T S C
CURSUL 1

INTRODUCERE

Testare si Diagnoza. Definitii.

Testarea unui sistem este constituita dintr-un sir de experimente prin care se exercita respectivul sistem în asa fel încât raspunsul sistemului, ca urmare a sirului de experimente, sa fie suficient si necesar pentru a stabili daca acesta functioneaza corect sau nu.
In cazul în care se stabileste ca sistemul functioneaza incorect se poate formula o a doua cerinta a testarii: diagnoza; aceasta însemnând localizarea sau evidentierea cauzei ce a condus la respectiva malfunctionare a sistemului. Diagnoza presupune, evident, cunoasterea structurii interne a sistemului testat.

Testarea la diferite nivele de abstractizare.

Deoarece cele expuse anterior pot fi considerate ca fiind asertiuni valabile pentru testarea sistemelor în general, în cele ce urmeaza ne vom restrânge aria de consideratii numai asupra sistemelor si subsistemelor numerice (digitale). Denumirile de sistem si subsistem, exceptând situatiile în care este evidenta o ierarhizare, vor fi utilizate alternativ - întrucât un sistem poate fi întotdeauna considerat un subsistem al altui sistem. O caracteristica importanta a sistemelor digitale o constituie complexitatea; de complexitatea unui sistem este strâns corelat gradul de abstractizare al considerarii acestuia. Nivelul de abstractizare al descrierii unui sistem poate fi succint caracterizat prin tipul de informatie procesat de respectivul sistem (vezi figura 1.1)

Control	Date	Nivel de abstractizare
1. Valori Logice (sau secvente de)	Valori Logice (sau secvente de)	Nivel Logic
2. Valori Logice	Cuvinte	Nivel Registru
3. Instructiuni	Cuvinte	Nivel Set de Instructiuni
4. Programe	Structuri de Date	Nivel Procesor
5. Mesaje	Mesaje	Nivel Sistem

Figura 1.1 Nivele de abstractizare a informatiei procesate de un sistem.

Nivelul Logic.

Informatia procesata la acest nivel poate fi reprezentata prin valori logice discrete: de regula valori logice binare (0 si 1). Nivele de abstractizare (sau modele) mai rafinate pot reclama mai multe valori logice. O distinctie mai profunda în acest sens poate fi facuta tinând seama de natura combinationala (nu reflecta o istorie anterioara a sistemului) sau secventiala

(raspunsul sistemului la un moment dat depinde atît de stimulii anterior aplicati cît şi de stimulii actual aplicati) a comportamentului sistemului.
De regula sistemele considerate la acest nivel sunt circuitele; orice sistem poate fi considerat la nivel de circuit dar efortul de analiza poate fi mult prea mare sau chiar imposibil.

Nivelele de abstractizare superioare nivelului logic apar atunci cînd considerarea operatiilor efectuate de un sistem la acest nivel conduce la modele greoaie, stufoase si dificil de folosit.

Nivelul Registru.

In mod uzual un sistem este conceput prin interactia informatiilor de procesat (datele) cu informatia de stare (control) a sistemului. Menţinînd definiţia functiei de control la nivel de valori logice, în cadrul acestui nivel se considera ca informatia procesata (datele) este grupata în vectori (cuvinte) de valori logice . Deoarece cuvintele sunt stocate în decursul procesarii în registre, acest nivel a primit numele corespunzator.

Nivelul Set de Instructiuni.

Nivelul acesta de abstractizare considera informatia de control grupata in cuvinte numite instructiuni.

Nivelul Procesor.

Acest nivel imediat superior celui anterior, considera un sistem numeric ca procesând secvente de instructiuni, sau programe, ce opereaza asupra unor blocuri de date numite structuri de date.

Nivelul Sistem.

Un mod diferit de a vedea un sitem numeric (nu neaparat la un nivel mai ridicat de abstractizare) este considerarea acestuia ca fiind constituit din mai multe subsisteme independente, sau unitati, ce inter-comunica prin blocuri de cuvinte numite mesaje.

In general, stimulii si raspunsul la acesti stimuli ai unui sistem oarecare, sunt elementele ce definesc un experiment de testare si corespund tipului de informatie procesat de sistemul aflat în test. Din acest punct de vedere testarea este un termen generic ce acopera un spectru larg de activitati si medii:
-unul sau mai multe subsisteme ce testeaza un altul prin transmiterea si receptia unor mesaje;
-un procesor ce se autotesteaza prin executia unui program de diagnostic;
-un echipament automat de testare (**EAT**) ce verifica un circuit prin aplicarea unor vectori binari de test, operatie conjugata cu examinarea vectorilor binari emisi de respectivul circuit ca urmare a stimulilor aplicati.

O alta categorie de teste sunt testele parametrice. Aceasta testare trateaza problema examinarii caracteristicilor electrice ale circuitelor (cum ar fi tensiuni de polarizare, curenti de pierderi, tensiuni de prag, etc). Aceste tipuri de testari nu fac

obiectul acestui curs.

Erori si Defecte

O aparitie (instantiere) a unei malfunctionari a unui sistem testat (sau **UAT**, prescurtare pentru **Unitate Aflata in Test**) se numeste *eroare* (observata). De remarcat faptul ca in general conceptul de eroare are interpretari diferite la nivele de abstractizare distincte. Spre exemplu, o eroare observata la nivelul unui program de diagnostic poate apare ca un rezultat incorect al unei operatii aritmetice (sa admitem), în timp ce pentru un EAT o eroare înseamna de obicei o valoare binara incorecta.
Cauzele unei erori observate pot consta din: erori de proiectare, erori de fabricatie, defecte de fabricatie si defecte fizice.

Exemple tipice de erori de proiectare:
-specificatii incomplete sau contradictorii;
-corespondente si aplicatii eronate ale diferitelor nivele de proiectare;
-nerespectarea conventiilor si regulilor de proiectare.

Printre erorile ce au loc pe durata fabricatiei sunt cuprinse:
-componente necorespunzatoare;
-conexiuni incorecte;
-scurtcircuite datorate unor conexiuni sudate sau metalizate inadecvat.

Defectele de fabricatie nu sunt direct atribuibile erorilor umane; mai degraba aceste defecte rezulta dintr-un proces de manufacturare imperfect. Spre exemplu scurtcircuitele (uneori prescurtate "scurturi") si intreruperile sunt defecte comune in fabricatia circuitelor MOS pe Scara Larga de Integrare (se va folosi totusi prescurtarea **MOS LSI**, devenita clasica). Alte defecte de fabricatie includ profile de dopaj necorespunzatoare, erori de aliniere a mastilor si încapsulari defectuoase. O corecta localizare a defectelor de fabricatie este importanta pentru cresterea productivitatii manufacturarii respective.

Defectele fizice apar pe durata timpului de viata al sistemului si sunt datorate degradarii componentelor si/sau factorilor de mediu. Spre exemplu, conexiunile de aluminiu din interiorul circuitelor integrate (**CI**) se subtiaza cu timpul (datorita migratiei electronilor si coroziunii) si pot sa se intrerupa. Factorii de mediu cum ar fi temperatura , umiditatea si vibratiile, accelereaza îmbatrânirea componenetelor. Radiatii cosmice si particule alfa pot induce defecte in chip-urile ce contin memorii de mare densitate cu acces aleator (**RAM**-uri).

Erorile de proiectare, defectele de fabricatie si defectele fizice sunt la un loc referite ca *defecte fizice*. In raport cu stabilitatea lor în timp, defectele pot fi clasificate ca fiind:
-permanente, adica fiind prezente întotdeauna dupa aparitia lor;
-intermitente, adica au existenta marginita numai pe durata unor intervale de timp;
-tranzitorii, adica aparitii singulare cauzate, de regula, de

schimbarea temporara a unui factor de mediu.
In general, defectele fizice nu permit o tratare matematica, o modelare, a testarii si diagnozei. Solutia acestei probleme este abordarea prin modelarea defectelor fizice cu *defecte logice*, care ofera o reprezentare convenabila a efectelor defectelor fizice asupra operarii normale a unui sistem. Un defect este *detectat* prin observarea erorii cauzate de acesta. Ipotezele de baza privitor la natura defectelor logice sunt referite generic ca fiind *modelul defectului*. Cel mai larg utilizat model de defect este acela al unei linii singulare (fir, conexiune, legatura) ce este permanent "blocata" (stuck at, in literatura anglo-americana de profil) la o valoare logica (1 sau 0). Notatia curenta pentru o linie arbitrara w este : w *blocata-la-0*, sau w *blocata-la-1* (abrevierile curente sunt w *b-1-0*, respectiv w *b-1-1*).

Modelare si Simulare

Deoarece erorile de proiectare preced fabricatia unui sistem, se poate realiza *testarea verificarii unui proiect* printr-un experiment de test ce foloseste un model al sistemului proiectat. In acest context, "model" înseamna o reprezentare numerica pe calculator în termeni de structuri de date si/sau programe.
Modelul poate fi exercitat prin simularea acestuia cu reprezentarea semnalelor de intrare. Procesul se numeste *simulare logica* (deasemenea se mai spune *simularea verificarii proiectarii* sau *simularea valorii adevarate*). Simularea logica determina evolutia în timp a semnalelor din model ca raspuns la secventa de intrare aplicata.

Evaluarea Testului

O problema importanta în testare este *evaluarea testului*, prin aceasta referindu-se determinarea eficientei, sau calitatea, unui test. Evaluarea testului se face uzual în contextul unui model al defectului, iar calitatea unui test este masurata prin raportul dintre numarul de defecte detectate si numarul total de defecte din universul asumat de defecte, pentru sistemul respectiv; acest raport este referit sub denumirea de *acoperirea defectelor*. Evaluarea testului (sau calificarea testului) este facuta printr-un experiment de testare simulat, numit *simularea defectului*, care calculeaza raspunsul circuitului (sistemului) în prezenta defectelor pentru care testul este evaluat. Un defect este detectat daca exista o diferenta, o deosebire, între raspunsul circuitului (sistemului) liber de defecte - circuitul (sistemul) functionând normal - si raspunsul circuitului (sistemului) afectat de defectul respectiv.

In acest sens se spune ca un circuit (sistem) este *redundant* daca exista cel putin un defect care nu poate fi pus în evidenta atunci când i se aplica respectivului circuit (sistem) la intrare orice secventa de stimuli posibila, admisa, în conformitate cu specificatiile respective de functionare.

Tipuri de testare

Metodele de testare pot fi clasificate în raport cu mai multe criterii.

Testarea prin *programe de diagnoza*, se realizeaza "off-line", la viteza normala de functionare, si la nivelul sistemului. Stimulii îsi au originea în interiorul sistemului însusi, care lucreaza în regim de auto-testare. In sistemele a caror logica de control este microprogramata, programele de diagnostic pot fi deasemenea microprograme (microdiagnosticari). Anumite parti ale sistemului, numite global *inima sistemului ("hardcore",*in original) trebuie sa fie libere de defecte pntru a putea permite rularea programelor. Stimulii sunt generati prin software sau prin firmware si pot fi aplicati adaptiv. Programele de diagnostic sunt de regula rulate în testarea de întretinere (preventiv) sau de rutina.

Emularea în circuit, este o metoda de testare ce elimina necesitatea unei parti perfect functionale a sistemului testat pentru rularea programelor de diagnostic. Aceasta metoda este folosita în testarea placilor si sistemelor cu microprocesoare, si se bazeaza pe îndepartarea microprocesorului de pe placa, pe durata testarii, si pe accesarea conexiunilor microprocesorului cu restul sistemului, circuitului, testat. Testorul poate emula functia microprocesorului îndepartat (de regula printr-un microprocesor de acelasi tip). Aceasta configuratie permite rularea programelor de diagnostic folosind memoria si microprocesorul testorului.

In *testarea "on-line"*, stimulii si raspunsurile sistemului sunt cunoscute dinainte, deoarece stimulii sunt furnizati prin secvente receptionate pe durata modului normal de operare. Obiectul de interes in testarea "on-line" consta nu atât din raspunsul însusi, cât din unele proprietati ale raspunsului, proprietati ce ar trebui sa ramâna invariante pe durata unei functionari normale. Spre exemplu o singura iesire a unui multiplexor liber de defecte trebuie sa aiba valoarea logica 1. Scopul unui bit de paritate este crearea unei proprietati invariante usor de verificat. Bitul de paritate este *redundant*, in sensul ca acesta nu poarta nici o informatie strict utila functionarii normale a sistemului. Acest tip de *informatie redundanta* este caracteristica pentru sisteme ce folosesc *coduri detectoare si coduri corectoare de erori*. Un alt tip de proiectare fiabila bazata pe redundanta este *redundanta modulara*, care se bazeaza pe replicarea de un numar de ori a sistemului. Modulele replicate (acestea trebuie sa aiba aceeasi functionare, posibil cu diferite implementari) lucreaza cu acelasi set de intrari, iar proprietatea invarianta este aceea ca toate modulele replicate trebuie sa produca acelasi raspuns. Sistemele cu auto-verificare au subcircuite speciale numite verificatoare ("*checkers*"), dedicate testarii proprietatilor invariante.

Testarea cu proba ghidata este o tehnica folosita în testarea la nivel de placa. Daca sunt detectate erori pe durata testarii initiale la pinii conectorilor (faza a testarii cunoscute adesea sub numele de test merge/nu merge), testorul decide care linie interna a circuitului sa fie monitorizata si instruieste

operatorul sa plaseze o proba (sonda) pe linia selectata. Apoi se reaplica testul. Principiul este sa se traseze înapoi, regresiv, propagarea erorii (erorilor) de-alungul unei cai prin circuit. Dupa fiecare aplicare a testului, testerul verifica rezultatul obtinut pe linia monitorizata si determina când a fost atins locul implantarii defectului si/sau când se continua trasarea înapoi. In loc sa monitorizeze o singura linie la un moment dat, unele testoare pot monitoriza un grup de linii, de regula pinii unui CI.

Testarea cu proba ghidata este o procedura secventiala de diagnostic, în care un subset al liniilor interne accesibile este monitorizat la fiecare pas. Anumite testoare folosesc un dispozitiv special de fixare numit *pat-de-cuie* ("*bed-of-nails*", in original) care permite monitorizarea tuturor liniilor interne accesibile ale circuitului, într-un singur pas.

Testarea in circuit, are drept scop verificarea componentelor deja plantate pe placa. Un tester exterior foloseste un conector cleste ("clip" in original) de CI pentru a putea aplica vectorii de test direct la intrarile unui CI si pentru a-i observa raspunsul. Testerul trebuie sa fie capabil sa izoleze electronic CI testat de restul placii pe care se afla plantat, altfel poate supraincarca iesirile altor CI conexe.

Testarea algoritmica se refera la generarea vectorilor de intrare pe durata testarii. Numaratoare si registre de deplasare cu reactie sunt exmple clasice de "hardware" folosit in generarea stimulilor de intrare. Generarea algoritmica a vectorilor de test este capacitatea anumitor testere de a produce combinatii a unor vectori fixati. Combinatia dorita este determinata pritr-un program de control scris într-un limbaj orientat pe aplicatie. Raspunsul asteptat poate fi generat pe durata testarii fie de la un exemplar copie al UAT (unitatii aflate in test) cunoscut ca fiind bun functional - asa-zisa unitate etalon - sau folosind o emulare in timp real a UAT. Acest tip de testare este numit uneori *testare prin comparatie*, ceea ce constituie cumva o denumire improprie, deoarece compararea raspunsului asteptat este inerenta oricarei metode de testare.

Metodele bazate pe verificarea anumitor functii $f(R)$ deduse din R raspunsul UAT, in loc sa se foloseasca chiar R, se spune ca realizeaza *testarea compacta*, iar $f(R)$ se spune ca este reprezentarea *compresata*, sau *semnatura* raspunsului R. Spre exemplu se pot contoriza numarul de valori 1 (sau numarul de tranzitii din 1 in 0, sau din 0 in 1) obtinut la iesirea circuitului si apoi o comparare a acestuia cu numarul asteptat al circuitului liber de defecte.
O astfel de procedura de testare simplifica procesul de testare, deoarece în loc sa se compare întreaga iesire R a circuitului bit cu bit cu iesirea asteptata se compara doar reprezentarile compresate sau semnaturile acestor iesiri. Sunt micsorate semnificativ cerintele de memorie ale testorului, deoarece nu mai este necesar sa se stocheze întregul raspuns asteptat (martor). Tehnicile de compresare sunt folosite în mod deosebit în circuitele cu autotestare, unde implementarea functiei $f(R)$ se face printr-un "hardware" special adaugat suplimentar circuitului. Circuitele cu autotestare au deasemenea "hardware"

aditional pentru generarea stimulilor.

Diagnostic si Reparatie

Daca o UAT ce se dovedeste ca functioneaza incorect trebuie sa fie reparata, este necesara diagnosticarea cauzei erorii observate. Intr-un sens mai larg, termenii diagnostic si reparatie se aplica atât defectelor fizice cât si erorilor de proiectare (pentru cazul din urma "reparatie" înseamna "reproiectare"). Totusi, câta vreme defectele fizice pot fi efectiv reprezentate prin defecte logice, nu acelasi lucru se întâmpla cu defectele de proiectare. Deaceea , în abordarea discutiei diagnosticului si reparatiei ne vom restrânge subiectul doar la defectele fizice (si logice).

Sunt posibile doua tipuri de abordari pentru diagnosticul defectelor. Prima abordare este *analiza cauza-efect*, care enumera toate defectele (cauzele) posibile sa existe într-un model al defectului si determina, înainte de experimentul de testare, toate raspunsurile (efectele) corespunzatoare unui test aplicat. Acest proces ce se bazeaza pe simularea defectelor, construieste o baza de date numita *dictionarul defectelor*. Diagnosticul este un proces de cautare în dictionar, în care se încearca sa se potriveasca raspunsul actual al UAT cu unul din raspunsurile precalculate. Daca potrivirea are loc , dictionarul de defecte indica defectele posibile (sau componenetele defecte) din UAT.

Alte tehnici de diagnostic, cum ar fi testarea cu proba ghidata, folosesc o abordare *analiza efect-cauza*. O analiza efect-cauza proceseaza raspunsul actual al UAT (efectul) si încearca sa determine direct numai defectele (cauzele) ce ar putea produce acest raspuns.

Generarea Testului

Generarea Testului, (GT) este procesul de determinare al stimulilor necesari sa testeze un sistem numeric. GT depinde în primul rând de metoda de test folosita. Metodele de testare "on-line" nu necesita GT. Un efort mic este necesitat pentru GT atunci când vectorii de intrare (de test) sunt furnizati de un registru de deplasare cu reactie lucrând ca un generator de secvente pseudoaleatoare. In schimb, GT pentru testare si verificarea proiectarii, ca si pentru dezvoltarea programelor de diagnostic implica, în general, un efort considerabil care din nefericire este înca în mare masura o activitate manuala.

Generarea testului automata (**GTA**) se refera la algoritmi care, dat fiind un model al sistemului, pot genera teste pentru acesta. GTA au fost dezvoltate in mod deosebit pentru testarea la nivel de pini-conectori cu secventa testate.

GT poate fi orientata pe defecte sau orientata pe functie. In GT orientata pe defecte, GT incearca sa genereze testele ce vor detecta (si posibil localiza, in anumite limite de distinctibilitate) defectele specificate.

Proiectare pentru Testabilitate

Costul unei testari pentru un sistem poate deveni o componenta majora in costul proiectarii, fabricatiei si intretinerii acestuia. Costul testarii reflecta multi factori, cum ar fi costul GT, costul timpului de testare, costul echipamentului automat de testare (costul **EAT**), etc. Este cumva uimitor ca un microprocesor cu un cost de cca. 10 $US, necesita testere de sute de ori mai scumpe.

Proiectarea pentru testabilitate (PPT) cumuleaza o seama de algoritmi si tehnici ce s-au dezvoltat mult în ultimul timp. Scopul acestora este reducerea costului testarii prin introducerea criteriilor de testabilitate cât mai devreme cu putinta în etapa de proiectare. Consideratiile de testabilitate au devenit atât de importante încât au ajuns chiar sa impuna structura generala a unui proiect.

Bibliografia cursului:

1. Miron Abramovici, Melvin Breuer, Arthur Friedman
   "DIGITAL SYSTEMS TESTING AND TESTABLE DESIGN",
   Computer Press

2. Melvin Breuer, Arthur Friedman
   "DIAGNOSIS & RELIABLE DESIGN OF DIGITAL SYSTEMS",
   Computer Science Press, Inc.

3. Danut Olteanu, Constantin Popescu
   "CIRCUITE INTEGRATE PE ARII DE PORTI LOGICE",
   Editura Tehnica, Bucuresti 1991.

4. Mircea Vladutiu, Marius Crisan
   "TEHNICA TESTARII ECHIPAMENTELOR AUTOMATE
   DE PRELUCRARE A DATELOR",
   Editura Facla, Timisoara 1989.

5. Vasile M. Catuneanu, Angelica Bacivarof
   "STRUCTURI ELECTRONICE DE INALTA FIABILITATE
   toleranta la defectari",
   Editura Militara, Bucuresti 1989.

6. Dan Teodorescu,
   "AUTOMATIZARI MICROELECTRONICE",
   Editura Tehnica, Bucuresti 1988.

7. Constantin C. Timoc
   "LOGIC DESIGN FOR TESTABILITY",
   IEEE Computer Society Press

8. International Symposium on Fault-Tolerant Computing
   Proceedings, IEEE.

9. The International Test Conference Procedings, IEEE.

10. Design Automaton Conference Proceedings, ACM si IEEE.

11. R.G. Bennetts (Cirrus Computers Limited)
    "DESIGN OF TESTABLE LOGIC CIRCUITS",
    Addison-Wesley Publishing Company, London 1984.