverage Report*), která nám říká, jak důkladně byly ověřeny jednotlivé specifikované funkce. Pro získání vysokého pokrytí je třeba zajistit přísun dostatečného počtu vhodných vstupů (*Stimuli*). To je úkolem generátoru, který tyto vstupy generuje (*Constraint-random Generator*).



Obrázek 1: Základní princip funkční verifikace.

3 Cíle disertační práce a návrh zvoleného řešení

Ze studia aktuálního stavu poznání v oblasti FT systémů založených na FPGA vyplynula potřeba nalézt odpověď na několik otázek: *Jaké budou výsledky FT metodik na reálných systémech? Lze spoléhat na to, že ne všechny poruchy v elektronické části systému se projeví na chování řízené mechanické aplikace? Lze využít funkční verifikaci pro ověřování vlivu poruch na FT systémy?* Na základě těchto otázek byly formulovány cíle disertační práce:

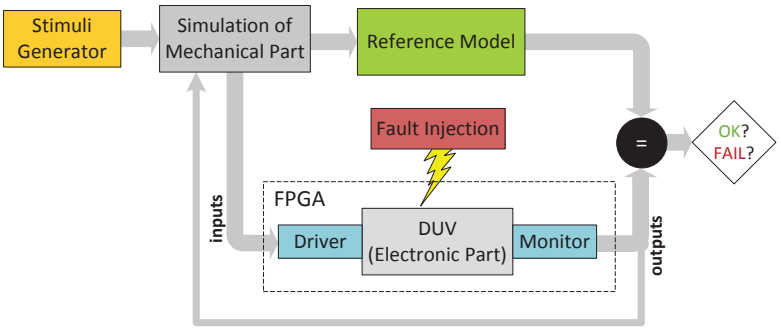
1. Navrhnout a vytvořit platformu, která bude založená na technologii FPGA a bude sloužit k testování FT metodik a k sledování vlivu poruch nejen na výstup elektronické části, ale také na řízenou mechanickou aplikaci. Bude využita technika funkční verifikace a injektor poruch vyvinutý týmem doc. Kotáska. Jádrem platformy bude experimentální elektromechanická aplikace.
2. V návaznosti na vytvořenou testovací platformu navrhnout proces ověřování FT metodik s přihlédnutím ke specifikům elektromechanických systémů. Tyto postupy budou využívat navrženou a vytvořenou testovací platformu. Proces bude navržen na základě experimentování s vytvořenou platformou. Součástí bude popis činností před zahájením procesu ověřování. Proces bude ověřen a demonstrován na dalším experimentálním systému. Tím dojde k zobecnění získaných výsledků.

3.1 Platforma pro ověřování FT metodik a proces ověřování FT metodik

Platforma bude koncipována jako rozšířené prostředí pro funkční verifikaci (Obrázek 2) a umožní ověřovat FT na experimentálních elektromechanických systémech. V uvedeném schématu lze nalézt jak elektronickou část (*Electronic Part*), tak mechanickou část (*Mechanical Part)* experimentálního systému. Elektronická část zde reprezentuje verifikovaný obvod (DUV), rozdíl oproti klasické verifikaci je umístění DUV na FPGA, což s sebou nese potřebu komunikace mezi FPGA a PC. Součástí každého verifikačního prostředí je referenční model (*Reference Model*), jehož výstup je porovnáván s výstupem DUV, které bude, díky použití injektoru poruch (*Fault Injector*), vystaveno působení poruch. Jelikož se zaměřujeme na ověřování metodik cílených na FPGA, poruchy budou injektovány pouze do FPGA (typicky SEU poruchy) a nikoliv do ostatních elektronických prvků (čidla atd.). Samozřejmě bude třeba vygenerovat takové stimuly, které povedou k dostatečnému funkčnímu pokrytí, což zajistí generátor stimulů (*Stimuli Generator*). V případě elektromechanické aplikace se jedná o různé konfigurace experimentální elektromechanické aplikace (různé verifikační scénáře). K vyhodnocení míry funkčního pokrytí lze s výhodou použít nástroje pro funkční verifikaci.

Postup ověřování FT metodik je rozdělen na tři fáze. V *první fázi* je potřeba vytvořit verifikační prostředí, včetně referenčního modelu, pro elektronickou řídicí jednotku. Je třeba vzít v úvahu mechanickou část tak, aby byla součástí verifikačního prostředí. V první fázi se nijak nevyužívá FPGA. Odhalují se chyby v implementaci tak, aby nebyly později zaměněny s projevy injektovaných poruch. Výsledkem této fáze bude sdělení, zda elektronická řídicí jednotka správně zpracuje vygenerované verifikační scénáře a také sada použitých scénářů, pro které je zaručené správné chování. *Druhá fáze* již využívá verifikaci na FPGA, což umožňuje použít injektor poruch. Vstupem

jsou ověřené verifikační scénáře, každý je několikrát opakován, při každém opakování je injektována jiná porucha či sada poruch a jsou sledovány jejich projevy. Výstupem je seznam poruch, které v kombinaci s daným verifikačním scénářem způsobily nesprávný výstup elektronického řadiče. Tyto poruchy budou detailněji analyzovány v *třetí fázi*. Každá injektovaná porucha tak bude zařazena do jedné z těchto kategorií: (1) Výstup z DUV a referenčního modelu se shoduje, porucha se neprojevila. (2) Výstupy se neshodují, navzdory tomu mechanická část splnila svůj úkol. (3) Výstupy se neshodují a zároveň došlo k nesplnění cíle mechanické část. Poruchy s vlivem na mechanickou část budou detailněji analyzovány. Důležitou fází při návrhu procesu ověřování bude navržení vhodných kombinací poruch se zaměřením na ověřovanou FT metodiku. Například při ověřování TMR bude zajímavé chování při výskytu stejné poruchy ve dvou kopiích stejného funkčního bloku.



Obrázek 2: Využití funkční verifikace pro testování odolnosti proti poruchám.

4 První verze testovací platformy

Robot pro hledání cesty v bludišti byl zvolen jako *první experimentální elektromechanický systém*. Elektronická část je zde reprezentována řídicí jednotkou robota implementovanou v FPGA, zatímco mechanickou část reprezentuje robot v bludišti. Není použit reálný robot, ale robot a jeho prostředí jsou pouze simulovány pomocí volně dostupného simulačního prostředí *Player/Stage* [11], což umožňuje velmi snadno měnit verifikační scénáře, tedy počáteční a cílovou pozici a bludiště, které robot prochází. Ověřované FT metodiky budou aplikovány na elektroniku v FPGA, tedy na řídicí jednotku robota. Tato řídicí jednotka je navržena tak, aby reprezentovala komplexní systém obsahující nejrůznější aspekty návrhu číslicových systémů (sekvenční a kombinační obvody, konečné automaty, paměti a sběrnice), což umožní testovat různě zaměřené metodiky.

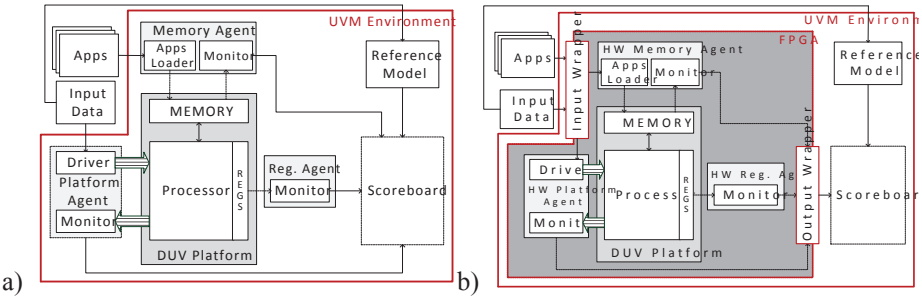
Byla implementována *první verze platformy*, prozatím bez aplikace principu funkční verifikace. Experimentálním systémem v první verzi platformy se stal zmíněný robot pro hledání cesty v bludišti. Platformu tvoří dva základní bloky - počítač a FPGA deska. Tyto bloky jsou spolu propojeny pomocí dvou komunikačních kanálů, první je rozhraní JTAG využívané injektorem poruch, druhý komunikační kanál slouží pro přenos dat mezi simulačním prostředím robota a jeho řídicí jednotkou. První verze platformy je rozdělena na 3 základní funkční části: (1) vývojová deska ML506 s FPGA Virtex 5, kde je implementována řídicí jednotka, (2) simulační prostředí *Player/Stage* [11] pro simulaci robota a kontrolu reakcí mechanické části na pokyny řídicí jednotky běžící na PC a (3)injektor poruch, který také běží na PC, umožňuje vkládat poruchy do řídicí jednotky robota [8].

5 Verifikační prostředí pro procesor běžící na FPGA

V další práci bych se chtěl zaměřit také na vytvoření jiného experimentálního elektromechanického systému. V současné době jsou k řízení nejrůznějších systémů používány procesory, můžeme se s nimi setkat v automobilech, letadlech a podobně. Chtěl bych se tedy zaměřit i na elektromechanický systém, kde bude elektronická část tvořena procesorem implementovaným na FPGA. To umožní ověřovat FT metodiky zaměřené nejen na číslicové systémy, ale také tzv. *Software Fault Tolerant* techniky. Rozhodl jsem se použít Codix RISC procesor [12], který je možné vygenerovat pomocí nástroje Codasip Studio (FIT vlastní akademickou licenci). Jedná se o 32-bitový procesor typu RISC (*Reduced Instruction Set Computer*) se 7 stupni zřetězeného zpracování, 32 volně použitelnými registry, 512 kB paměti a instrukční sada obsahuje 59 instrukcí.

S odkazem na *první fázi* procesu ověřování FT metodik je nutné nejprve vytvořit *prostředí pro funkční verifikaci pro procesor* a spustit verifikaci bez injekce poruch. Verifikační prostředí podle standardu UVM ukazuje Obrázek 3a, toto verifikační prostředí je implementováno v jazyce SystemVerilog a lze jej automaticky generovat pomocí nástrojů Codasip Studia.

Druhá fáze v procesu ověřování kvality metodik pro zajištění odolnosti proti poruchám je funkční verifikace systému implementovaného v FPGA, v tomto případě se jedná o procesor běžící na FPGA. V této fázi se také dostává ke slovu injekce poruch do FPGA. Pro tyto účely je třeba upravit původní verifikační prostředí tak, aby procesor běžel na FPGA, takové verifikační prostředí ukazuje Obrázek 3b. Je třeba poznamenat, že téměř všechny UVM komponenty byly přesunuty do FPGA, mimo referenční model (*Reference Model*) a porovnání výstupů (*Scoreboard*). UVM agenti a jejich vnitřní komponenty jsou nahrazeny HW agenty v FPGA. Komunikace mezi softwarovou a hardwarovou částí verifikačního prostředí je zajištěna použitím proprietárního rozhraní. Výstupy referenčního modelu jsou porovnávány s výstupy DUV, které jsou získány z HW prostřednictvím komponenty *Output Wrapper*. Porovnává se obsah paměti a registrového pole po provedení každého programu a také data na výstupním portu procesoru. Komponenta *Input Wrapper* slouží k odesílání programů, které vykonává procesor, a vstupních dat.



Obrázek 3: Verifikační prostředí pro procesor a) čistě SW verze, b) rozšíření o FPGA.

Při experimentování s akcelerovaným verifikačním prostředím bylo zjištěno, že verifikační prostředí akcelerované pomocí FPGA potřebuje pro splnění stejného úkolu kratší čas. Bylo dosaženo zrychlení přibližně 1,7x, což je horší výsledek, než bylo očekáváno. Je to způsobeno zejména komplexním referenčním modelem, který běží pomaleji, než DUV na FPGA. Nicméně pro účely testování metodik pro zajištění odolnosti proti poruchám není míra zrychlení důležitá. Významné je, že procesor běží na FPGA a je zajištěna komunikace se SW stranou, což umožňuje využití v navrhované platformě.

6 Závěr

V tomto článku byla představena základní myšlenka disertační práce zabývající se využitím funkční verifikace pro testování FT metodik v systémech založených na FPGA. Byl zde představen úvod do řešené problematiky, který položil základy pro formulaci cílů disertační práce. Na základě představených cílů byl nastíněn způsob řešení. Druhá část článku se zabývala prezentací již realizovaných výsledků, mezi které patří implementace experimentálního elektromechanického systému, první verze platformy pro testování, prozatím bez aplikace funkční verifikace. Představen byl také koncept akcelerace funkční verifikace procesoru pomocí FPGA, nebylo dosaženo příliš dobrých výsledků ve zrychlení verifikace, ale představený procesor běžící na FPGA bude použit jako základ dalšího experimentálního systému. Jak již bylo řečeno, dosavadní výsledky jsou shrnuty v časopise Microprocessors and Microsystems [1].

Další práce bude směřována především k rozšíření první verze platformy o techniky funkční verifikace, bude tedy vytvořeno verifikační prostředí pro řídicí jednotku robota běžící na FPGA. Toto verifikační prostředí bude dále rozšířeno o injekci poruch do FPGA, což umožní provádění dalších rozsáhlejších experimentů.

Reference

[1] Podivinsky, J.; Cekan, O.; Simkova, M.; Kotasek, Z.: The evaluation platform for testing fault-tolerance methodologies in electro-mechanical applications, Microprocessors and Microsystems, 2015, ISSN 0141-9331, Přijato k publikaci, http://dx.doi.org/10.1016/j.micpro.2015.05.011.

[2] Leen, G.; Heernan, D.: Expanding automotive electronic systems. Computer, ročník 35, č. 1, 2002. ISSN 0018-9162, s. 88-93.

[3] XILINX: Partial Reconfiguration User Guide.

[4] Ceschia, M.; Violante, M.; Reorda, M.; aj.: Identification and classification of single-event upsets in the configuration memory of SRAM-based FPGAs. *IEEE Transactions on Nuclear Science*, ročník 50, č. 6, 2003. ISSN 0018-9499, s. 2088-2094.

[5] Bernardeschi, C.; Cassano, L.; Domenici, A.; aj.: Accurate simulation of SEUs in the configuration memory of SRAM-based FPGAs. In *IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems* (DFT), 2012. s. 115-120.

[6] Rudrakshi, S.; Midasala, V.; Bhavanam, S.: Implementation of FPGA based fault injection Tool (FITO) for testing fault tolerant designs. *IACSIT International Journal of Engineering and Technology*, ročník 4, č. 5, 2012. s. 522-526.

[7] Alderighi, M.; Casini, F.; d'Angelo, S.; aj.: Evaluation of single event upset mitigation schemes for SRAM based FPGAs using the FLIPPER fault injection platform. In *IEEE International Symposium on Defect and Fault-Tolerance in VLSI Systems*, 2007. s. 105-113.

[8] Straka, M.; Kastil, J.; Kotasek, Z.: SEU Simulation Framework for Xilinx FPGA: First Step Towards Testing Fault Tolerant Systems. In *Conference on Digital System Design*, IEEE Computer Society, 2011. ISBN 978-0-7695-4494-6, s. 223-230.

[9] IEEE Std. 1800-2005, *IEEE Standard for SystemVerilog - Unified Hardware Design, Specification, and Verification Language*. 2005.

[10] Rosenberg, S.; Meade, K.: *A practical guide to adopting the universal verification methodology (UVM)*. Cadence Design Systems, 2013.

[11] Gerkey, B.; Vaughan, R. T.; Howard, A.: The player/stage project: Tools for multi-robot and distributed sensor systems. In *Proceedings of the 11th international conference on advanced robotics*, ročník 1, 2003, s. 317-323.

[12] Codasip: Codix RISC. Květ. 2015. URL <https://www.codasip.com/products/codix-risc/>