Testování programovatelných hradlových polí

Kvalita technologie výroby i spolehlivost integrovaných obvodů neustále roste a to platí i o programovatelných hradlových polích. Tím se opět v novém kontextu oživuje odvěký spor, zda je diagnostika potřebná či ne, a pokud ano, kde je její místo.

Úvod

Je nepochybné, že mnoho uživatelů FPGA (Field-Programmable Gate Arrays) se již bez diagnostiky obešlo, takže jejich testování možná nepovažuje za potřebné. Přesto se domnívám, že testování integrovaných obvodů s vysokým stupněm integrace nelze opomíjet, protože např. hradlové pole, stejně jako každý jiný polovodičový obvod, je vystaveno riziku výskytu poruch, jejichž intenzita odpovídá stupni integrace, kvalitě technologie, pracovnímu režimu a řadě dalších vlivů. Proto je v zájmu uživatele, aby se o jeho technickém stavu přesvědčoval, a to nejen ve chvíli, kdy obvod selže. Na rozdíl od ostatních, nerekonfigurovatelných integrovaných obvodů může dokonce u FPGA nastat situace, kdy jsou na diagnostiku kladeny náročnější požadavky ve smyslu lokalizace poruchy. Ta může být užitečná, pokud uživatel dokáže reagovat na zjištěnou poruchu tak, že změní konfiguraci, a tím se vyhne použití poruchového místa na čipu.

Obvody FPGA se mezi konstruktéry i uživateli těší velké oblibě, což lze sledovat nejen na růstu objemu jejich výroby, ale i na růstu cen akcií firem, které je vyrábějí. Důvodem obliby těchto obvodů je především jejich pružnost a přizpůsobivost, díky níž lze návrhářské práce výhodně rozdělit mezi výrobce polovodičových součástek a konstruktéry zařízení, v nichž tyto součástky mají být použity [1]. Právě tato dělba práce však komplikuje testování obvodů FPGA, protože zákazník (tedy konstruktér, který si obvod FPGA koupí), vlastně dostává do ruky polotovar, o jehož konečné funkci výrobce nic neví. V souvislosti s testováním obvodů FPGA tak vznikají dvě samostatné úlohy: testování u výrobce a testování u zákazníka. I když podobné dvě úlohy existovaly při výrobě polovodičových součástek odjakživa, novinkou je jejich vzájemná rozdílnost. Pokud totiž zákazník kupuje hotový obvod s přesně určenou funkcí (např. procesor), může jednak požadovat, aby výrobce právě tuto funkci otestoval, a navíc může sám použít stejné (nebo alespoň podobné) testy jako výrobce, např. při oživování výrobku nebo pokud chce provádět periodickou diagnostiku. Naproti tomu výrobce programovatelných součástek (a to se samozřejmě nevztahuje jen na FPGA) musí zaručit, že obvod bude možno naprogramovat na libovolnou funkci a že tuto funkci bude vykonávat správně. Zákazníkovi naopak většinou stačí ověřit jednu funkci, kterou bude používat.

Několik pojmů z oblasti diagnostiky

Problémům spojeným s testováním hradlových polí lépe porozumíme, když si zopakujeme nejdůležitější zásady, jimiž se diagnostika řídí. V rámci tohoto článku samozřejmě nelze rekapitulovat vše, čeho již bylo v oblasti diagnostiky číslicových obvodů dosaženo. Případné zájemce o podrobnější informace je tedy třeba odkázat na základní studijní texty, např. [2]. Úkolem diagnostiky je zjištění technického stavu testované jednotky, který může být buď bezporuchový, nebo zatížený poruchami (zkráceně se označuje jako poruchový). Zvenčí se o výskytu poruch (což jsou fyzikální změny ve struktuře obvodu, jako např. zkraty, přerušení vodičů, nedovolené změny zpoždění, apod.), obvykle dozvíme prostřednictvím chyb, tedy na základě nesouhlasu mezi hodnotou dat zjištěnou na výstupu a hodnotou, která by tam správně měla být. To znamená, že v převážné většině případů musí diagnostik přistupovat k testované jednotce jako k černé skříňce, o jejímž vnitřku získá informace jen nepřímo, na základě vyhodnocení jejího chování. Potřebný vzorek chování, charakterizující všechny důležité funkce, které má testovaná jednotka provádět, se označuje jako test. Má-li test poskytnout veškerou informaci o technickém stavu testované jednotky, musí být především úplný, tedy musí být schopen detekovat všechny poruchy, které se v jednotce mohou vyskytnout. Navíc ale musí vyhovovat řadě dalších požadavků, mezi nimiž obvykle dominují ekonomické aspekty, protože z hlediska výroby i z hlediska provozu vyrobené jednotky je doba věnovaná testu neproduktivní, takže se ji snažíme zkrátit. Vytváření (generování) testů, které jsou úplné a současně co nejkratší, je klíčovou úlohou diagnostiky, jejíž řešení má rozhodující vliv na její kvalitu jako celku.

Snaha najít nejúčinnější a nejlépe aplikovatelnou testovací metodu vedla ke vzniku poměrně rozmanité nabídky způsobů testování elektronických obvodů a systémů. Podle místa generování a vyhodnocování testovací posloupnosti lze testy rozdělit na vnější a vnitřní. Při použití vnějších testů jsou testovací vektory generovány a vyhodnocovány mimo testovanou jednotku a vnitřní strukturu obvodu není třeba kvůli nim měnit. Vhodnou změnou vnitřní struktury testované jednotky však lze testování významně usnadnit. Pro generování testů se v praxi obvykle používá program, který iterativně vytváří jednotlivé testovací kroky potřebné pro detekci poruch, zkráceně označovaný ATPG (Automatic Test Pattern Generator). Tento program může být poměrně jednoduchý, má-li generovat testy pro kombinační obvody, nebo velmi složitý, pokud má testovat obvody s pamětí. Oddělení kombinační a sekvenční části testovaného obvodu (tzv. strukturovaný návrh) je nejčastěji používaným prostředkem pro zjednodušení (a hlavně zrychlení) činnosti programu ATPG.

Vnitřní (autonomní) testy jsou založeny na použití vestavěných testovacích prostředků BIST (built-in self-test), které do obvodu obvykle vkládá výrobce. Návrh obvodu, který v reálném čase vygeneruje testovací posloupnost pro daný obvod, je poměrně složitá úloha, kterou zatím zdaleka nelze považovat za uspokojivě vyřešenou. Tento generátor je v zásadě možno koncipovat jako pseudonáhodný, deterministický nebo kombinovaný. Pseudonáhodný generátor, realizovaný obvykle jako lineární zpětnovazební posuvný registr (LZPR) nebo celulární automat, generuje velké množství různých vektorů, jejichž efektivita (počet nově pokrytých poruch) s časem klesá. Proto tento postup obvykle vede k použití velkého počtu kroků testu (výjimkou nejsou desítky až stovky tisíců) a tím i na dlouhou dobu testu. Proto se někdy dává přednost deterministickému přístupu, kdy pro posloupnost vstupních vektorů, vygenerovanou programem ATPG (její délka bývá o mnoho řádů kratší než v předchozím případě), se navrhne jednoúčelový sekvenční obvod, který zadanou posloupnost vygeneruje v reálném čase. Protože však takový obvod bývá poměrně složitý, může být výhodným řešením kombinace obou postupů. To znamená, že pseudonáhodný generátor testovacích vektorů použijeme pouze pro detekci části poruch a pro zbývající poruchy, označované jako náhodně netestovatelné (random resistant), vygenerujeme testovací vektory programem ATPG. Základem strategie je to, že pseudonáhodné generování ukončíme ve vhodnou chvíli (když křivka diagnostického pokrytí přestává růst a přechází do stavu nasycení). Tohoto stavu dosáhne generátor obvykle poměrně rychle a dosažené pokrytí přitom často překročí 90 %. Počet zbývajících nepokrytých poruch je tedy poměrně malý, takže malý je i počet deterministicky vygenerovaných testovacích vektorů. Pro ně pak stačí navrhnout jednoduchý sekvenční obvod, který je bude generovat v reálném čase. Celková doba generování testu se tím významně zkrátí.

Nezávisle na předchozím dělení lze testy dále rozdělit na periodické, při nichž test probíhá v krátkých intervalech, které jsou k tomu určeny a v nichž obvod nemůže vykonávat svou normální funkci, nebo průběžné, při nichž jsou diagnostické informace získávány na základě nepřetržité kontroly správnosti zpracovávaných dat. Obvody, v nichž je použita redundantní informace, jejíž správná hodnota je stále kontrolována prostřednictvím hlídačů kódu, se označují jako samočinně kontrolované.

Popsané metody testování lze souhrnně označit jako logické, protože při nich vesměs pracujeme s dvouhodnotovými signály, které lze reprezentovat logickými proměnnými. Kromě logických testů však existují

i metody detekce poruch elektronických obvodů založené na měření analogových veličin, především klidového proudu, který obvod odebírá ze zdroje, tzv. I_{DDQ} (direct drain quiescent current), případně proudu, který obvod odebírá během přechodového děje, tzv. I_{DDT} (direct drain transient current). Toto měření může probíhat buď na vývodech čipu, nebo prostřednictvím senzorů vestavěných přímo do obvodu.

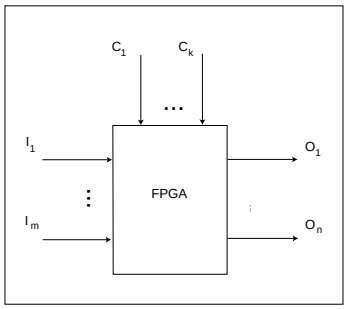
Z popsaných variant a forem realizace diagnostických testů jsou pro FPGA použitelné všechny. Za nejrozšířenější lze považovat periodické vnější testy, proto se jim budeme věnovat nejdříve. Přesto ani ostatní testovací metody nezůstanou stranou naší pozornosti.

Poruchy programovatelných hradlových polí

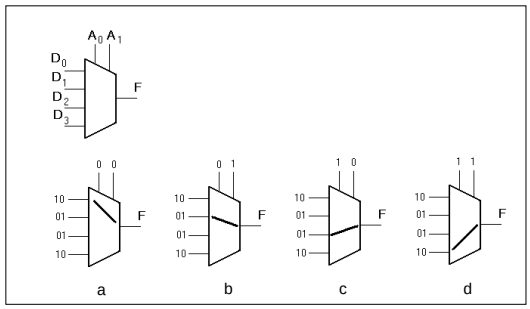
Pokud máme zjišťovat výskyt poruch, případně i místo jejich výskytu pouze na základě odezvy obvodu na test, musíme znát vztah mezi poruchou samou a způsobem, jakým ovlivní signály ve svém okolí. Hlavní problém spočívá v tom, že množina fyzikálních poruch je potenciálně nekonečná. To znamená, že pokud např. mezi dvěma vodiči vznikne svod, jeho odpor se může pohybovat od jednotek ohmů do stovek kiloohmů. I když každý takový svod může ovlivnit chování obvodu trochu jiným způsobem, ve skutečnosti existuje jen několik typů zkratů, které je z hlediska chování třeba rozlišovat. Znamená to, že pro všechny možné poruchy stačí vybrat jen několik typických poruchových stavů, které budou dostatečně věrně popisovat projevy možných fyzikálních poruch. Tento postup se obecně označuje jako tvorba modelu poruchy. Takový model se pak používá tak, že při odvozování diagnostických postupů se zaměříme na detekci, případně lokalizaci tohoto modelu jako reprezentanta poruchy samé. Jako u každého jiného modelu je i v tomto případě třeba najít kompromis mezi dvěma požadavky. Model musí být na jedné straně co nejpřesnější, současně však je třeba, aby byl jednoduchý a umožňoval pokud možno snadnou manipulaci. V praxi se pak přednostně volí model takových poruch, které jsou nejpravděpodobnější, takže kontrola zaměřená na jejich výskyt dává rozumnou záruku, že žádné další poruchy v obvodu nezbyly.

Každá nová technologie vyžaduje kontrolu, do jaké míry lze dosavadní model poruchy považovat za použitelný. Nejjednodušší model poruchy, s nímž diagnostika vystačila od reléové techniky až po bipolární polovodičové součástky, je porucha typu *t* neboli trvalá nula a trvalá jednička (zkráceně *t0* a *t1*). Bylo sice od první chvíle známo, že poruchy typu zkrat tímto modelem reprezentovat nelze, avšak různé nepřímé metody jako např. vícenásobné testování jedné poruchy typu *t* umožnily detekovat též podstatnou část zkratů. Skutečné problémy nastaly až s příchodem technologie CMOS, protože v ní přerušení spoje uvnitř hradla může vést ke vzniku sekvenčního chování obvodu, který byl původně čistě kombinační. Tomu lze čelit tím, že namísto z jednotlivých vektorů sestavíme test z dvojic vektorů, které jsou vhodným způsobem uspořádané (první vždy nastavuje výchozí stav a druhý kontroluje jeho změnu).

Stejně jako u ostatních typů logických obvodů, nelze ani u hradlových polí do zvoleného modelu zahrnout všechny poruchy, které se v obvodu mohou vyskytnout. Musíme se opět soustředit pouze na ty, které jsou nejpravděpodobnější a jim věnovat pozornost při tvorbě testu. Detailní strukturu obvodu FPGA však zná pouze výrobce, a proto uživateli ve většině případů nezbývá, než se spokojit s blokovým schématem, které je k dispozici, a v něm vytvořit modely změn chování jednotlivých funkčních modulů. Praxe ukázala, že i z blokové struktury hradlových polí lze odvodit použitelný model poruch platný pro tento typ odvodů. Tento model obvykle zahrnuje poruchy *t0* a *t1* jednotlivých vstupů a výstupů použitých součástek (multiplexorů, spínačů, paměťových členů, apod.).



Obr. 1 Vstupy hradlového pole



Obr. 2 Úplný test multiplexoru se čtyřmi datovými vstupy

FPGA jako testovaná jednotka

Pokud hledáme vhodnou metodu testování FPGA, musíme nejprve upřesnit, jakému účelu má test sloužit. Z tohoto hlediska lze rozlišit dva hlavní typy testů: výrobní a uživatelské. Uživatelské testy se dále mohou lišit podle toho, zda uživatel používá jednu konfiguraci, nebo konfigurace mění.

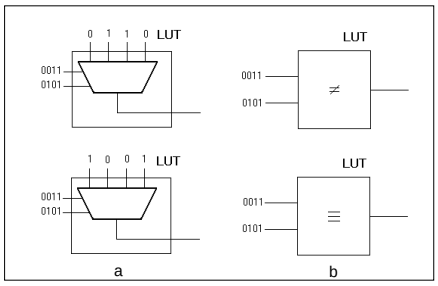
Výrobní testy, tedy testy, které provádí výrobce na nově vyrobeném hradlovém poli, mají ověřit, že obvod je použitelný v libovolném režimu, který připouští technická specifikace výrobce. Lze tedy předpokládat, že tyto testy budou velmi rozsáhlé a budou vyžadovat použití dokonalého přístrojového vybavení. Naproti tomu uživatelský test jedné konfigurace je poměrně jednoduchý, protože v tomto případě se uživatel pouze opakovaně přesvědčuje o tom, že jeho obvod správně vykonává tu funkci, kterou si uživatel zvolil. Někde mezi oběma extrémy leží složitost uživatelských testů FPGA určených pro opakovanou rekonfiguraci. Taková situace typicky vzniká, pokud uživatel používá FPGA pro nějakou časově omezenou funkci, např. pro ověřování prototypu (rapid prototyping). V takovém případě uživatel typicky vytvoří určitou konfiguraci, provede s ní potřebný počet experimentů, aby si ověřil, zda navrhovaný systém bude pracovat správně a v případě, že funkce neodpovídá specifikacím, změní konfiguraci a provede další sadu experimentů, atd. Po dosažení správné funkce může buď použít FPGA jako součástku, která bude požadovanou funkci vykonávat v konečném výrobku (to platí především tehdy, jedná-li se o kusovou výrobu), nebo přenese funkci z FPGA do zákaznického obvodu (ASIC), čímž se FPGA uvolní pro další experiment. Četnost a rozsah testů, které uživatel bude provádět při popsaném způsobu využití FPGA, bude mimo jiné záviset na poruchovosti obvodu, na prostředí, v němž je obvod provozován, na jeho zatížení, na době, která uplyne mezi jednotlivými experimenty, atd. Uživatel tedy musí být především schopen otestovat každou konfiguraci, kterou vytvoří. Má-li však důvod pochybovat o bezporuchovém stavu obvodu jako celku, měl by mít k dispozici i výrobní test a provést jej před zahájením konfigurace. Toto je zvlášť nutné u dynamicky rekonfigurovatelných aplikací. Proti většině běžných (nerekonfigurovatelných) integrovaných obvodů FPGA představuje z hlediska testování komplikovanější objekt, mimo jiné i proto, že jeho vstupy nejsou rovnocenné. Schématicky je tento rozdíl naznačen na *obr. 1*, kde jsou odděleny tzv. konfigurační vstupy C_i, jejichž prostřednictvím lze do hradlového pole vložit informace určující požadovanou konfiguraci, a datové vstupy I_i (někdy označované též jako operační), jimiž přivádíme data v operačním režimu. Při testování je podstatné, že tyto dva typy vstupů se značně liší z hlediska ovladatelnosti vnitřních součástek obvodu. Datové vstupy lze připojit přímo na libovolnou logickou buňku, zatímco informace přivedené na konfigurační vstupy se vkládají do velmi dlouhých sériových řetězců paměťových obvodů.

Před zahájením testu se především musíme přesvědčit o správnosti naprogramování, tedy o tom, že data zapsaná do konfiguračních pamětí jsou správná. K tomu slouží jednak kontrola cyklickým kódem, jednak možnost přečíst obsah pamětí po zápisu. Test samotného hradlového pole se pak skládá z testu konfigurovatelných bloků (včetně periferních) a testu spojení. Těmito dvěma fázemi se budeme zabývat v dalším textu. Celá testovací posloupnost pro výrobní test hradlového pole má pak tvar K_1, T_1, K_2, T_2, ..., K_n, T_n, kde K_i je *i*-tá konfigurační posloupnost, T_i je testovací posloupnost pro tuto konfiguraci a *n* je celkový počet konfigurací, které je během testu třeba vytvořit. Pokud se snažíme zkrátit délku testu na nejmenší možnou míru, musíme si předem ujasnit, jaké máme možnosti. Délka konfigurační posloupnosti je pevně daná a vyplývá z počtu paměťových míst, do nichž je třeba zapsat konfigurační informace. Můžeme se tedy zaměřit na sní-

žení počtu různých konfigurací n a na zmenšení délky testovacích posloupností T_i určených pro testování jednotlivých konfigurací. Všechny tyto veličiny jsou vzájemně závislé, takže přesná optimalizace by představovala nesmírně složitou matematickou úlohu. V praxi se však ukazuje, že délka konfigurační posloupnosti K_i je mnohonásobně (obvykle o několik řádů) větší než délka kterékoli testovací posloupnosti. Přijatelných výsledků se proto dá dosáhnout pouze tak, že se především soustředíme na zmenšení počtu konfigurací, zatímco délkám testovacích posloupností není třeba věnovat velkou pozornost.

Výrobní testy

Příprava efektivních výrobních testů FPGA je nesmírně složitý problém, jehož řešením se zabývají nejen výrobci sami, ale s nimi i řada výzkumných týmů na celém světě,



Obr. 3 Test generátoru funkce realizovaného multiplexorem

o čemž svědčí velké množství publikací věnovaných tomuto tématu. Jako vždy takovýto zájem signalizuje, že na jedné straně je řešení problému naléhavě nutné, na druhé straně, že žádná z dosud navržených metod nevyhovuje beze zbytku. Potíž spočívá v tom, že má-li výrobce zaručit, že uživatel bude skutečně moci vytvořit na daném obvodu libovolnou konfiguraci, kterou mu nabízí technický popis výrobku, měl by ji předem sám ověřit během výstupního testu. Počet možných konfigurací je však nepředstavitelně vysoký, o čemž se můžeme přesvědčit jednoduchou úvahou. Pro popis konfigurace jednoho konfigurovatelného logického bloku (CLB) hradlového pole Xilinx řady 4000 je třeba vložit do pole 68 bitů, takže samotný tento blok lze konfigurovat celkem 2^{68} různými způsoby. Tomu odpovídá v desítkové soustavě číslo s dvaceti nulami, takže pokud bychom pro nahrání a otestování jedné konfigurace potřebovali např. 10 µs, trvalo by podrobné ověření všech možných konfigurací jediného bloku téměř sto milionů let.

To ovšem zdaleka není všechno. Navíc je třeba konfigurovat i spoje mezi těmito bloky, takže celkový počet konfiguračních bitů, které v průměru připadají na jeden blok, je asi 350. Obvod XC4013 obsahuje 24×24=576 bloků, takže pro jednu konfiguraci je třeba vložit do obvodu více než 200 000 bitů. Při rychlosti přenosu 10 Mb/s dosáhne doba konfigurace téměř 25 ms, a je proto zcela pochopitelným požadavkem snížit co nejvíce počet různých konfigurací použitých během testu. To je také hlavní cíl všech metod hledajících optimální metodu testování logických bloků hradlových polí, zatímco počet testovacích vektorů přivedených na vstupy obvodu a vyhodnocených na výstupech při jedné konfiguraci hraje podružnou roli.

Logické bloky FPGA nelze testovat současně tak, že bychom na jejich vstupy paralelně přiváděli testovací vstupy z primárních (tj. vnějších) vstupů obvodu a jejich výstupy paralelně vyhodnocovali na primárních výstupech, protože hradlové pole nemá dostatek vývodů na to, abychom na ně připojili všechny bloky současně. Proto je obvyklé zajišťovat přístup k jednotlivým blokům pro-

Tabulka 1 Testovací konfigurace

XC4000	5
XC3000	4
SPARTAN	4
ALTERA	3
ORCA 2C	9

střednictvím jejich sousedů, tedy spojovat bloky do řetězců. To samo o sobě by mohlo znamenat značnou komplikaci z hlediska počtu konfigurací a délky testu pro každou z nich. Naštěstí však na kombinační část všech bloků lze uplatnit teorii iterativních sítí, které lze testovat v konstantním čase.

Teorie iterativních sítí je samostatná kapitola diagnostiky, jejíž výklad by sám mohl zabrat celý článek. Zde se omezíme jen na konstatování, že iterativní kombinační obvod je testovatelný v konstantním čase (C-testovatelný), jestliže libovolně dlouhý řetězec stejných buněk lze otestovat stejným počtem kroků jako jednu buňku. Při tom předpokládáme, že každá buňka přijímá určité signály z primárních vstupů a může být připojena i na primární výstupy, avšak určitá část signálů propojuje buňky mezi sebou. To samozřejmě znamená, že každá buňka v libovolném místě řetězce dostává od své předchůdkyně signály tvořící součást testu a že její výstupy směřující k následující buňce musí být nezkresleně přeneseny až na primární výstupy.

Nejnázornějším příkladem C-testovatelného iterativního obvodu je generátor parity sestavený z obvodů nonekvivalence. Každý člen nonekvivalence sám potřebuje pro své úplné otestování všechny čtyři možné kombinace hodnot na svých vstupech (triviální test). Spojíme-li členy nonekvivalence do libovolně dlouhého řetězce, lze výsledný iterativní obvod otestovat také čtyřmi kroky, protože při průchodu signálu členem nonekvivalence se žádná diagnostická informace neztratí.

Zkonfigurovat hradlové pole tak, aby bylo C-testovatelné, znamená najít konfiguraci, která by se opakovala ve všech blocích, a tvořila tak základní buňku iterativní sítě. Část vstupů každého bloku pak bude připojena na signály přiváděné z primárních vstupů obvodu a část bude připojena na výstupy bloku, který je předchůdcem daného bloku v síti. Výstupy každého bloku budou připojeny pouze na vstupy následujícího bloku, takže na primární výstupy obvodu bude připojen jen poslední blok v řetězci. Základním požadavkem kladeným na vnitřní strukturu zkonfigurovaného bloku je schopnost přenášet diagnostické informace ze vstupů na výstupy a navíc generovat během testu na svých výstupech takové vektory, které jsou potřebné při testu následujícího bloku. Tato koncepce je společná v podstatě všem navrženým metodám testování hradlových polí. Rozdíl je pouze v tom, na kolik se podařilo zredukovat počet konfigurací a kolika vektory je testována každá z nich. Minimální potřebné počty konfigurací odvozené v [2] pro nejznámější typy hradlových polí jsou uvedeny v *tabulce 1*. Pro každou z uvedených konfigurací je třeba vygenerovat úplný test a přesvědčit se, že je správně proveden i v iterativní síti sestavené z bloků v dané konfiguraci.

Odvození testovacích konfigurací

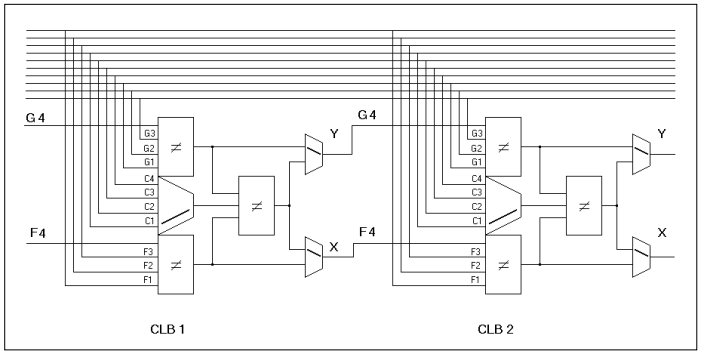
Pro potřeby diagnostiky je účelné rozdělit CLB na kombinační a sekvenční část a každou z nich testovat zvlášť. Propojením vhodně zkonfigurovaných kombinačních částí CLB pak můžeme vytvořit C-testovatelný iterativní obvod.

Kombinační část typického CLB je tvořena multiplexory a generátory funkcí, označovanými LUT (look-up table). Nejprve si povšimneme možností testování těchto základních stavebních modulů samostatně. Jednoduchý příklad multiplexoru je uveden na *obr. 2*. Má čtyři datové vstupy D_0, D_1, D_2, D_3 a dva adresové vstupy A_0, A_1. Prostřednictvím adresy přivedené na vstupy A_0, A_1 připojíme jeden ze vstupů D_i na výstup F, takže hodnota přivedená na zvolený vstup bude beze změny přenesena na výstup. Lze odvodit, že úplný test multiplexoru z *obr. 2* je tvořen osmi vektory, protože je třeba ověřit správný výběr každého ze čtyř vstupů a při tomto výběru vyzkoušet, zda se ze vstupu správně přenáší nula i jednička. Tento test je symbolicky znázorněn na *obr. 1* (*a* až *d*), kde výhybka uvnitř symbolické značky multiplexoru naznačuje, který vstup byl právě vybrán. Vzhledem k tomu, že adresové vstupy multiplexorů se na hradlovém poli nastavují během konfigurace, lze popsaný test provést tak, že postupně nahrajeme čtyři konfigurace a pro každou přivedeme na datové vstupy dva kroky testu, při nichž použijeme vektory 1001 a 0110.

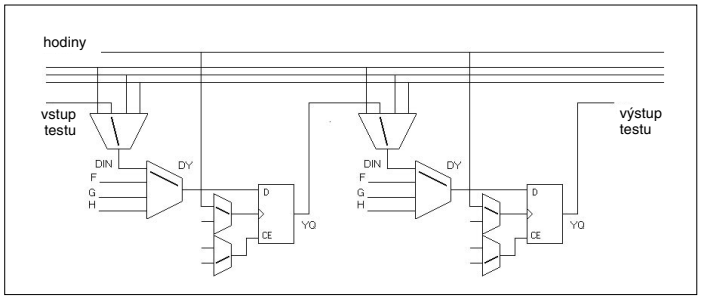
Test generátoru funkce (LUT) lze odvodit obdobně. Generátory funkcí jsou na programovatelných hradlových polích obvykle realizovány také jako multiplexory, ale s prohozenou úlohou konfiguračních a datových vstupů. Zachováme-li značení, při němž je konfigurace přiváděna shora a data zleva, můžeme generátor funkce dvou proměnných symbolicky znázornit podle *obr. 3*. Na vstupy multiplexoru D_0, D_1, D_2, D_3 tentokrát přivádíme hodnoty generované funkce, které nastavíme během konfigurace pole (tyto hodnoty se zapisují do paměťových buněk připojených ke vstupům D_i). Na vstupy A_0, A_1 přivádíme hodnoty vstupních proměnných, takže zvolenou adresou vybereme z tabulky hodnot tu, která se má objevit na výstupu.

Multiplexor z *obr. 3* je třeba testovat stejně jako multiplexor z *obr. 2*. To znamená, že je třeba znovu vybrat postupně všechny vstupy D_i a na každý přivést dvě hodnoty. Protože však jsou nyní jednotlivé vstupy ovládány odlišně, použijeme dvě konfigurace (1001 a 0110) a pro každou z nich čtyři testovací vektory přivedené na adresové vstupy. Tyto vektory jsou uvedeny v *obr. 3*. Funkce generovaná na výstupu obvodu pro dvě zvolené konfigurace je nonekvivalence (XOR) a ekvivalence. Proto jsou pro tyto konfigurace na *obr. 3* použity symbolické značky ≠ a ≡.

Popsané základní principy testování multiplexorů a generátorů funkcí lze poměrně snadno zobecnit na obvody větších rozměrů a využít při testování kombinační části CLB. Na *obr. 4* je příklad nastavení jedné testovací konfigurace kombinační části CLB a způsob propojení dvou CLB do iterativní sítě. Nastavení multiplexorů je znázorněno výhybkami, všechny LUT jsou zkonfigurovány do funkce XOR. Výstupy *X*, *Y* jsou spojeny vždy se vstupy *F4*, *G4* následujícího bloku. Vstupy *F4*, *G4* prvního bloku v řetězci jsou připojeny na primární vstupy obvodu, výstupy *X*, *Y* posledního bloku jsou připojeny na primární výstupy. Ostatní vstupy každého bloku (C_i, F_i, G_i) jsou připojeny na společný rozvod signálů přiváděných z primárních vstupů. Počet logických bloků skutečně propojených do jednoho řetězce lze přizpůsobit možnostem, které nabízejí vývody pole. Logické bloky lze propojit buď do jednoho dlouhého řetězce, nebo do několika kratších řetězců testovaných paralelně. Rozdělení dlouhého řetězce do několika kratších je výhodnější, pokud požadujeme též lokalizační informaci.



Obr. 4 Příklad nastavení jedné testovací konfigurace kombinační části CLB a způsob propojení dvou CLB do iterativní sítě



Obr. 5 Příklad konfigurace použitelné pro test paměťových členů

Jak bylo uvedeno v předchozím odstavci, pro XC4000 je třeba vytvořit 5 takových konfigurací. Zbývající čtyři konfigurace se odvodí tak, že ve třech z nich ověříme přenos signálu ze vstupů F_3, G_3 až F_1, G_1 a v poslední změníme funkci LUT na ekvivalenci a přepneme výstupní multiplexory do opačné polohy. Pro každou konfiguraci je třeba vygenerovat úplný test, který má asi 40 kroků.

Po testu kombinační části CLB je třeba ještě otestovat paměti obsažené v těchto blocích. Pro paměťovou část stačí použít jednoduchý test na *t1* a *t0* každého klopného obvodu zvlášť, protože v tomto případě se nejedná o paměťovou matici, v níž by bylo třeba ověřovat i citlivost na různé vzorky. Příklad konfigurace použitelné pro test paměťových členů je uveden na *obr. 5*. Paměťové členy jsou propojeny do posuvného registru, jehož funkci snadno ověříme posloupností střídavých nul a jedniček vloženou na jeho začátek a čtenou na jeho konci. O počtu posuvných registrů testovaných paralelně platí podobná úvaha jako v předchozím případě. Rozdíl spočívá v tom, že u iterativního kombinačního obvodu délka testu nezáleží na délce řetězce, zatímco u posuvného registru ano.

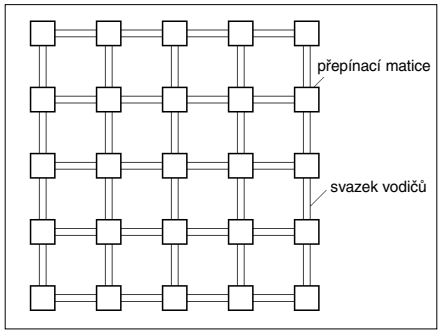
Prof. Ing. Jan Hlavička, DrSc.,
katedra počítačů FEL ČVUT
(*Pokračování v příštím čísle)*

Testování programovatelných hradlových polí

(pokračování z minulého čísla)

Test propojovacích struktur

Při testování logických bloků se současně testuje i značná část propojovací sítě hradlového pole, protože mezi bloky se informace přenáší právě pomocí této sítě. Obvykle se uvádí, že během testu logických bloků se otestuje až 80 % sítě čipu. Přesto nelze takový test považovat za uspokojivý, takže po testu logických bloků musí ještě následovat test sítě. Popis metod testování propojovací sítě FPGA musíme opět zahájit studiem její struktury a upřesněním typů poruch, s nimiž musíme počítat. Propojovací síť je obvykle tvořena svazky vodičů, které obepínají logické bloky (viz obr.6). V průsečících těchto svazků jsou přepínací matice, které umožňují propojit některé z vodičů v křižících se svazcích. Struktura jedné přepínací matice je znázorněna na



Obr. 6 Struktura propojovací sítě hradlového pole

obr.7. Ze šestnácti vodičů vstupujících do matice je možno propojit vždy ty, které mají stejný index. Na obr.8 jsou čárkovaně vyznačeny všechny možnosti propojení vodičů N_4, E_4, S_4 a W_4.

Mezi poruchami propojovací sítě musíme vzít v úvahu především přerušení vodičů a zkraty mezi nimi. Kromě toho musíme počítat s poruchami spínacích tranzistorů v maticích, konkrétně s trvale sepnutým a s trvale rozepnutým stavem. Zkraty mezi těmito prvky lze převést na zkraty vodičů mimo matici.

Pro úplný test přepínací matice byl v [4] odvozen postup vyžadující 3 konfigurace, přičemž tento počet je nezávislý na velikosti matice. Tyto konfigurace můžeme symbolicky označit jako „levá diagonála“, „pravá diagonála“ a „kříž“. Levá diagonála je schématicky znázorněna na obr.8. Paralelní spoje ve spínacích maticích spojují vždy dva vývody se stejným indexem, tedy např. W_1–S_1, W_2–S_2 atd. Diagonální spoje typu W–S a N–E jsou vytvořeny ve všech přepínacích maticích ležících uvnitř pole. Naproti tomu přepínací matice ležící na okrajích pole jsou zkonfigurovány tak, aby všechny diagonály byly navzájem spojeny za sebou, takže všechny spoje lze projít „jedním tahem“.

Pravá diagonála tvoří obrazec, který je symetrický s obr.8 podle svislé osy. Konfigurace typu „kříž“ je znázorněna na obr.9. Vnitřní matice nyní spojují protilehlé vodiče a okrajové matice spojují řádky a sloupce mezi sebou. Lze ukázat, že i po vystřídání těchto tří konfigurací jsou některé matice na okraji pole testovány jen částečně. To lze kompenzovat např. tak, že jednu z popsaných konfigurací otočíme o 90°. V každé z konfigurací jsou všechny spínací matice propojeny do jednoho řetězce a protože v cestě nestojí žádná paměť, můžeme je testovat současně. Pro počet testovacích vektorů potřebných pro testování výše uvedených poruch propojovací sítě v každé z uvedených konfigurací byl odvozen vzorec $\log_2(k+2)$, kde k je počet vodičů v jednom svazku. Pro náš případ ($k=4$) by to znamenalo 3 vektory.

Zápis do rozhraní

Mezi vestavěnými diagnostickými prostředky je třeba se zmínit též o tzv. zápisu do rozhraní (boundary scan). Je to technika umožňující přímý přístup ke všem vývodům integrovaného obvodu během testu. Cílem úpravy je mít možnost ovládat vývody, případně číst hodnoty na těchto vývodech u obvodu, který je již zapojen ve schématu (např. na desce), a byl by tedy jinak z konektoru přímo nepřístupný. Zápis do rozhraní se řídí mezinárodní normou IEEE 1149.1, takže obvody vybavené touto technikou lze vzájemně propojovat a ovládat stejným způsobem, i když pocházejí od různých výrobců. Úprava obvodů pro zápis do rozhraní je poměrně nákladná, protože na každý vývod pouzdra je přidán jeden klopný obvod, schopný pracovat v režimu paralelního nebo posuvného registru. Přesto je tato metoda u nových typových řad programovatelných obvodů již brána téměř jako samozřejmost, protože výhody převažují nad nevýhodami. Režimy činnosti obvodů rozhraní se přepínají pomocným řídicím vstupem obvodu. Posuvný registr obepíná celý obvod a všechny posuvné registry obvodů na jedné desce se obvykle spojují do série, takže vznikne dlouhý řetězec klopných obvodů, do něhož lze v diagnostickém režimu zapsat hodnoty, které potřebujeme pro provedení jednoho kroku testu, a z něhož lze také přečíst všechny hodnoty, které vznikly na výstupech obvodů jako odezva na test. I když vznik zápisu do rozhraní byl původně motivován snahou usnadnit především test plošných spojů mezi obvody, lze jej dobře využít i k testování obvodů samých. Typický postup je pak následující:

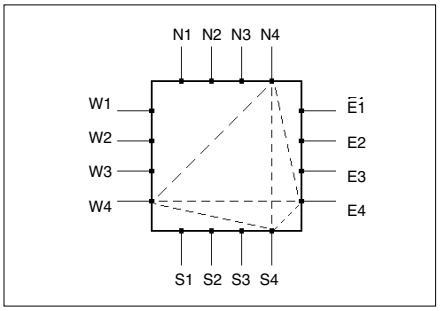
– sériové naplnění posuvného registru rozhraní hodnotami tvořícími vstupní vektory testu,
– provedení jednoho nebo několika kroků testu,
– zápis výstupních hodnot testu do paměťových členů rozhraní,
– přečtení získaných hodnot sériově v režimu posuvného registru.

Tento postup je samozřejmě velmi pomalý, protože před každým krokem testu je třeba sériově naplnit celý posuvný registr, který může na desce obsahovat i několik tisíc paměťových členů propojených do série. Čtení výsledků (odezvy na test) může probíhat současně se zápisem nových hodnot.

Funkci zápisu do rozhraní lze s výhodou využít i při testech programovatelného hradlového pole, proto ji výrobci FPGA běžně používají. V okolí FPGA je pak třeba vytvořit obvody, které budou z posuvných registrů vkládat testovací vektory do rozhraní a posuvných registrů, které budou z rozhraní testovaného obvodu číst odezvy.

Uživatelské testy

Uživatelský test FPGA je obvykle mnohem snadnější než test u výrobce, protože uživateli většinou stačí zkontrolovat, že ta konfigurace, kterou si pro svoji aplikaci zvolil, funguje správně. Uživatelský test tedy spočívá v tom, že na datové vstupy zkonfigurovaného obvodu je přiveden potřebný počet vstupních vektorů, jejichž odezva je vyhodnocena. Tyto vstupní a výstupní vektory je možno vygenerovat např. programem ATPG, do nějž vložíme informace o vnitřní struktuře hradlového pole a o použité konfiguraci. Protože však nastavená konfigurace stěží bude odpovídat některé z transparentních konfigurací tvořících iterativní síť a citovaných v souvislosti s výrobními testy, je třeba počítat s tím, že potřebný počet vektorů bude podstatně vyšší (řádu stovek až tisíců). Délku testu výrazně ovlivní především to, že uživatelská konfigurace zpravidla používá hradlové pole jako sekvenční obvod. Pokud uživatel sám nepoužil strukturovaný návrh, budou zjednodušené metody generování testů pro kombinační obvody a jimi odvozené krátké testy v tomto případě nepoužitelné.



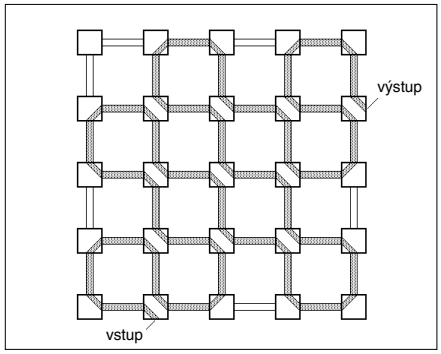
Obr. 7 Přepínací matice

Je-li uživatelský test prováděn jako vnější, může být hradlové pole testováno buď ve zkoušeči integrovaných obvodů (jehož existence u běžného uživatele je spíše nepravděpodobná), nebo v systému (na desce), kde je normálně provozováno. Ve druhém případě musí uživatel mezi funkce desky zařadit i možnost přivést na vstupy hradlového pole potřebné vstupní vektory testu a vyhodnotit na desce získané odezvy. Testovací vektory přitom mohou být buď čteny z paměti, kam byly předem uloženy, nebo mohou být generovány v reálném čase jednoúčelovým obvodem. Podobně i odezvy mohou být jednotli-

vě srovnávány s předem uloženými správnými odezvami, nebo (což je obvyklejší) komprimovány (např. paralelním příznakovým analyzátorem) a vyhodnoceny jako celek.

Použití vestavěných diagnostických prostředků

Pokud je po realizaci požadované funkce na programovatelném hradlovém poli ještě místo pro vestavěné diagnostické prostředky, může uživatel použít autonomní test programovatelného hradlového pole (BIST). Konfigurace pole opět může zůstat po dobu testu nezměněna, takže jediným problémem je zvolit typ generátoru testovací posloupnosti a příznakového analyzátoru a obě tyto jednotky vhodným způsobem realizovat v programovatelném hradlovém poli. Pro realizaci generátoru testovací posloupnosti je možno po malé úpravě použít též posuvný registr obepínající vstupy obvodu vybaveného možností zápisu do rozhraní. Kromě generátoru testů je třeba doplnit též řídicí jednotku, která na základě povelu zvenčí zahájí a ukončí test



Obr. 8 Konfigurace „levá diagonála"

a ohlásí jeho ukončení včetně výsledku na některém výstupním vývodu.

Použití metody BIST je úmyslně zařazeno pouze mezi uživatelské testy, protože pro výrobce by tato metoda byla použitelná jen velmi obtížně. BIST může generovat a vyhodnocovat pouze testovací vektory, zatímco konfiguraci (jejíž součástí je vytvoření obvodů BIST) je třeba nahrát zvenčí.

Samočinně kontrolované obvody

V nabídce metod testování hradlových polí je třeba citovat i možnost použít průběžnou diagnostiku, tedy navrhnout obvod realizovaný na FPGA jako samočinně kontrolovaný. Zvolíme-li tuto metodu, nemusíme provádět periodické testy, protože technický stav obvodu bude průběžně sledován a signalizován navenek. Za tuto úsporu času však zaplatíme cenou nadbytečných obvodů, které musíme na hradlovém poli realizovat. Jsou to kodéry a dekodéry bezpečnostních kódů a struktury potřebné pro přenos, záznam a zpracování nadbytečné informace. Existuje celá řada metod návrhu takových obvodů, mezi nimiž vynikají především tzv. úplně samočinně kontrolované obvody, které zaručují nepřetržitou signalizaci chybné funkce ve všech částech včetně hlídačů kódu, takže použití periodických testů zcela odpadá. Vzhledem k tomu, že se opět jedná o testování jedné konkrétní konfigurace, je tato metoda použitelná pouze pro uživatelské testy.

FPGA jako vnořené jádro

Vnořené jádro (embedded core) nebo též IP-blok (intellectual property block) je obvykle poměrně složitý funkční celek, který návrhář čipu použije jako hotový produkt, a tím si ušetří práci spojenou s jeho návrhem, ověřováním, optimalizací, oživováním, atd. Jádra si lze vybrat podle katalogu dodavatelů, kteří se na jejich tvorbu specializují a dodávají je buď jako „soft", tedy ve formě popisu na úrovni jazyka pro logický návrh (např. VHDL), nebo „hard", kdy návrh jádra je dokončen a předává se uživateli ve tvaru konkrétní struktury určené pro realizaci v rámci určité technologie. Typickými jádry jsou procesory, signálové procesory, operační paměti, vyrovnávací paměti, periferní obvody, apod. V poslední době se navíc objevila i jádra tvořená programovatelnými logickými poli.

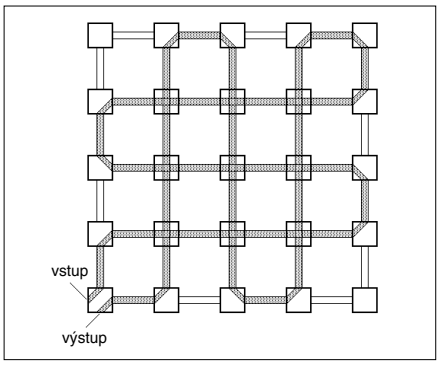
Testování vnořených jader obecně má několik specifik a jádro typu FPGA k nim přidává ještě některá další omezení. Hlavním problémem při testování vnořených jader je jejich omezená dostupnost z vývodů pouzdra, protože mezi primárními vstupy obvodu a vstupy jádra na jedné straně a výstupy jádra a primárními výstupy obvodu na druhé straně může ležet řada různých jednotek, např. jiná jádra, nebo konstruktérem navržená propojovací logika (glue logic). Pro řešení tohoto problému byla navržena velmi radikální, bohužel však značně nákladná metoda řešení, spočívající ve vytvoření sériové přístupové cesty z vývodů pouzdra přímo na vstupy a výstupy každého jádra (jakási analogie zápisu do rozhraní, používaného běžně na úrovni pouzder integrovaných obvodů).

Druhým závažným problémem testování vnořených jader je omezená znalost jejich struktury. Dodavatel zpravidla pouze zaručí, že jádro vykonává předepsanou funkci podle katalogu a nemá povinnost (a většinou ani zájem, proto se mluví o intelektuálním vlastnictví), seznamovat uživatele s vnitřní strukturou jádra. Pokud se uživatel nechce spokojit s funkčním testem nebo se vydat na trnitou cestu reverzního inženýrství, musí použít test dodaný výrobcem jádra.

V souvislosti s programovatelnými obvody může být koncepce vnořeného jádra uplatněna na dvou úrovních. V nabídce výrobců programovatelných hradlových polí se dnes jako „jádra" nejčastěji označují konfigurace (makra), pokrývající část pole a realizující určitou často požadovanou funkci (např. procesor). Návrh pak spočívá v umístění určitého počtu takových jader na programovatelném poli a v jejich propojení logikou podle vlastního návrhu. Pod tlakem návrhářů však výrobci začínají nabízet i jiný typ jader FPGA, a to nezkonfigurované hradlové pole, které lze vnořit do zákaznického obvodu. Taková kombinace obou technik návrhu může být užitečná např. tehdy, potřebuje-li návrhář hotový návrh na poslední chvíli přizpůsobit měnícím se podmínkám, např. nové normě upřesňující přenosový protokol. Pak se jedná v pravém slova smyslu o vnořené jádro FPGA do obvodu s pevnou strukturou. V takovém případě musí výrobce obvodu provést úplný test vnořeného programovatelného hradlového pole stejně jako výrobce samostatného pole. Metoda návrhu obvodu, zajišťující přístup k jádru, by mu měla umožnit, že tento test proběhne v přijatelné době. Pokud bude obvod s vnořeným jádrem testovat koncový uživatel, je pravděpodobné, že se omezí na tu konfiguraci, kterou používá. Test může i v případě jader typu FPGA významně zkrátit použití vestavěných diagnostických prostředků, případně samočinnou kontrolu realizované funkce.

Několik poznámek na závěr

Testování hradlových obvodů je pouze jednou z řady nových úloh, před něž je diagnostika neustále stavěna v důsledku pokroku v oblasti technologie. Cílem tohoto článku bylo ve stručnosti upozornit na nejdůležitější úskalí této úlohy a naznačit také možná řešení. S ohle-



Obr. 9 Konfigurace „kříž"

dem na mnohotvárnost struktur programovatelných hradlových polí různých výrobců je však třeba počítat s tím, že v každém konkrétním případě bude třeba popsaný postup v podstatě zopakovat. To znamená, že musíme začít od analýzy poruchových mechanizmů a volby modelu poruchy. Pak bude následovat vyčlenění jednotlivých konstrukčních bloků (logických, propojovacích), jejich analýza a odvození testu. Jedná-li se o výrobní test, je třeba najít testovací postup, při němž bude použit minimální počet konfigurací. Uživatel se naproti tomu obvykle může soustředit na tu konfiguraci, kterou právě používá.

Prof. Ing. Jan Hlavička, DrSc.
katedra počítačů FEL ČVUT

LITERATURA

[1] Chan, P.K., Mourad, S.: Digital design using field programmable gate arrays. Prentice-Hall, 1994

[2] Hlavička, J.: Diagnostika a spolehlivost. Praha, Vydavatelství ČVUT 1998, str. 153

[3] Renovell, M., et al.: Test configuration generation for FPGA cells. Digest of papers, 1st IEEE Latin American Test Workshop LATW2000. Rio de Janeiro, March 2000, str. 202–208

[4] Renovell, M., et al.: Testing the interconnect structures of unconfigured FPGA. Proc. IEEE European Test Workshop, Sete, June 1996