REDUNDANŢĂ HARDWARE

Sergiu COJUHARI[1], Radu MELNIC[1], Gheorge CEBAN[1], Aliona PUTERE[2]
Universitatea Tehnică a Moldovei[1]
Universitatea Agrară de Stat din Moldova[2]
kojuharis@yahoo.com

***Abstract* — În lucrarea dată sunt prezentate scheme logice de asigurare a redundanţei dispozitivelor tehnicii de calcul. Sunt prezentate trei scheme: 1:1, 1:N şi redundanţă în lanţ. Schema 1:1 se foloseşte pentru dispozitive unice (în sistem) cu conexiune simplă. 1:N se foloseşte în cazul utilizării a mai multor dispozitive de acelaşi tip cu conexiuni simple. În această schemă pentru o serie de dispozitive se foloseşte un singur dispozitiv de asigurare a redundanţei. Lanţuri de redundanţă se folosesc in cazul când există mai multe dispozitive unice cu conexiuni complicate.**

***Cuvinte cheie* — redundanţă, dispozitiv redundant, dispozitiv activ, dispozitiv pasiv, Heath Check, comenzi SNMP.**

Arhitectură hardware redundantă reprezintă nişte scheme de conectare a dispozitivelor ce funcţionează în paralel. Se utilizează trei tipuri de scheme: active, pasive[1], [2] şi mai rar hibride[3].

Unele dispozitive sunt principale (active), iar altele sunt secundare (pasive). Dispozitivele secundare se activează numai dacă există suspecţii asupra disponibilităţii dispozitivului activ. Redundanţa reprezintă abilitatea de conectare a dispozitivelor de rezervă fără a întrerupe funcţionarea reţelei şi timpul necesar pentru activarea dispozitivului redundant (downtime) este foarte redus.

În tabelul de mai jos sunt prezentate diferite nivele de disponibilitate (în procente), timpul indisponibilităţii pe an şi numărul de defecte per milion de rânduri de cod, ultima coloană referindu-se mai mult la programare.

Disponibilitate (%)	Timp indisponibil pe an	Defecte per milion
99,000%	3 zile, 15 ore, 36 minute	10.000
99,900%	8 ore, 46 minute	1.000
99,990%	53 minute	100
99,999%	5 minute	10

Profitul utilizării redundanţei în sisteme mari este evident în legătură cu creşterea disponibilităţii şi reducerea timpului de blocare a sistemului. Timpul downtime-ului este foarte costisitor şi foarte des este inclus în contractele comerciale a companiilor de prestare a serviciilor. Astfel investiţiile în redundanţă sunt recuperate la un nivel foarte înalt. Aşa cum nu este posibil de asigurat redundanţa totală, existenţa diferitor scheme permite optimizarea cheltuielilor şi aplicarea unor scheme optime de redundanţă. În lucrarea dată sunt caracterizate trei scheme de asigurare a redundanţei dintre cele mai cunoscute şi utilizate.

SCHEMA 1:1

Această schemă se foloseşte pentru asigurarea disponibilităţii unor dispozitive unice. Evident această schemă este costisitoare deoarece pentru un dispozitiv activ în sistem se asigură un dispozitiv redundant[4].

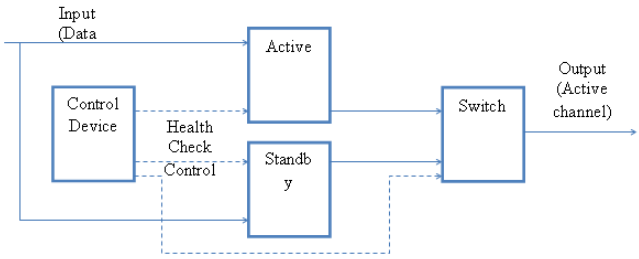

fig.1 Redundanţa 1:1

Comutarea între elementul activ şi redundant se dirijează de un alt dispozitiv ce reglează logica comutării (fig.1). Dispozitivul dat poate controla toate elementele redundante din sistem, verifica baza de date sau fişierele de configurare etc., şi la rândul lui poate fi supus redundanţei, însă deja prin alte scheme.

Dispozitivul de control verifică starea dispozitivelor active şi pasive printr-un mecanism ce se numeşte Heath Check – în varianta cea mai simplă ping-uirea dispozitivelor; în variantele mai complicate comenzi SNMP sau protocol propriu de verificare a disponibilităţii dispozitivelor. Datele de intrare sunt conectate la ambele dispozitive. Însă în dependenţă de starea lor numai unul poate fiind activ generând date de ieşire. Pentru a controla ieşirile dispozitivelor, ele sunt conectate la un switch, iar switch-ul la fel este dirijat de dispozitiv de control. Cu ajutorul configurării porturilor deschise şi închise se asigură output-ul corect. Timpul de comutare este destul de mic în dependenţă de numărul de teste HealthCheck utilizate, după care se începe procedura de comutare plus timpul necesar pentru schimbarea porturilor în Switch. Pentru switch nu se asigură redundanţă specială, deoarece acest lucru este foarte greu de realizat, iar într-al doilea rând companiile producătoare asigură o disponibilitate maximă - 99,999%. Dacă totuşi e necesar de realizat o redundanţă suplimentară, de regulă se folosesc scheme şi mai complexe – redundanţa prin dublare a sistemului întreg. De regulă acest gen de redundanţă şi comutare nu se execută automat dar manual de către operatorul sistemului [5].

SCHEMA 1:N

Această schemă se foloseşte pentru asigurarea disponibilităţii a mai multor dispozitive identice. Evident

această schemă este cu mult mai ieftină: un singur dispozitiv poate susţine mai multe dispozitive active, mai puţină energie utilizată etc. Însă schema are şi neajunsuri, principalul fiind: dacă iese din funcţiune mai multe dispozitive odată sistemul în întregime nu va mai putea funcţiona.

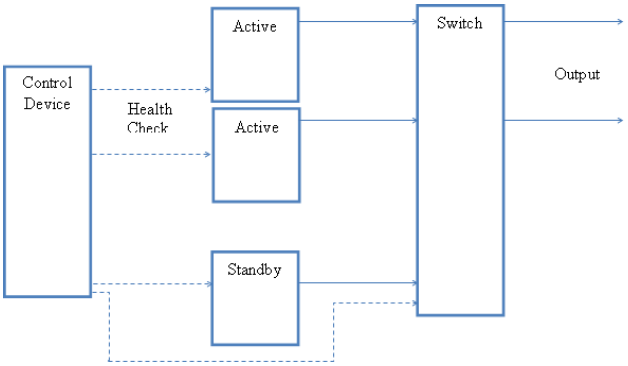

Fig. 2 Redundanţa 1:N

Cum se vede din schema de mai sus (fig.2) aici avem mai multe dispozitive identice şi unul singur element de asigurare a redundanţei. Schema de conectare însă este complet diferită faţă de schema 1:1. Conectarea input nu mai este directă şi se asigură prin aplicarea adreselor IP într-o subreţea. În rest logica de determinare a stării dispozitivelor este identică. Dacă un dispozitiv nu mai este disponibil controlul se transmite dispozitivului standby, adresa IP şi comanda de reboot se transmit către Switch. Porturile ce corespunde dispozitivului eşuat sunt închise, iar cele ale Stanbdy se deschid. Schema este comparativ simplă şi benefică.

SCHEMĂ REDUNDANŢĂ ÎN LANŢ

Schemele precedente presupun conexiuni simple – prin cabluri de reţea şi setarea unor anumite IP când o magistrală comună asigură conectarea tuturor dispozitivelor. Însă, în unele cazuri e necesar de asigurat conexiuni speciale ca de exemplu: garantarea vitezei maxime posibile, continuitatea fluxului de date, ne digitizarea datelor, etc. [6]. În aceste cazuri ar trebui de folosit alt gen de redundanţă. Această diferenţă constă în faptul că fiecare element execută operaţiuni speciale şi deci trebuie de garantat redundanţa a mai multor elemente ca a unui element întreg vezi fig.3.

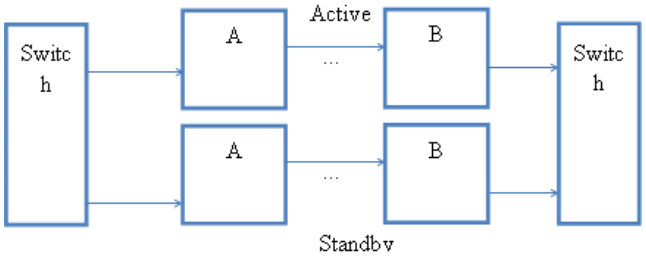

Fig. 3 Redundanţa unui lanţ de dispozitive

Elementele A...B sunt specifice şi au câte un element redundant (Standby), însă redundanţa se asigură nu pentru un element aparte dar pentru întregul lanţ de elemente. Pe schemă nu este indicat elementul de control însă el există. În cazul dat validitatea nu se verifică pentru fiecare element in parte dar se verifică datele de ieşire. De regulă acest tip de redundanţă să foloseşte când fluxul de date este permanent - continuu. Din acest motiv verificarea este destul de simplă. Dacă din careva motive un element din lanţ nu mai funcţionează se execută comutarea lanţului în întregime.

CONCLUZII:

În sisteme mari – server-ele prestatorilor de servicii, sisteme de căutare\stocare\prelucrare a datelor etc., unde sunt folosite multe dispozitive speciale, redundanţa lor este realitatea zilelor noastre. Asigurarea disponibilităţii sistemului necesită investiţii mari şi cunoştinţe speciale pentru asigurarea redundanţei ceia ce conduce la optimizarea cheltuielilor şi aplicarea schemelor optime. În lucrarea dată s-a încercat de a face o prezentare generală a celor mai populare şi simple scheme de redundanţă. În special au fost descrise scheme active şi pasive.

BIBLIOGRAFIE:

[1] Elena Dubrova, "International Master Program in System on Chip Design", "Fault Tolerant System Design", page 3. http://web.it.kth.se/~dubrova/FTCcourse/LECTURES/lecture4.pdf

[2] Axel Krings, University of Idaho in Moscow, "Fault-Tolerant Systems", Sequence 3 . http://www2.cs.uidaho.edu/~krings/CS449/Notes.F11/449-11-03.pdf.

[3] Matteo Ainardi şi alţii. "Fault Tolerant Processor using Hybrid Hardware Redundancy", Programmable System Design. Testing and Design for Reliability of Digital Systems, March 2009 . http://www.scribd.com/doc/13329601/Fault-Tolerant-Processor-Using-Hybrid-Hardware-Redundancy.

[4] http://www.eventhelix.com/realtimemantra/HardwareFaultTolerance.htm - Hardware Fault Tolerance.

[5] SIP Endpoint SDK Disaster Recovery and Geo-Redundancy, http://developerzone.genesyslab.com/wiki/index.php?title=Category:SIP_Endpoint_SDK_Developer%27s_Guide

[6] Stephen Y H Su "A Hardware Redundancy Reconfiguration Scheme for Tolerating Multiple Module Failures", journal "IEEE Transactions on Computers", Volume 29 Issue 3, March 1980, State University of New York.