Fişa suspiciunii de plagiat / Sheet of plagiarism's suspicion	Indexat la: 24/06

Opera suspicionată (OS) Suspicious work	Opera autentică (OA) Authentic work

OS	Stanciu, A., „Autocontrolul ca mijloc de creştere a fiabilităţii şi disponibilităţii sistemelor de calcul", *An. Univ."Aurel Vlaicu" Arad, Fasc. Electrotehnică, Electronică, Automatizări*, p.257-272, 2000.
OA	Morun, C., „Creşterea fiabilităţii la transmiterea şi stocarea informaţiei." Teza de doctorat, Universitatea Tehnică din Timişoara, 1998.

Incidența minimă a suspiciunii / Minimum incidence of suspicion	
p.28:28 – p.40:01	p.257:12 – p.270:16

Fişa întocmită pentru includerea suspiciunii în Indexul Operelor Plagiate în România de la www.plagiate.ro

UNIVERSITATEA POLITEHNICA TIMIŞOARA
Facultatea de Automatică şi Calculatoare

TEZĂ DE DOCTORAT

Creşterea fiabilităţii la transmiterea şi stocarea informaţiei

Conducător ştiinţific
Prof.dr.ing.Mircea Vlăduţiu

Doctorand
Ing. Cezar Morun

1998

CUPRINS

1 METODE DE CREŞTERE A FIABILITĂŢII ŞI DISPONIBILITĂŢII

1.1 Indicatori pentru caracterizarea fiabilităţii şi disponibilităţii

1.1.1 Clasificarea defecţiunilor

Într-un sistem există trei categorii majore de surse de erori:

greşeli de proiectare;

- defecţiuni fizice;
- erori de interacţiune ale operatorului uman *(procedural errors)*.

Aceste surse de erori contribuie la căderile sistemului. Pe de o parte, greşelile de proiectare apar atât în *hardware*, cât şi în *software*. Acestea din urmă sunt predominante şi se elimină greu din sistem [Toy88]. Pe de altă parte, defecţiunile fizice sunt rezultatul îmbătrânirii componentelor sau a influenţei mediului, care cauzează devierea unor caracteristici ale echipamentului peste limitele câmpului specificat al toleranţei. Se produc, astfel, erori determinate de aşa-numitele defecţiuni parametrice *hardware*.

Acţiunile improprii ale operatorului la panourile de control şi de întreţinere pot determina funcţionarea defectuoasă a sistemului. Ceea ce introduce operatorul are prioritate mare, sistemele fiind foarte vulnerabile la comenzi improprii şi la erori de operare. Proiectarea unui sistem tolerant la defecţiuni necesită găsirea unei metode care să împiedice ca o componentă *hardware* deteriorată să determine o defecţiune logică şi ca aceasta să determine, la rândul ei, o eroare.

În relativ scurta istorie a designului tolerant la defecte au existat multe confuzii între definiţii şi concepte fundamentale. De exemplu, se foloseau, în mod interschimbabil, termeni ca: *defecţiune, deteriorare* şi *eroare*. Unii autori susţin că o deteriorare a avut loc atunci când calculatorul nu mai răspunde la comenzi. Dimpotrivă, alţi autori afirmă că o deteriorare este o defecţiune fizică specifică, mai degrabă, componentelor electronice. Datorită diversităţii de păreri în privinţa înţelegerii acestor noţiuni şi pentru a se evita echivocul în lectura acestei lucrări, consider utilă definirea unor termeni, care, deşi au serioase trimiteri bibliografice - [Lala85] şi [John89] -, sunt încă departe de acceptarea universală. Aceste definiri sunt:

- Deteriorarea *(failure)* apare atunci când ieşirea obţinută diferă de cea proiectată. Ieşirea poate fi privită la diferite niveluri de abstractizare: la nivel de componentă electronică, la nivel de bloc sau la nivel de sistem;
- Defecţiunea sau defectul *(fault)* este o stare eronată a maşinii sau a sistemului de programe, stare generată de: deteriorări ale componentelor, interferenţe fizice cu mediul, erori ale operatorului uman sau proiectare incorectă;
- Eroarea *(error)* este manifestarea unei defecţiuni într-un program sau într-o structură de date. Ea poate apărea la o anumită distanţă în spaţiu şi timp faţă de locul, respectiv momentul, defectării;

Definiţiile de mai sus pot fi caracterizate de următoarele adjective:

- Adjectivul „permanent“ (idem în engleză) descrie o deteriorare, o defecţiune sau o eroare cu caracter continuu şi stabil. Dacă este vorba despre echipament, atunci o deteriorare permanentă reflectă o schimbare fizică ireversibilă;
- Adjectivul „intermitent“ *(intermittent)* descrie o defecţiune sau o eroare care se repetă ocazional;
- Adjectivul „tranzitoriu“ *(transient)* descrie o defecţiune sau o eroare non-repetitivă, cauzată de condiţii temporare. Exemple sunt acţiunea particulelor α asupra memoriilor semiconductoare sau variaţia tensiunii de alimentare a echipamentelor. Astfel de erori sunt nereparabile, pentru că nu există defect fizic al echipamentului. Evitarea lor se poate face doar printr-o proiectare mai riguroasă.

Deteriorările sistemului sunt clasificate, funcţional vorbind, în defecţiuni *hardware*, erori *software* şi erori de procedură. Există relaţii strânse între cele trei tipuri de deteriorări de sistem şi categoriile surselor de defecţiuni. Defecţiunile fizice apar numai la echipament. Erorile de proiectare, chiar dacă apar în *hardware*, produc, în majoritatea lor, erori de tip *software*. Erorile de interacţiune sunt cauzate, de obicei, de greşelile făcute de operator.

În afara deteriorărilor normale ale dispozitivelor, la defecţiunile *hardware* mai contribuie condiţiile mediului înconjurător (perturbaţiile), erorile de proiectare (greşelile) şi cronografiile improprii *(timing problems)*. Greşelile de proiectare, ca în cazul erorilor software, sunt foarte greu de apreciat cantitativ. Calculele de fiabilitate se bazează mai mult pe defecţiunile dispozitivelor.

1.1.2 Estimarea fiabilităţii

Estimarea fiabilităţii este un proces de apreciere a fiabilităţii realizabile de un articol (element, subsistem sau sistem), având disponibile datele ratei de defectare, adică, prin estimarea fiabilităţii, se apreciază probabilitatea îndeplinirii obiectivelor articolului respectiv pentru o aplicaţie specifică. Aceste calcule sunt utile în stadiile de început ale unui proiect.

După îndepărtarea defecţiunilor timpurii, componentele se menţin pe o perioadă lungă la o rată de defectare aproximativ constantă. În timpul acestei perioade, rata de defectare este, de obicei, scăzută şi este puţin probabil ca defecţiunile să provină de la o singură cauză. Rata constantă de defectare, reprezentată în porţiunea de viaţă utilă, scoate în evidenţă faptul că probabilitatea de defectare este independentă de vârstă. Pentru orice rată constantă de defectare, valoarea fiabilităţii depinde numai de timp. Funcţia fiabilităţii $R(t)$ (*reliability*), care este caracterizată de o rată constantă de defectare este o distribuţie negativă exponenţială şi are forma:

$R(t) = e^{-\lambda t}$, unde λ este rata de defectare, iar t este timpul.

Timpul mediu dintre defectări (*Mean Time Between Failures*; MTBF) este timpul mediu, exprimat, de obicei, în ore, în care un articol ar putea să funcţioneze corect înainte de a se deteriora şi reprezintă măsura cantitativă a fiabilităţii [Lala85], [Toy88]. În general, MTBF-ul unui sistem ar putea fi tratat ca integrala funcţiei de fiabilitate a întregului sistem: $\int_0^\infty R(t)\,\mathrm{d}\,t$

În câteva cărţi de literatură, intervalul este definit ca timp mediu de defectare (*Mean Time To Failure*; MTTF). Din punct de vedere tehnic, MTTF şi MTBF nu sunt identici, MTBF-ul poate fi definit, într-o formă alternativă, astfel:

$MTBF = MTTF + MTTR$, unde MTTR este timpul mediu de reparaţie *(Mean Time To Repair)*.

Rata de reparaţie μ *(repair rate)* sau inversul timpului mediu de reparaţie, adică

$$\mu = \frac{1}{MTTR}$$

este un alt factor care afectează substanţial fiabilitatea şi mentenabilitatea unui sistem. Când o unitate dintr-un sistem duplicat este defectă, sistemul este dependent de a doua unitate pentru a-şi

continua operaţia. Dacă unitatea defectă este reparată repede, riscul deteriorării întregului sistemului este mic deoarece unitatea a doua va opera astfel încât să păstreze integritatea funcţionării sistemului. Întrucât sistemul este vulnerabil numai până la reparaţia unităţii defecte, un timp scurt de reparaţie poate creşte foarte mult fiabilitatea sistemului.

În cazul RAID, pentru estimarea fiabilităţii, nu mai sunt suficienţi MTBF, MTTF şi MTTR, parametri care descriu cu precădere caracteristicile fizice ale dispozitivelor. Astfel, ca de curând RAB [Mass97] a introdus:

- **MTDA** – *Mean Time until (Loss of) Data Availability,* timpul mediu până ce căderea unei componente cauzează o pierdere a accesibilităţii într-o matrice de discuri. Pierderea accesibilităţii nu implică pierderea datelor (pentru multe tipuri de căderi datele rămânând intacte – ex. Căderea unui controler neredundant). Evident că MTDA se poate folosi şi pentru RAIC, la fel ca şi MTDL.

  MTDL – *Mean Time until Data Loss,* timpul mediu până ce căderea unei componente generează o pierdere permanentă de date într-o matrice de discuri. Conceptual, MTDL este similar cu MTBF, dar ţine cont de posibilitatea ca redundanţa RAID-ului să protejeze împotriva pierderii de pentru un număr limitat de căderi de componente.

În prezenta teză de doctorat, autorul introduce trei noi indicatori necesari pentru evaluarea fiabilităţii, pentru RAID 7 şi RAIC 7:

MTDR – *Mean Time until Data is Recovered,* timpul mediu de la apariţia unei căderi de disc până când sunt regenerate toate blocurile pierdute în spaţiile de rezervă

MTRD – *Mean Time until Data is Regenerated,* timpul mediu în care RAID sau RAIC răspunde la o cerere dintr-o zonă afectată, prin regenerarea datelor utilizând redundanţa.

MTAR – *Mean Time until Array is Reconstructed,* timpul mediu de reconstrucţie (în starea iniţială) a matricei după înlocuirea discului sau canalului căzut.

1.1.3 Disponibilitatea

Disponibilitatea *A(t) (availability)* unui echipament este o funcţie de timp. Ea se poate defini ca probabilitate de funcţionare a echipamentului la momentul t, atât timp cât este folosit conform specificaţiilor.

Conceptul de disponibilitate este utilizat în măsurarea eficacităţii sistemului. Atât fiabilitatea, cât şi mentenabilitatea, îşi aduc aportul la definirea conceptului de disponibilitate. Această conexiune poate fi exprimată în felul următor [Lala85], [Toy88]:

$$A(t) = \frac{MTTF}{MTTF + MTTR}$$

unde funcţia fiabilităţii se presupune că este distribuţia exponenţială $R = e^{-\lambda t}$. Deoarece MTTF-ul este o măsură a fiabilităţii, iar MTTR-ul este o măsură a mentenabilităţii, ele se pot negocia astfel încât să se obţină o disponibilitate dată.

1.1.4 Dependabilitatea

În 1982, Laprie şi Coste au definit formal dependabilitatea ca un cadru de specificaţii de nivel înalt, care include mărimi ca fiabilitatea şi disponibilitatea, ca două faţete distincte ale specificaţiilor sistemului. Pe baza caracteristicilor unui sistem, dependabilitatea lui se descrie fie prin fiabilitate, fie prin disponibilitate. De exemplu, putem caracteriza mai bine un sistem nereparabil prin fiabilitatea lui, iar un sistem reparabil prin disponibilitatea lui.

În literatura mai recentă [John89], se menţionează că termenul de dependabilitate include, pe lângă conceptele de fiabilitate şi disponibilitate menţionate anterior, şi pe cele de siguranţă în funcţionare, mentenabilitate, performabilitate şi testabilitate. Dependabilitatea reprezintă calitatea deservirii furnizate de către un anumit sistem. Fiabilitatea, disponibilitatea, siguranţa, mentenabilitatea, performabilitatea şi, în fine, testabilitatea sunt mărimi utilizate pentru estimarea cantitativă a dependabilităţii unui sistem. Ţinta proiectării tolerante la defecţiuni este îmbunătăţirea dependabilitaţii prin înarmarea unui sistem pentru realizarea funcţiei cerute, în prezenţa unui număr dat de defecte. Totuşi, este de notat că un sistem tolerant la defecţiuni nu este neapărat de înaltă dependabilitate şi nici înalta dependabilitate nu necesită neapărat toleranţa la defectare.

1.1.5 Mentenabilitate

Sistemele de calcul, ca orice alte produse, se pot clasifica în două categorii: a) sistemele cu funcţie unică (simplă), la care prima defectare constituie şi finalul vieţii lor şi b) sistemele cu funcţie repetată sau sistemele cu reînnoire (restabilire), la care elementele defecte pot fi înlocuite cu altele bune, caz în care aceste produse au caracter reparabil. Prin mentenanţă, se înţelege ansamblul tuturor acţiunilor tehnico-organizatorice necesare, efectuate în scopul menţinerii sau restabilirii

unui produs în starea necesară îndeplinirii funcţiei cerute. Aceste acţiuni pot avea caracter corectiv (depistarea cauzei unei defecţiuni, repararea defectului prin înlocuirea elementelor defecte, verificarea corectitudinii operaţiilor de mentenanţă întreprinse) sau caracter preventiv (revizii, reglaje, verificări şi reparaţii planificate, executate în vederea evitării unor viitoare defecţiuni).

În aceste condiţii, este necesar să se exprime cantitativ aptitudinea unui produs de a fi repus în funcţiune în urma unui defect. Exprimarea se face, ca şi în cazul fiabilităţii, printr-o probabilitate în funcţie de timp. Mentenabilitatea este, aşadar, aptitudinea unui produs ca, în condiţii date, să fie menţinut sau restabilit în starea de a-şi îndeplini funcţia specificată, atunci când acţiunile de mentenanţă se efectuează în condiţii precizate şi într-un timp dat, cu procedee şi remedii prescrise. Corespunzător acestei definiţii, legătura dintre aspectul probabilistic şi cel funcţional se exprimă astfel:

$$M(t_r) = \text{Prob}(t_r \le T_r)$$

unde t_r este timpul de restabilire, T_r este o limită impusă duratei de restabilire, iar $M(t_r)$ este funcţia de mentenabilitate.

Relaţia între mentenabilitatea M şi rata de reparaţie μ sau timpul mediu de reparaţie MTTR este următoarea [Lala85]:

$$M(t_r) = 1 - e^{-(\mu \cdot t_r)} = 1 - e^{-\left(\frac{t_r}{MTTR}\right)}$$

La fel ca fiabilitatea, testabilitatea, dependabilitatea etc., mentenabilitatea trebuie planificată şi estimată sau evaluată încă din fazele cele mai timpurii ale conceperii unui produs.

1.1.6 Cerinţele fiabilităţii aplicate

Cerinţele fiabilităţii variază considerabil de la o aplicaţie la alta. De exemplu, obiectivele fiabilităţii unui oficiu telefonic cu comutare digitală, proiectat pentru aplicaţii telefonice, sunt:

- timpul total în care sistemul rămâne neoperativ să nu depăşească trei minute pe an, cu un timp mediu de reparaţie (MTTR) de patru ore;
- în timpul funcţionării să nu se piardă sau să nu se trateze incorect mai mult de 0,01% dintre apeluri.

Funcţionarea satisfăcătoare nu înseamnă, în acest caz, o fiabilitate de 100%. Sunt permise, totuşi, câteva apeluri incorecte, avându-se în vedere că utilizatorul va forma din nou numărul dorit, obţinând corecţia erorii. Pe de altă parte, o defecţiune într-un echipament critic, cum ar fi

transponderele de telecomunicaţii de pe sateliţii artificiali, ar determina căderea unor întregi sisteme, ceea ce este inadmisibil. În astfel de cazuri, funcţionarea satisfăcătoare impune o fiabilitate de 100%.

Durata timpului de funcţionare pentru un sistem de comutare telefonică este, pur şi simplu, durata de viaţă a echipamentului care a fost folosit în sistem. Sistemul de comutare trebuie să funcţioneze continuu, fără întreruperi, până când echipamentul este înlocuit la sfârşitul vieţii lui sau este îndepărtat pentru orice alt motiv. Deoarece servirea trebuie să fie asigurată 24 de ore pe zi, nu poate să existe timp programat pentru reparaţii sau pentru întreţinere, timp în care sistemul să fie neoperaţional. Obiectivul fiabilităţii de trei minute de neoperare pe an (două ore de nefuncţionare în patruzeci de ani [Prad86]) şi MTTR-ul de patru ore pot fi traduşi într-o disponibilitate de sistem de 99,9995%.

În contrast, aplicaţii critice pentru controlul avioanelor, ca şi în cazul lui SIFT *(Software-Implemented Fault Tolerance)* şi FTMP *(Fault-Tolerant MultiProcessor)* pretind fiabilitate ultra-înaltă. Parametrii proiectării cer ca sistemul să treacă prin mai puţin de 10^{-9} deteriorări în timpul unei misiuni de 10 ore cu un MTTR de 10 ore. Factorul de disponibilitate este egal, în această situaţie, cu 99,9999999%.

Cerinţele fiabilităţii determină într-un grad mai mare arhitectura sistemului, tipul şi cantitatea de redundanţă precum şi costul sistemului. Redundanţa dual ă se dovedeşte a fi adecvată aplicaţiilor telefonice. Pe de altă parte, pentru controlul critic al avioanelor, triplarea cu rezervă este necesară pentru a achita un grad mare de fiabilitate.

1.1.7 Tehnici de redundanţă

Defecţiunile *hardware* pot afecta secvenţele de comandă sau cuvintele de date care se află în interiorul maşinii. Acest lucru duce la două tipuri de erori:

- secvenţa programului este neschimbată, dar defecţiunea a afectat rezultatele finale;
- secvenţa programului este schimbată, iar programul nu mai execută algoritmul specificat.

Erorile *software* sunt rezultatul unor traduceri (translatări) greşite sau ale unor implementări improprii ale algoritmilor originali şi, de asemenea, deviază execuţia instrucţiunilor de la secvenţa corectă. În multe situaţii, din păcate, defecţiunile *hardware* nu pot fi distinse de cele *software*. Din acest motiv, sistemele trebuie să fie tolerante la defecţiuni, indiferent de sursa acestora.

O modalitate bună de a produce calculatoare tolerante la defecte este aceea de a introduce redundanţa prin multiplicarea părţilor lor. Redundanţa permite calculatoarelor să ocolească erorile, în felul acesta rezultatele fiind corecte. Această metodă este cunoscută ca redundanţă de protecţie, fiind compusă din redundanţe *hardware*, *software* şi de timp. Redundanţa *hardware* este constituită din componente adiţionale, care detectează şi corectează erorile. Redundanţa *software* conţine programe adiţionale, care au capacitatea de a restabili funcţionalitatea sistemului, în condiţii problematice. De asemenea, mai poate conţine programe de detecţie a erorilor, de diagnoză şi de autocontrol, prin care să testeze periodic toate circuitele logice ale calculatorului. Redundanţa temporală este obţinută prin repetarea unei operaţii eronate, imediat după repetarea unei erori. Redundanţa temporală se implementează adesea prin *hardware*. De exemplu, logica *hardware* poate iniţia recitirea automată a unei locaţii de memorie în care s-a detectat o eroare de paritate.

Chiar dacă redundanţa de protecţie este clasificată funcţional în trei tipuri diferite, în practică ele se contopesc adesea, pentru asigurarea unei fiabilităţi sporite. În cazul redundanţei *software*, programul de control are nevoie atât de spaţiu de memorie *(hardware)*, cât şi de timp de execuţie.

Fiecare dintre aceste tipuri de redundanţă, precum şi diferitele lor combinaţii, au fost folosite în proiectarea calculatoarelor tolerante la defecţiuni. Prioritatea acordată la implementare unuia dintre tipurile de redundanţă enumerate mai sus se face în funcţie de tipul aplicaţiei şi de restricţiile impuse de costuri, de cerinţele de timp şi de cerinţele de fiabilitate.

1.2 Metode de autocontrol, implementate *hardware*

Atât structura redundanţei dinamice cât şi cea a redundanţei statice, descrise mai sus, sunt metode care folosesc părţi de rezervă pentru a face sistemul capabil să tolereze defecţiunile. Atunci când se foloseşte redundanţa statică, unităţile de rezervă (circuite, componente sau subsisteme) sunt părţi permanent active ale sistemului. Ele corectează erorile sau le maschează împiedicând, astfel, propagarea lor în sistem. Funcţia mascării intervine automat, iar acţiunea de corectare este imediată şi încorporată *(wired in)*. Tipuri statice de redundanţă au fost folosite, mai întâi, în aplicaţiile militare care pretindeau o înaltă fiabilitate pentru o perioadă scurtă de timp, iar, mai recent, în aplicaţiile comerciale. Redundanţa dinamică, la care subsistemele adiţionale

- reiniţializarea *(reset*-ul).

1.3.5 Diagnosticarea erorii

Ţinând cont că detectarea erorii determină corectitudinea funcţionării unui circuit, diagnoza erorii localizează defecţiunea la o unitate înlocuibilă. Unitatea înlocuibilă poate fi o componentă, un circuit sau un subsistem. Rutina de diagnosticare a erorii utilizează *hardware* pentru detectarea erorii şi secvenţe de test în scopul localizării unităţii defecte. Dacă sursa erorii o constituie o singură entitate, atunci defecţiunea este corectată simplu, prin înlocuirea entităţii respective, fără să mai fie necesară diagnoza. Dacă circuitul de detecţie este mai puţin specializat, atunci rutina de diagnosticare a erorii poate fi solicitată la izolarea, în continuare, a unităţii afectate.

Cele mai dificile sarcini ale proiectării întreţinerii sunt: restabilirea sistemului şi diagnosticarea. Eficacitatea lor poate fi determinată prin simularea modului de comportare a sistemului în prezenţa unei anumite defecţiuni. Prin intermediul simulării deficienţele proiectării pot fi identificate şi corectate înainte de a se da sistemul spre utilizare. Este necesar să se evalueze abilitatea sistemului la detectarea erorilor, la revenirea automată la funcţionarea normală şi la furnizarea informaţiei de diagnoză (de exemplu: locaţia defecţiunii). Simularea, fizică sau digitală, a defecţiunii este un aspect important al proiectării întreţinerii.

1.3.6 Autocontrolul ca mijloc de creştere a fiabilităţii şi disponibilităţii sistemelor de calcul

1.3.6.1 Metode de autocontrol aplicate la diferite niveluri de structură

1.3.6.1.1 Caracteristicile metodelor de proiectare structurată la nivel de bistabil

Progresele recente din tehnologia VLSI au avut o influenţă mare asupra testării sistemelor digitale (numerice). Sistemele digitale de astăzi au de la o sută de mii până la un milion de porţi de logică aleatoare şi celule de memorie ceea ce face ca generarea testului şi simularea defecţiunilor să fie extrem de dificile. Chiar dacă se pot folosi maşini *hardware* dedicate sau super-computere pentru a genera teste eficace, costul acestora face imposibilă aplicarea lor în practică. Pentru a se depăşi această problemă, cercetarea s-a îndreptat către găsirea altor metode de testare a sistemelor digitale, incluzând proiectarea pentru testabilitate şi generarea de test la nivel funcţional (simularea defecţiunilor).

Dintre toate tehnicile propuse până în prezent, proiectarea pentru testabilitate (DFT, *Design For Testability*) este cea mai renumită, iar dintre toate tehnicile DFT, cea mai folosită este tehnica scanării *(scan-design)*. În tehnica scanării (în [Marc93] i se spune tehnică sau metodă SCAN), bistabilele sunt conectate într-un circuit serial şi sunt folosite ca terminale de tip I/O. Prin aplicarea tehnicii de scanare putem transforma un circuit secvenţial în unul combinaţional, făcând astfel mai simplă generarea *pattern*-urilor de test pentru circuit. De asemenea, se poate partiţiona logic circuitul în câteva subcircuite şi se pot genera, pentru fiecare în parte, *pattern*-urile de test.

Firma NEC a aplicat cu succes tehnica proiectării de scanare la sistemele comerciale de calcul încă din anul 1968. Tehnica numită cale de scanare *(scan-path)* a fost implementată de la nivel de capsulă la nivel de sistem şi a devenit o ustensilă puternică pentru testarea şi diagnoza sistemelor de calcul VLSI. Conceptul *scan in/scan out*, care stă la baza tehnicii de scanare, a fost introdus pentru prima dată în anul 1964 de către Carter s.a. pentru dezvoltarea testelor de localizare a defecţiunilor la sisteme IBM 360. NEC a dezvoltat metoda căii de scanare pentru a rezolva problemele din anii 70, cum ar fi utilizarea intensivă a MSI/LSI din sistemele comerciale de calcul, precum şi numărul mare de porţi aflate pe plăcile acestor sisteme.

Metoda căii de scanare pare să fie o soluţie promiţătoare pentru rezolvarea acestor probleme deoarece permite mai multă controlabilitate şi observabilitate asupra circuitului supus testării. În plus, se pot testa atât circuitele combinaţionale cât şi cele secvenţiale. Conceptele de controlabilitate şi observabilitate se definesc în felul următor [Lala85], [Prad86], [John89]:

- **Controlabilitatea** arată cât de uşor se poate produce un semnal arbitrar, valid la intrările unei componente (subcircuit) prin excitarea intrărilor primare ale circuitului. De fapt, ea reprezintă abilitatea de a comanda un semnal doar prin intermediul intrărilor primare. O linie care poate fi poziţionată pe „1“ logic se numeşte 1-controlabilă, altă linie care poate fi poziţionată pe „0“ logic se numeşte 0-controlabilă, iar linia care poate avea ambele stări logice se numeşte complet controlabilă;

  Observabilitatea arată cât de uşor se poate determina la ieşirile primare ale circuitului ceea ce se întâmplă la ieşirile unei componente (subcircuit). Cu alte cuvinte, ea este abilitatea activării unei căi de la semnalul cercetat până la un punct măsurabil.

Prin componente se înţeleg fie circuite integrate standard (SSI şi MSI) pentru circuite la nivel de placă, fie celule standard de module de bibliotecă pentru circuite LSI şi VLSI. De asemenea, se presupune că mai multe componente sunt conectate cu legături unidirecţionale.

Circuitul testat prin metoda căii de scanare poate fi partiţionat logic în mai multe subcircuite, iar programele de test pot fi generate, independent pentru fiecare subcircuit. Costul total al întregului circuit scade la α^2 / n, unde n este numărul de subcircuite, iar α este rata de suprapunere . Tabelul următor arată o comparaţie între mărime şi cost.

Denumire	Mărime	Cost
Circuit original	1	1
Subcircuit	α / n	$(\alpha / n)^2$
Circuit expandat	α	α^2 / n

Figura 1-1

Evident că α depinde de n şi, din acest motiv, proiectanţii trebuie să decidă cu atenţie numărul de subcircuite necesare. Pentru un circuit bine proiectat, care este pregătit pentru partiţionare, valoarea lui α este sub 1,7. Calea de scanare poate facilita localizarea defecţiunii pentru că procesorul de diagnoză poate să urmărească uşor starea internă a sistemului. *Hardware*-ul adiţional este foarte puţin (de la 4% la 10%) şi este potrivit pentru un calculator VLSI deoarece necesită relativ puţini pini adiţionali de I/O.

Bistabilul transformat cu cale de scanare, utilizat ca şi circuitul de bază, include două tipuri de semnale de *clock*, C_1 pentru operaţia normală şi C_2 pentru operaţia de deplasare. Astfel, bistabilele unei plăci logice sunt conectate în serie printr-o cale de scanare şi operează ca un registru serial de deplasare utilizând *Clock* II, intrarea de test *(scan in)* şi ieşirea de test *(scan out)*. Utilizând o adresă de selecţie a plăcii logice, putem selecta o anumită placă dintr-o unitate logică.

Chiar dacă tehnica scanării este o tehnică DFT puternică, pentru testarea logicii aleatoare, aplicarea ei poate pune probleme în diferite tipuri de circuite. Un astfel de tip sunt şirurile de memorie încorporate în circuite de logică aleatoare. Aplicarea directă a tehnicii de scanare la fiecare

celulă de memorie din şir costă foarte mult. O soluţie posibilă este aplicarea tehnicii de scanare la un anumit cuvânt din şirul de memorie şi accesarea cuvântului fixat în timpul testării circuitului. Oricum, din moment ce, în timpul testării, memoria funcţionează ca un registru, ea trebuie testată utilizând altă procedură.

Tehnica scanării s-a dovedit foarte eficace pentru proiectarea circuitelor bipolare, dar este greu de aplicat la un circuit MOS şi mai ales la un circuit MOS dinamic. Dificultatea apare, în primul rând, din cauza *hardware*-ului adiţional. Un bistabil MOS dinamic este limitat la câteva tranzistoare şi supraîncărcarea *hardware*-ului adiţional este mare pentru un circuit VLSI MOS.

În 1977, IBM introduce tehnica LSSD *(Level Sensitive Scan Design)* care include tehnica scanării şi, în plus, impune ca toate schimbările de stare să fie controlate de nivelul semnalului de ceas şi nu de front *(Level Sensitive Design)* [Russ85], [Lala85], [Yarm90]. Această abordare reduce dependenţa funcţionării de timpii de propagare, eliminând cursele sau hazardul. Celula de bază este elementul de memorare a informaţiei, dependent de nivel, care asigură şi tratarea semnalului de scan în modul test, element numit SRL *(Shift Register Latch)*.

1.3.6.1.2 Caracteristicile metodelor de proiectare structurată la nivel de registru

Deşi s-au scris foarte multe despre standardul de test *boundary-scan* (IEEE 1149.1 *Standard Test Acces Port and Boundary-Scan Architecture*), doar câtorva dispozitive li s-a implementat această arhitectură. Acordarea completă cu acest standard, ceea ce înseamnă că proiectanţii ar putea să folosească orice dispozitiv, cu asigurarea că este compatibil cu *boundary-scan*, va apare numai după câţiva ani. Până atunci, plăcile cu cablaj imprimat vor fi hibride, conţinând circuite (dispozitive) scanabile (după standardul IEEE) şi dispozitive nescanabile. Aceste plăci necesită metodologii de testare bazate pe ambele tehnici: pat de cuie *(bed of nails)* şi *boundary-scan*.

Testarea cu *boundary-scan* poate micşora drastic timpul de dezvoltare al sistemelor complexe. De asemenea, ea oferă posibilitatea testării sistemelor care utilizează componente cu montare la suprafaţă de mare densitate şi plăci multistrat complexe. Astăzi, aceste sisteme se pot testa cu ajutorul tehnicii patului de cuie, dar există probleme din cauza geometriei micşorate a pinilor, geometrie impusă de tehnologia montării la suprafaţă. Şi în acest caz, având unelte potrivite şi dispozitive de

susţinere pentru *boundary-scan*, toată placa poate fi testată complet, folosind numai calea *boundary-scan*.

Pe de altă parte, patul de cuie şi, în general, metodele de testare în circuit *(in-circuit)* oferă câteva avantaje importante spre deosebire de *boundary-scan*. Un astfel de avantaj cheie al testării în circuit este abilitatea diagnozei deteriorărilor multiple la o singură trecere prin test. În plus, detectarea scurt-circuitelor poate fi realizată înainte de punerea sub tensiune a plăcii, reducând posibilitatea deteriorării catastrofale cauzate de acestea între pini şi tensiunea de alimentare. De asemenea, testele în circuit permit realizarea simultană a testării parametrice cu cea digitală.

Boundary-scan, în general, nu se referă la testarea dispozitivelor analogice sau pasive. Contrar impresiei larg răspândite, *boundary-scan* este compatibil cu testarea în circuit şi folosind o combinaţie a celor două se poate simplifica semnificativ testarea plăcilor.

Dispozitivul XC4005 FPGA este primul dintr-o familie de circuite compatibile cu standardul *boundary-scan*. Fiecare pin de I/O al acestui dispozitiv poate fi complet controlat şi observat prin utilizarea unor *pattern*-uri de date introduse serial în registrul de *boundary-scan* al lui XC4005 prin pinul TDI *(Test Data Input)* şi prin controlul prin pinii TMS *(Test Mode Select)* şi TCK *(Test Clock)* proveniţi din canalele ATE *(Automatic Test Equipment)*. Acest lucru este echivalent cu stimularea complexă a intrărilor pentru a obţine aceeaşi acoperire de test fără *boundary-scan*.

1.3.6.1.3 Proiectarea structurată la nivel de bloc

Încorporarea facilităţilor de test (BIT, *Built-In Test*) poate micşora considerabil timpul de întreţinere şi costurile pentru sistemele electronice complexe prin accelerarea detecţiei şi izolării defecţiunilor. La nivel de placă, testul încorporat (BIT) se realizează, de obicei, prin aplicarea unui set de *pattern*-uri pseudo-aleatoare pentru a se verifica existenţa defecţiunilor şi prin comprimarea informaţiei răspunsului de test pentru a se obţine o semnătură. Aceste tehnici sunt eficace pentru plăcile utilizate în scopuri generale atât timp cât seturile de cipuri care se testează reacţionează la cele două tehnici, dând rar semnături identice pentru plăci bune sau defecte.

Notă: În [D&TR89], McCluskey precizează că nu este recomandată utilizarea expresiei *test-response compression* ca un sinonim al expresiei *test-response compaction*, deoarece prin compactare se pierde din informaţie, în timp ce prin comprimare se elimină doar redundanţa, fără