Generování testu s nulovým maskováním poruch

Robert Hülle
2. ročník prezenčního studia
Školitel: Petr Fišer, školitel specialista: Jan Schmidt

České vysoké učení technické v Praze, Fakulta informačních technologií
Thákurova 9, Praha, 16000
hullerob@fit.cvut.cz

***Abstrakt*—V tomto článku shrnuji své výsledky za 2. rok doktorského studia. Prezentuji automatický generátor testovacích vektorů (ATPG), který je schopen vygenerovat test s nulovým maskováním v daném libovolném kompaktoru: ZATPG. Diskutuji silné a slabé stránky tohoto algoritmu, včetně námětů, jak jeho slabé stránky překonat. V závěru nastiňuji další směřování výzkumu, které by mělo vést k disertační práci.**

***Klíčová slova*—test, testování, generování testu, porucha trvalá 0/1, ATPG, SAT, kompakce odezvy, maskování poruch, nulové maskování, LFSR, MISR, BIST.**

I. Úvod

V moderním světě se dennodenně setkáváme s číslicovými systémy, které nám pomáhají organizovat náš čas, poskytují nám zábavu, na nichž závisí naše životy. Není tedy divu, že testování číslicových systémů je velmi důležité. Se zvyšováním složitosti návrhu roste i náročnost generování, aplikace, a vyhodnocení testu.

Jedna z možností, jak snížit cenu aplikace testu a zvýšit pokrytí poruch, je využít prostředků pro vestavěnou diagnostiku (Built-in Self-Test, BIST).

A. Prostředky vestavěné diagnostiky

Prostředky vestavěné diagnostiky (BIST) sestávají z několika částí:

1) Generátor testovacích vektorů: jedná se o sekvenční obvod schopný generovat testovací vektory a ty posléze aplikovat na vstupy testovaného obvodu.

2) Analyzátor odezev: rozhoduje, zda odezva testovaného obvodu odpovídá teoretické odezvě bezporuchového obvodu. Tento obvod lze opět navrhnout mnoha způsoby, časté je jeho rozdělení na statický kompaktor, dynamický kompaktor a komparátor.

Statický kompaktor je kombinační obvod, který zmenšuje počet testovacích výstupních signálů, např. paritní strom.

Dynamický kompaktor je sekvenční obvod, který zpracovává sekvenci odezev testovaného obvodu na sekvenci testovacích vektorů a vytváří z nich *otisk* (signature) obvodu.

Komparátor porovnává výsledný otisk obvodu na konci testu s předpočítaným otiskem bezporuchového obvodu. Pokud se otisk liší, je obvod vyhodnocen jako poruchový.

3) BIST řadič: jedná se o obvod, který řídí běh testu. Tento obvod je zodpovědný za inicializaci generátoru testovacích vektorů i dynamického kompaktoru.

B. Maskování poruch

V případě, že testujeme obvod s poruchou, která se projeví jako chyba na výstupu obvodu, ale výsledný otisk obvodu je shodný s otiskem bezporuchového obvodu, nastalo tzv. *maskování*. Jedná se o jev nanejvýš nežádoucí, neboť snižuje celkové pokrytí poruch testem. Může nastat jak ve statickém, tak v dynamickém kompaktoru.

Ve statickém kompaktoru mohou nastat dva typy maskování. První způsobí maskování při daném testovacím vektoru, ale existuje jiný testovací vektor, který danou poruchu pokryje. Druhý typ maskování do obvodu zavádí dodatečnou redundanci, danou poruchu již nelze otestovat, stala se z ní nedetekovatelná porucha. Pro oba typy maskování existuje mnoho způsobů, jak se jim vyhnout konstrukcí kompaktoru [1]–[5].

V dynamickém kompaktoru maskování nastává, když porucha f_2, která byla detekována již dříve, vyvolá odezvu na testovací vektor pro poruchu f_1 takovou, že vnitřní stav dynamického kompaktoru se překlopí do stavu stejného, jako je stav, ve kterém by se nacházel pro neporuchový obvod. V tomto kompaktoru je předcházení maskování obtížnější, protože již nezáleží na jednom testovacím vektoru, ale na celé testovací sekvenci. I zde existuje celá řada postupů, jak maskování omezit či zcela potlačit. [6]–[11]

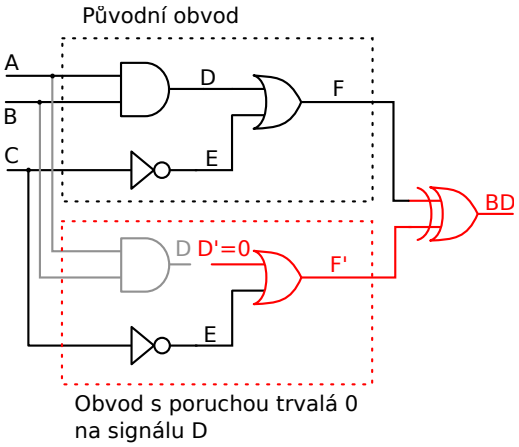
Tyto metody mají vesměs společné to, že vyžadují návrh nového či speciálního kompaktoru. Lze je jen těžko, nebo vůbec, použít na konkrétní, daný kompaktor.

Zde uvedené reference jsou staršího data, ač jsme se snažili, nenašli jsme novější relevantní práce, které neopakovaly fakta a experimenty z dřívějších prací.

C. Dosavadní výstupy

1) Rekapitulace předchozích výstupů: V předchozím ročníku semináře PAD jsem prezentoval dva články, které jsem sepsal během prvního ročníku studia [12]. První článek „SAT-ATPG for Application-Oriented FPGA Testing“ [13] jsem prezentoval na konferenci BEC 2016. Druhý článek „On Properties of ATPG SAT Instances“ jsem odeslal na konferenci DSD 2016, kde nebyl přijat.

2) SAT-based ATPG for Zero-Aliasing Compaction: Pojednává o generování testu s nulovým maskováním poruch v obecném dynamickém kompaktoru. Pro speciální případ kompaktoru, MISR založený na LFSR, experimentálně srovnává pokrytí a maskování vygenerovaného testu.



Obrázek 1: Příklad konceptuálního obvodu pro detekci poruchy trvalá 0. Tento obvod vychází z metod booleovské diference (BD).

II. Generování testu s nulovým maskováním

Jak je uvedeno výše, současné postupy potlačení maskování vyžadují manipulaci s návrhem kompaktoru. Náš přístup umožňuje potlačit maskování pro obecný kompaktor, bez zásahu do jeho architektury a bez zvětšení plochy obvodu.

Potlačení maskování realizujeme pečlivým výběrem testovacích vektorů. Protože je maskování způsobeno nešťastnou odezvou obvodu, nikoli samotným testovacím vektorem, je nutné omezit možné odezvy, tedy mít omezení na výstupech.

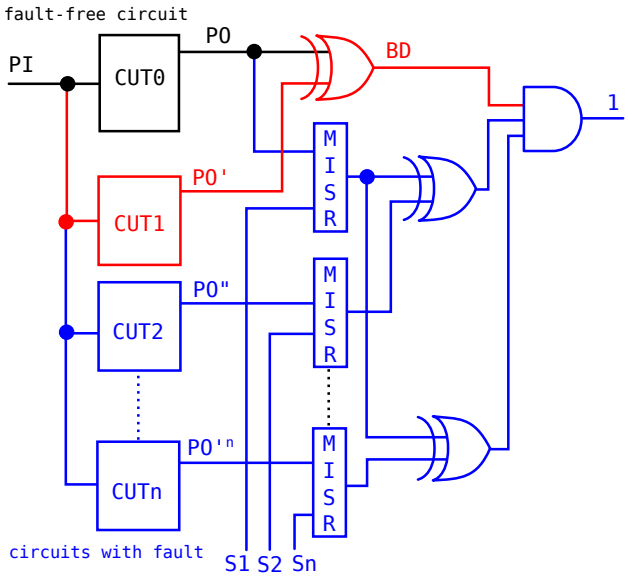
Z toho plyne nevhodnost moderních strukturních ATPG, neboť ty hledají testovací vektory od primárních vstupů (PI). Jakékoli omezení výstupů by se tedy projevilo hluboko v rozhodovacím stromě ATPG algoritmu.

Naopak ATPG založená na řešení problému Booleovské formule (SAT) tímto neduhem netrpí. Přestože SAT patří mezi NP-těžké kombinatorické problémy, moderní SAT řešiče, jako je např. MiniSAT [14], dokáží řešit i obtížné instance pocházející z ATPG procesu efektivně [15]–[17].

A. SAT ATPG

SAT ATPG převádí problém nalezení testovacího vektoru na problém splnitelnosti Booleovské formule. Jedná se o přístup podobný Booleovské diferenci, modelujeme obvod bez a s poruchou, jejich vstupy spojíme, jejich výstupy kombinujeme funkcí XOR, tím vznikne koncepční obvod, příklad je vyobrazen na obrázku 1. Porucha je detekována právě tehdy, když se odezva obvodu s poruchou liší od odezvy obvodu bez poruchy, tedy XORovaný výstup je v logické hodnotě 1 [18].

Tento koncepční obvod je poté převeden Tseitinovou transformací [19] na Booleovskou formuli v konjunktivní normální formě (CNF). Tato formule je dále zpracována SATovým řešičem, který nalezne ohodnocení odpovídající testovacímu vektoru. Takové ohodnocení existuje právě tehdy, když je porucha detekovatelná.



Obrázek 2: Konceptuální obvod pro obecný kompaktor. Bloky *MISR* jsou rozbalené kompaktory, vektory S_1–S_n představují předchozí stav kompaktoru.

B. Omezení testovacích vektorů

Na obrázku 2 je k vidění rozšířený konceptuální obvod. Obvody *CUT0* reprezentující původní testovaný obvod a *CUT1* modelující poruchu f_1, pro kterou hledáme testovací vektor. K tomuto klasickému obvodu dále přidáváme obvody *CUT2* až *CUTn*, které modelují poruchy f_2–f_n. Jedná se o poruchy, které byly otestovány dříve jiným testovacím vektorem, a pro které chceme zaručit, že nenastane maskování po aplikaci právě generovaného vektoru.

Protože maskování nastane až *po aplikaci* vektoru, který není ještě vygenerován, musíme předvídat budoucí stav kompaktoru. Toho dosáhneme rozbalením kombinační části kompaktoru a zpracováním odezvy obvodů *CUT2–CUTn* těmito kompaktory. Dále potřebujeme znát stav kompaktoru pro každou poruchu a pro bezporuchový obvod po aplikaci dosud nalezených testovacích vektorů, ty jsou reprezentovány vektory S_1–S_n. Budoucí stavy kompaktorů pro nemaskované poruchy se musí lišit od stavu bezporuchového obvodu, toto omezení je zaručeno XORováním výstupů rozbalených kompaktorů.

C. Algoritmus

Algoritmus iterativně prochází seznamem nepokrytých poruch, pro které se pokouší nalézt testovací vektor. Pro vygenerovaný vektor se provede simulace, včetně simulace kompaktoru, pro všechny poruchy. Jsou identifikovány poruchy, které byly detekovány i které byly maskovány. Pokud počet nově detekovaných poruch je větší než počet maskovaných a zároveň počet maskovaných poruch je menší než volitelný parametr M, je vektor přijat.

V případě nepřijetí testovacího vektoru jsou do procesu generování přidána omezení na nulové maskování poruch

maskovaných nepřijatým vektorem. Celý proces generování a simulace je poté opakován, dokud není nalezen vyhovující vektor, nebo již žádný vektor splňující všechna přidaná omezení neexistuje.

Celý algoritmus končí v okamžiku, kdy se pro žádnou nedetekovanou poruchu nepodaří najít vyhovující testovací vektor.

Tento algoritmus může vést k testu, který nepokrývá všechny poruchy. To ovšem nevadí ve smyslu, že přidání testovacích vektorů pro nedetekované poruchy by vedlo k maskování většímu, než je počet nově pokrytých poruch, tedy ke snížení pokrytí, pokud detekci měříme až z otisku obvodu v kompaktoru na konci testu, nikoli na testovacích výstupech obvodu.

D. Experimentální vyhodnocení

1) Popis experimentu: Tento algoritmus jsme experimentálně vyhodnotili na vybraných testovacích obvodech ze sady ISCAS'85 [20] a ITC'99 [21].

Jako dynamický kompaktor jsme použili lineární zpětnovazební registr s paralelními vstupy (MISR). Pro každý obvod jsme uvažovali několik velikostí tohoto kompaktoru, vždy jsme použili primitivní charakteristický mnohočlen. Velikost MISRu určujeme jako velikost vnitřního stavu, tedy počet bitů či počet klopných obvodů.

Použitý statický kompaktor byl paritní les, neboli několik paritních stromů (strom XOR hradel). Tyto paritní stromy mají disjunktní vstupy. Celý statický kompaktor má nulové maskování druhého typu, což znamená, že neomezuje množinu testovatelných poruch. Pro účely experimentu byl tento kompaktor součástí obvodu a testovací vektory byly generovány i pro poruchy v kompaktoru.

V experimentu porovnáváme maskování a pokrytí poruch mezi ZATPG a běžným ATPG, které při generování testu nebere v potaz kompaktor. Pro snadnější porovnání různých obvodů bereme v potaz pouze poruchy, které jsou testovatelné. To znamená, že pro každý obvod lze najít test s úplným pokrytím. Jako poruchový model jsme zvolili jednu poruchu trvalá 0/1.

Pro ZATPG i běžné ATPG používáme ATPG založené na SATu [13]. Pro řešení SATu používáme MiniSAT [14].

2) Pokrytí poruch: Jak je ilustrováno v tabulce I, pokrytí dosažené naším přístupem je pro kompaktory velmi malých rozměrů ($s < 4$) srovnatelné s běžným ATPG, ač je mírně nižší. To si vysvětlujeme ukončovací podmínkou běhu ZATPG, kdy pro některé poruchy není vygenerován testovací vektor.

Pokrytí dosažené s většími kompaktory je ovšem vyšší pro ZATPG. Zde jsou výsledky lepší, protože generování testovacích vektorů je vedeno směrem k nižšímu maskování, kdežto u běžného ATPG je maskování náhodné, dané pravděpodobností maskování v kompaktoru.

Prázdné buňky v tabulce I indikují obvody, pro které jsme neměli k dispozici statické kompaktory požadované velikosti.

3) Velikost kompaktoru s nulovým maskováním: Nulového maskování a úplného pokrytí dosáhne ZATPG pro kompaktory menších velikostí, než běžné ATPG. Tento rozdíl je k vidění v tabulce II.

E. Diskuse

Ač ZATPG dosahuje lehce nižšího pokrytí pro velmi malé kompaktory, než klasické ATPG, u větších kompaktorů naopak dosahuje lepšího pokrytí. Úplného pokrytí dosáhne u menších kompaktorů, než ATPG.

Jedním z problémů tohoto algoritmu je jeho nízká robustnost, která platí i pro běžné ATPG. Robustnost by mohlo jít zvětšit vytvořením heuristiky pro řazení poruch a pro selekci poruch bez maskování. Tyto heuristiky jsou jedním z možných budoucích vylepšení algoritmu.

III. Další směřování výzkumu a cíle disertační práce

Další směřování mého výzkumu jakož i cíle mé disertační práce vidím nadále v oblasti generování testu (ATPG) a prostředků vestavěné diagnostiky (BIST). Konkrétně se plánuji zaměřit na následující témata:

A. Omezení ATPG s ohledem na HW generátory

Přestože ZATPG je zajímavým výsledkem, má jeden problém, který jsem záměrně opomíjel. Tímto problémem je generování testovacích vektorů v BIST architektuře. Testovací posloupnost ze ZATPG lze uložit v paměti, aplikovat z ATE, nebo generovat složitým obvodem.

Jedno téma, kterým bych se tedy mohl zabývat, je další omezení ATPG takové, které by vedlo na testovací sekvenci snadno implementovatelnou v logickém obvodu.

B. Komutativní dynamické kompaktory

Velké omezení, na které jsem se ZATPG narazil, je závislost maskování na pořadí testovacích vektorů. Vzhledem k principu generování testovacích vektorů v ZATPG již s testovací posloupností nelze hýbat. To nejenže ztěžuje implementaci TPG v logickém obvodu, ale vylučuje jakékoli další zpracování testu, např. kompakci testovacích vektorů.

Za pozornost by mohlo stát využití komutativních kompaktorů, tedy takových, kde nezáleží na pořadí testovacích vektorů, resp. odezev testovaného obvodu.

C. Nelineární kompaktory

Využití nelineárních kompaktorů a prozkoumání jejich chování v součinnosti s ZATPG je taktéž možný směr výzkumu.

D. Periodické odezvy

Byly popsány metody, jak dosáhnout nulového (nebo sníženého) maskování za pomoci generování periodických odezev. Bylo by zajímavé zkusit upravit ZATPG pro generování testu s periodickou odezvou.

Tabulka I: Pokrytí poruch pro klasický ATPG a ZATPG

délka MISRu [b]		2		3		4		5		6		7		8	
obvod	poruchy	atpg [%]	zatpg [%]	atpg [%]	zatpg [%]	atpg [%]	zatpg [%]	atpg [%]	zatpg [%]	atpg [%]	zatpg [%]	atpg [%]	zatpg [%]	atpg [%]	zatpg [%]
b04	2846	78.71	79.20	91.10	87.80	94.65	95.11	97.99	97.89	98.80	99.68	99.54	99.96	99.86	99.96
b11	2382	78.30	75.99	89.37	92.02	94.58	96.09	98.02	98.86	98.78	99.71	99.41	99.96	99.87	100
c499	970	82.37	70.10	90.70	86.78	95.65	77.23	97.82	90.87	98.86	96.57	98.85	99.90	99.79	100
c880	1582	79.33	83.19	90.95	92.34	93.79	98.86	97.65	100	98.48	100	99.30	100	99.24	100
c1355	2618	79.76	75.63	90.56	86.05	95.45	92.85	97.01	97.09	99.27	100	99.65	99.73	99.77	100
c1908	2581	78.61	72.99	92.25	84.41	96.39	89.99	98.29	94.95	99.03	98.33	99.69	99.88	99.65	99.96
c2670	3613	79.91	73.04	92.69	90.78	94.54	93.85	98.11	98.45	98.61	98.00	99.83	100	99.53	100
c5315	7964			89.45	90.29	94.75	95.01	97.01	98.52	98.53	99.90	99.57	99.82	99.90	100
c7552	10921			90.62	88.42	94.85	92.97	97.55	96.35	98.28	98.94	99.32	99.95	99.65	100
průměr		79.57	75.73	90.85	88.77	94.96	92.44	97.72	97.00	98.74	99.01	99.46	99.91	99.70	99.99

Tabulka II: Nejmenší kompaktor s nulovým maskováním

		délka MISRu [b]		pokrytí [%]	
obvod	poruchy	ATPG	ZATPG	ATPG	ZATPG
b04	2846	11	9	100	100
b11	2788	10	8	100	100
c499	970	12	8	100	100
c880	1582	12	5	100	100
c1355	2618	10	6	100	100
c1908	2581	12	9	100	100
c2670	3613	12	7	100	100
c5315	7964	14	8	100	100
c7522	10921	12	8	100	100

E. Shrnutí

Jako jeden cíl své disertační práce si stanovuji rozšířit algoritmus popsaný v tomto článku o schopnost generovat test, který bude snadno implementovatelný v HW generátoru testovacích vektorů pro obvody BIST (bod III-A).

Další cíl, který si stanovuji je analýza představeného algoritmu při použití s jinými typy kompaktorů. Případně návrh nových kompaktorů, které by mohly lépe využít vlastností ZATPG (body III-B – III-D).

PODĚKOVÁNÍ

Autor děkuje za poskytnuté výpočetní a úložné zdroje programu Metacentra CESNET LM2015042 a CERIT-CS CZ.1.05/3.2.00/08.0144.

Tento výzkum byl částečně podporován z projektu ČVUT SGS17/213/OHK3/3T/18.

Tento výzkum byl částečně podporován z grantu české grantové agentury GA16-05179S.

LITERATURA

[1] K. Chakrabarty, "Zero-aliasing space compaction using linear compactors with bounded overhead," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 17, no. 5, pp. 452–457, 08 2002.

[2] S. R. Das, M. Sudarma, M. H. Assaf, E. M. Petriu, W. B. Jone, K. Chakrabarty, and M. Sahinoglu, "Parity bit signature in response data compaction and built-in self-testing of VLSI circuits with nonexhaustive test sets," *IEEE Transactions on Instrumentation and Measurement*, vol. 52, no. 5, pp. 1363–1380, Oct 2003.

[3] K. Chakrabarty and J. P. Hayes, "Test response compaction using multiplexed parity trees," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 15, no. 11, pp. 1399–1408, Nov 1996.

[4] B. Pouya and A. Touba, Nur, "Synthesis of Zero-Aliasing Elementary-Tree Space Compactors," in *IEEE VLSI Test Symposium*, 1998, pp. 70–77.

[5] Y. Liu and A. Cui, "An Efficient Zero-Aliasing Space Compactor Based on Elementary Gates Combined with XOR Gates," in *IEEE/ACM International Conference on Computer-Aided Design*, 11 2013, pp. 95–100.

[6] K. Pradhan, D., M. Reddy, Sudhakar, and K. Gupta, Sandeep, "Zero aliasing compression," in *Fault-Tolerant Computing: 20th International Symposium*, 06 1990, pp. 254–263.

[7] G. Edirisooriya and P. Robinson, John, "Test Generation to Minimize Error Masking," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 12, no. 4, pp. 540–549, 04 1993.

[8] T. Bogue, M. Gossel, H. Jurgensen, and Y. Zorian, "Built-in self-test with an alternating output," in *Proceedings Design, Automation and Test in Europe*, Feb 1998, pp. 180–184.

[9] G. Edirisooriya, P. Robinson, John, and S. Edirisooriya, "On the performance of augmented signature testing," in *IEEE International Symposium on Circuits and Systems*, 05 1993, pp. 1607–1610.

[10] K. Pradhan, D. and K. Gupta, Sandeep, "A new framework for designing and analyzing BIST techniques and zero aliasing compression," *IEEE Transactions on Computers*, vol. 40, no. 6, pp. 743–763, 1991.

[11] M. Kopec, "Can nonlinear compactors be better than linear ones?" *IEEE Transactions on Computers*, vol. 44, no. 11, pp. 1275–1282, Nov 1995.

[12] R. Hülle, "Generování testu pro prostředky vestavěné diagnostiky," in *14. Pracovní seminář Počítačové architektury a diagnostika*, 9 2016.

[13] R. Hülle, P. Fišer, J. Schmidt, and J. Borecký, "SAT-ATPG for Application-Oriented FPGA Testing," in *15th Biennial Baltic Electronics Conference*, 10 2016, pp. 83–86.

[14] N. Eén and N. Sörensson, "An Extensible SAT-solver," in *Theory and Applications of Satisfiability Testing*, ser. Lecture Notes in Computer Science, E. Giunchiglia and A. Tacchella, Eds. Springer Berlin Heidelberg, 2004, vol. 2919, pp. 502–518.

[15] M. Prasad, P. Chong, and K. Keutzer, "Why is ATPG easy?" in *36th Design Automation Conference*, 1999, pp. 22–28.

[16] R. Monasson, R. Zecchina, S. Kirkpatrick, B. Selman, and L. Troyansky, "2+p-SAT: relation of typical-case complexity to the nature of the phase transition," *Random Structures & Algorithms*, vol. 15, no. 3-4, pp. 414–435, 1999.

[17] S. Eggersglüß, R. Krenz-Baath, A. Glowatz, F. Hapke, and R. Drechsler, "A new SAT-based ATPG for generating highly compacted test sets," in *15th IEEE Design and Diagnostics of Electronic Circuits and Systems*, April 2012, pp. 230–235.

[18] T. Larrabee, "Test pattern generation using Boolean satisfiability," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 11, no. 1, pp. 4–15, Jan. 1992.

[19] G. Tseitin, "On the Complexity of Derivation in Propositional Calculus," in *Automation of Reasoning*, ser. Symbolic Computation, J. Siekmann and G. Wrightson, Eds. Springer Berlin Heidelberg, 1983, pp. 466–483.

[20] F. Brglez and H. Fujiwara, "A Neutral Netlist of 10 Combinational Benchmark Circuits and a Target Translator in Fortran," in *IEEE International Symposium Circuits and Systems (ISCAS'85)*. IEEE Press, Piscataway, N.J., 1985, pp. 677–692.

[21] F. Corno, M. S. Reorda, and G. Squillero, "RT-level ITC'99 benchmarks and rst ATPG results," in *IEEE Design Test of Computers*, Jul 2000, pp. 44–53 vol.17.