POVEĆANJE ENERGETSKE EFIKASNOSTI RTS-A SA REDUNDANSOM U VREMENU ZA PREVAZILAŽENJE OTKAZA

Sandra Đošić, *Elektronski fakultet u Nišu, sandra.djosic@elfak.ni.ac.rs*
Milun Jevtić, *Elektronski fakultet u Nišu*, *milun.jevtic@elfak.ni.ac.rs*

Sadržaj – *U ovom radu analizira se rad RTS-a sa stanovišta potrošnje. Konkretno, razmatrani su RTS-i koji imaju mogućnost prevazilaženja prolaznih otkaza korišćenjem vremenske redundanse. Pretpostavljeno je da se otkazi mogu javiti u toku izvršenja zadataka i da se RTS oporavlja tako što ponovo izvršava zadatak kod koga je detektovana greška. Za analizu potrošnje mi predlažemo jedan heuristički DVS algoritam, a zatim pomoću njega analiziramo rad RTS-a sa posebnim osvrtom na vezu između potrošnje i redundanse u vremenu.*

1. UVOD

Sistemi za rad u realnom vremenu (eng. *Real-Time Systems* - RTSs) često se projektuju tako da imaju mogućnost da nastave sa radom čak i kada se pojavi neki otkaz (eng. *Fault-Tolerant Real-Time Systems* – FT RTSs). FT RTS-i mogu se definisati kao visoko-pouzdani sistemi koji i u vrlo nepovoljnim, pa i ekstremnim uslovima nastavljaju da izvršavaju svoje funkcije, [1]. Greške koje mogu dovesti do otkaza kod RTS-a mogu biti stalne (permanentne), prolazne (tranzijentne) ili povremene, pri čemu su prolazne greške najzastupljenije kod RTS-a. Za prolazne greške je karakteristično da se mogu pojaviti i zatim nestati u veoma kratkom vremenskom periodu, [2].

Da bi sistem uspešno nastavio sa radom čak i u prisustvu otkaza neophodno je da iskoristi neku od dostupnih redundansi. Redundansa se može definisati kao dodatni hardverski ili softverski izvor, dodatno vreme ili informacija, preko onoga što je potrebno za normalno funkcionisanje sistema, a sve sa ciljem postizanja što boljih performansi sistema, [3]. Naša pažnja usmerena je na vremensku redundansu koja koristi slobodno (dodatno) vreme za izvršavanje funkcija oporavka sistema pri nastanku otkaza, [4].

Još jedna veoma bitna karakteristika RTS-a jeste njihova energetska efikasnost. Tehnika poznata kao DVFS (eng. *Dynamic Voltage and Frequency Scaling*) danas se sve češće koristi kod RTS-a kada se želi povećati njihova energetska efikasnost smanjenjem potrošnje procesora, [5-7]. Ušteda u potrošnji ostvarena je smanjivanjem napona napajanja (a samim tim i radne frekvencije) procesora. Ovim se povećava vreme potrebno da processor izvrši zadatke pa se može reći da DVFS tehnika koristi slobodno vreme procesora kako bi smanjila potrošnju.

Kako slobodno vreme procesora koriste i DVFS tehnike i tehnike vezane za prevazilaženje otkaza korišćenjem vremenske redundanse postavlja se pitanje kojoj tehnici dodeliti koliko slobodnog vremena. Očigledno je da ako DVFS tehnike potroše više slobodnog vremena manje će ostati za tehnike prevazilaženja otkaza i obrnuto. Ova dilema je često obrađivana u literaturi [6-8]. Cilj je naći odgovarajuću frekvenciju na kojoj će procesor izvršiti zadatake tako da potrošnja RTS-a bude najmanja moguća a da se zadovolje svi zahtevi vezani za prevazilaženje otkaza.

U pokušaju da rešimo dilemu oko nalaženja kompromisa između energetske efikasnosti i sposobnosti prevazilaženja otkaza kod RTS-a, u radu ćemo predstaviti jedan heuristički DVFS algoritam koji smo mi razvili. Takođe, biće predstavljeni rezultati analize rada RTS-a, zasnovane na predstavljenom heurističkom DVFS algoritmu, na osnovu kojih je moguće povećati energetsku efikasnost FT RTS-a.

Ostatak rada organizovan je na sledeći način. U 2. delu dat je opis modela korišćenih u radu. Nakon toga sledi opis heurističkog DVFS algoritma. Rezultati simulacije dati su u 4. delu, nakon kojeg sledi zaključak.

2. OPIS MODELA

U radu se pretpostavlja jedan jednoprocesorski RTS koji ima mogućnost promene frekvencije rada procesora f_j (j = 1, ..., m) gde je $f_j < f_{j+1}$. Promenom f_j menja se i napon napajanja procesora pri čemu takođe pretpostvaljamo da postoji m naponskih nivoa. Ovakav sistem može se iskorititi za izvršavanje skupa sačinjenog od n *real-time* zadataka, $\Gamma=\{\tau_1, ..., \tau_n\}$. Za svaki zadatak τ_i poznat je period pojavljivanja T_i, vreme izvršenja zadatka u najgorem slučaju (eng. *worst case execution time* - WCET) C_i, rok za izvršenje D_i ($D_i \leq T_i$) kao i prioritet p_i. Parametar C_i odnosi se na vreme izvršenja zadatka τ_i kada procesor radi na maksimalnoj frekvenciji f_m. Radi jednostavnije dalje računice mi pretpostavljamo da se sa promenom frekvencije rada procesora f_j linearno menja i vreme izvršenja zadatka C_i tj. skaliranjem frekvencije f_j za faktor α neophodno je skalirati C_i za za faktor $1/\alpha$ odnosno važi da je

$$C_i(f_j) = C_i(f_m) f_m / f_j.$$

Što se potrošnje procesora tiče mi pretpostavljamo da važi sledeće

$$P_A(f) = P_d(f) + P_{ind},$$

gde su $P_d(f)$ i P_{ind} frekventno zavisne i frekventno nezavisna komponenta potrošnje, [9]. Komponenta koja zavisi od frekvencije može se napisati kao

$$P_d(f) = V^2(f)\, C_{ef} f$$

gde je sa V označen frekventno zavistan napon napajanja procesora, sa C_{ef} efektivna parazitna kapacitivnost tranzistora koji menjaju nivoe signala i sa f frekevncija rada procesora. Pored potrošnje za DVFS tehnike podjednako važna je i energija koja se definiše kao integral potrošnje po vremenu.

Pretpostavljamo da se greške mogu javiti u sistemu i da one mogu dovesti do prolaznih otkaza. Sistem se oporavlja tako što ponovo izvršava zadatak kod koga je detektovana greška pri tome ne menjajući njegov prioritet niti frekvenciju rada procesora. Ovo ponovno izvršavanje zadatka mora biti takvo da ne naruši vremenska ograničenja ostalih zadataka u Γ.

Za proveru izvodljivosti FT RTS-a tj. proveru da li svi *real-time* zadaci ispunjavaju vremenske zahteve, koristimo analizu vremena odziva (eng. *response time analysis* - RTA). U okviru RTA mogućnost prevazilaženja otkaza predstavljena je parametrom T_F koji odgovara minimalnom vremenu između dva uzastopna otkaza koje RTS može da prevaziđe bez narušavanja vremenskih ograničenja. Više o RTA može se naći u [10]. Jednačina koja karakteriše RTA

označena je sa (1) i pomoću nje moguće je izračunati vreme odziva R_i zadatka τ_i:

$$R_i^{n+1} = C_i + \sum_{j \in hp(i)} \left\lceil \frac{R_i^n}{T_j} \right\rceil C_j + \left\lceil \frac{R_i^n}{T_F} \right\rceil \max_{j \in hp(i) \cup i} (C_j) \qquad (1)$$

Jednačina (1) ima tri sabirka, prvi koji se odnosi na WCET zadatka τ_i, drugi koji predstavlja vreme potrebno za izvršenje svih zadataka višeg prioriteta od prioriteta p_i zadatka τ_i. Sa $hp(i)=\{\tau_j \in \Gamma \mid p_j > p_i\}$ označen je skup svih zadataka sa višim prioritetom od p_i. Treći sabirak odnosi se na vreme neophodno za oporavak zadatka usled nastale greške u RTS-u. Ako pretpostavimo da je minimalno vreme između pojave dve uzastopne greške T_F onda se u vremenskom intervalu koji odgovara vremenu odziva R_i može javiti maksimalno $\left\lceil \frac{R_i}{T_F} \right\rceil$ grešaka. Kako se ove greške mogu javiti u toku izvršenja zadatka τ_i ili nekog od zadataka višeg prioriteta, onda se trećem sabirku mora dodati i član $\max_{j \in hp(i) \cup i} (C_j)$.

Kako se R_i nalazi sa obe strane znaka jednakosti do rešenja se dolazi iterativnim postupkom pri čemu je početni uslov $R_i^0 = C_i$ a izlazni kriterijumi $R_i^{n+1} = R_i^n$ (trenutak kada se nalazi najgore vreme odziva zadatka τ_i) odnosno $R_i^{n+1} > D_i$ (slučaj kada τ_i ne može da zadovolji kriterijume trenutno aktivnog algoritma po kome se zadaci izvršavaju).

3. OPIS DVFS ALGORITMA

DVFS algoritam opisan u ovom poglavnju predstavlja jedan heuristički algoritam nastao kao rezultat našeg istraživanja vezanog za pronalaženje kompromisa između male potrošnje i velike verovatnoće prevazilaženja otkaza kod FT RTS-a. Predloženi algoritam nalazi frekevenciju na kojoj treba da radi procesor prilikom izvršavanja svakog pojedinačnog *real-time* zadataka iz skupa Γ tako da potrošnja bude minimalna moguća čak i kada postoji mogućnost pojave otkaza. Na Sl. 1 dat je pseudo kod predloženog heurističkog algoritma.

Osnova algoritma čini RTA koja se koristi za garantovanje izvodljivosti RTS-a kao i njegove mogućnosti prevazilaženja otkaza. Ulazni parametri algoritma su:

- frekvencijski nivoi na kojima procesor može da radi f_j (j = 1, ..., m) pri čemu je $f_j < f_{j+1}$ a m je broj nivoa;
- karakteristike svih n *real-time* zadataka: period T_i, WCET C_i ako processor radi na frekvenciji f_m, prioritet p_i i rok za izvršenje zadatka D_i, za i=1,..., n;
- minimalno vreme između pojave dve uzastopne greške T_F.

Algoritam počinje tako što svim *real-time* zadacima iz skupa dodeljuje maksimalnu frekvenciju izvršenja f_m, korak (1). Takođe su na početku svi zadaci "otključani" tj. imaju mogućnost da menjaju frekvenciju izvršenja. U svakoj iteraciji algoritma smanjuje sa radna frkevencija po jednog "otključanog" zadatka za po jedan frekvencijski nivo. Dalje se traži zadatak kod koga je smanjenje frekvencije dovelo do najveće uštede u potrošnji. Na primer, neka je zadatku τ_i privremeno smanjena frekvencija izvršenja sa f_j na f_{j-1}, korak (4). Zatim je neophodno proveriti izvodljivost RTS-a koristeći jednačinu (1), korak (5). Ako tako dobijeni skup zadataka nije izvodljiv zadatak τ_i se "zaključava", korak (6). U suprotnom, se računa razlika u potrošnji zadatka τ_i na nižoj (f_{j-1}) i višoj (f_j) frekvenciji izvršenja, korak (7). Nakon toga se frekvencija zadatka τ_i ponovo vraća na f_i. Cilj je naći "otključani" zadatak iz datog skupa kod koga je razlika u potrošnji maksimalna. Tom zadatku se dodeljuje frekvencija izvršenja smanjena za jedan frekvencijski nivo, korak (8). Zadaci se "zaključavaju" i kada im je dodeljena frekvencija izvršenja f_1. Algoritam završava u trenutku kada su svi zadaci iz skupa "zaključani". Izlaz algoritma je vektor sa frekvencija izvršenja koje su dodeljene svim zadacima iz skupa.

Input: operating frequency levels f_j (j=1..m),
characteristics for n real time tasks (C_i, D_i, T_i, p_i),
fault tolerant constraint (T_F)

---

(1) for each *Task* in *TaskSet* set *Task's_Freq* to f_m and set *Task's_Key* to true;

(2) repeat step (3) to (7) until there are true *Task's_Key* in the *TaskSet*;

(3) for each unlock *Task* in *TaskSet* do

(4) temporarily set *Task's_Freq* to *Lower_Task's_Freq*;

(5) *if* new *TaskSet* is not feasible

(6) then set *Task's_Key* to false;

(7) else calculate *ΔPower* as Power(*Task's_Freq*) – Power(*Lower_Task's_Freq*);

(8) find *Task* with maximum *ΔPower* and set *Task's_Freq* to *Lower_Task's_Freq*;

Output: *TaskSet* with new frequency assigne to each *Task*

Sl. 1. Pseudo kod predloženog heurističkog DVFS algoritma

U jednom od naših prethodnih radova dokazali smo predloženi heuristički DVFS algoritam i više o njemu može se naći u [11].

4. REZULTATI SIMULACIJE

Na bazi prethodno opisanog algoritma realizovali smo simulator koji nam je omogućio analizu rada RTS-a. Ulazni podaci za simulator su broj *real-time* zadataka i njihove vremenske karakteristike: period zadatka T_i, WCET C_i za radnu frekvenciju f_m, prioritet p_i i rok za izvršenje zadatka D_i. Takođe, ulazni podaci su i naponski i frekvencijski nivoi procesora kao i parametar T_F.

Simulator je iskorišćen za analizu velikog broja *real-time* skupova zadataka kako sintetizovanih tako i onih iz "realnog" sveta. U okviru ovog poglavlja daćemo rezultate simulacije RTS-a iz "realnog" sveta. Konkretno u pitanju je *real-time* skup zadataka vezan za Generic Avionics Platform (GAP), [12]. U tabeli 1 date su vremenske karakteristike deset zadataka iz GAP aplikacije.

Tabela 2 predstavlja podatke vezane za naponske i frekvencijske nivoe rada *Transmeta Crusoe* procesora, [13]. Ovaj procesor iskorišćen je u predstavljenoj simulaciji.

Tabela 1. *Real-time skup zadataka GAP aplikacije*

τ_i	p_i	$T_i=D_i$ (ms)	C_i (ms)
Nav_Status	1	1000	1
BET_E_Status_Update	2	1000	1
Display_Stat_Update	3	200	3
Display_Keyset	4	200	1
Display_Stores_Update	5	200	1
Nav_Steering_Cmds	6	200	3
Tracking_Target_Upd	7	100	5
Display_Hook_Update	8	80	2
Display_Graphic	9	80	9
Nav_Update	10	59	8

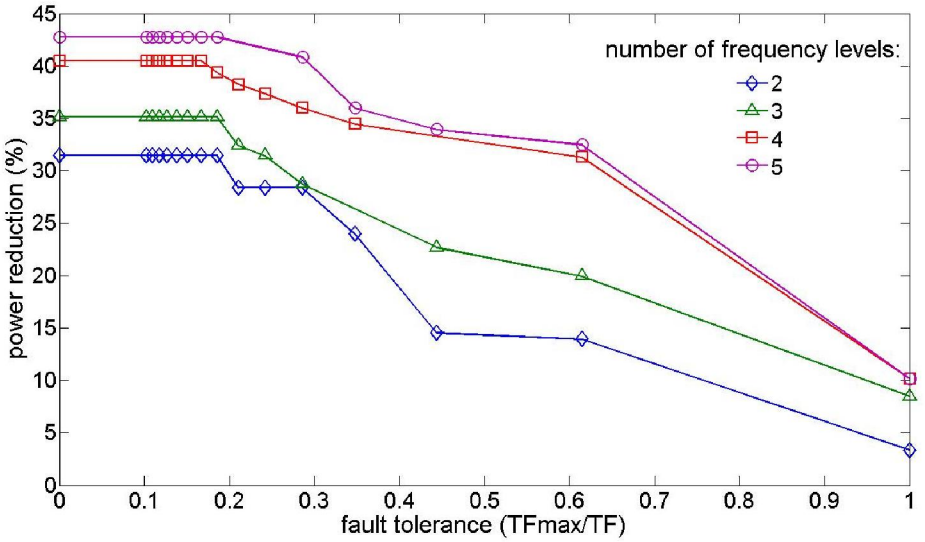
Rezultati simulacije predstavljeni su na Sl. 2. Simulacije su odrađene za skup zadataka iz tabele 1 i različiti broj frekvencijskih nivoa iz tabele 2. Za 2 frekvencijska nivoa korišćene su radne frekvencije (667MHz, 300MHz), za 3 (667MHz, 600MHz, 300MHz), za 4 (667MHz, 600MHz, 400MHz, 300MHz) i za 5 nivoa (667MHz, 600MHz, 533MHz, 400MHz, 300MHz). Na Sl. 2 na x-osi dat je odnos T_{Fmax} i T_F gde je T_{Fmax} minimalno vreme između dve uzastopne pojave greške koje system može da toleriše ako radi na maksimalnoj frekvenciji, a T_F je ulazni parameter simulacije. Ovaj odnos T_{Fmax}/T_F predstavlja normalizovanu vrednost parametra T_F i proporcionalan je sposobnosti sistema da prevaziđe otkaz. Kako je sposobnost sistema da prevaziđe otkaz proporcionalna vremenskoj redundansi sistema može se reći da x-osa predstavlja i slobodno (redundatno) vreme procesora. Na y-osi dat je odnos potrošnje RTS-a kada processor izvršava svaki zadataka na frekvenciji dobijenoj primenom heurističkog DVFS algoritma i potrošnje sistema ako processor radi uvek na maksimalnoj frekvenciji. Može se reći da ova osa predstavlja uštedu u potrošnji izraženu u procentima.

Tabela 2. *Frekvencija, napon i snaga Transmeta Crusoe procesora*

CPU Frequency (MHz)	Voltage (V)	CPU Power (W)
300	1.2	1.3
400	1.225	1.9
533	1.35	3
600	1.5	4.2
667	1.6	5.3

U skladu sa raspoloživim brojem frekvencijskih nivoa, na Sl. 2 mogu se videti četiri grafika. Ono što se prvo može zaključiti i što je zajedničko za sva četiri grafika je činjenica da se smanjenjem potrošnje smanjuje i mogućnost prevazilaženja otkaza i obrnuto.

Zatim, u skladu sa konkretnim rezultatima simulacije prikazanim na Sl. 2 sada se može preciznije proceniti odnos potrošnje i mogućnosti prevazilaženja otkaza. Recimo, neka se zahteva da ušteda u potrošnji bude između 40% i 45%. Sa Sl. 2 može se videti da procesor ako koristi 4 ili 5 frekvencijskih nivoa ispunjava date zahteve. Kako mogućnost prevazilaženja otkaza varira u okviru zahtevanog intervala smanjenja potrošnje, najbolje je onda izabrati slučaj kada je ta mogućnost maksimalna. Takođe, sa Sl. 2 se može videti da je ušteda u potrošnji veća kada se koristi procesor sa više radnih frekvencija.



Sl. 2. *Grafik zavisnosti uštede u potrošnji od mogućnosti prevazilaženja otkaza za različit broj radnih frekvencija procesora*

5. ZAKLJUČAK

U ovom radu bavili smo se potrošnjom FT RTS-a. Analizirani su RTS-i koji prevazilaze otkaze koristeći vremensku redundansu. Analiza je zasnovana na jednom heurističkom DVFS algoritmu koji smo mi razvili. Predloženi algoritam omogućava da se nađe kompromis između zahteva vezanih za energetsku efikasnost i zahteva vezanih za prevazilaženja otkaza RTS-a.

Koristeći predloženi heuristički DVFS algoritam realizovan je simulator koji nam je omogućio analizu energetske efikasnosti FT RTS-a. Na osnovu rezultata simulacije može se zaključiti da se veća ušteda u potrošnji RTS-a može ostvariti sa procesorom koji ima veću broj frekventnih odnosno naponskih nivoa na kojima može da radi. Naše mišljenje je da se predstavljeni simulator efikasno može iskoristiti u postupku projektovanja energetski efikasnih FT RTS-a.

ZAHVALNOST

Rezultati prikazani u ovom radu ostvareni su u okviru projekta TR-33051 čiju realizaciju finansira Ministarstvo nauke Republike Srbije.

LITERATURA

[1] Nobuyasu Kanekawa, Eishi H. Ibe, Takashi Suga, Yutaka Uematsu, *Dependability in Electronic Systems: Mitigation of Hardware Failures, Soft Errors, and Electro-Magnetic Disturbances*, Springer, 2010.

[2] S. Đošić, M. Jevtić, "Analysis of transient fault tolerance in hard real-time systems with time redundancy", *Facta Universitatis, Series: Automatic control and robotics*, vol. 8, no 1, pp. 149-163, 2009.

[3] S. Đošić, M. Jevtić, "Planiranje zadataka u sistemu za rad u realnom vremenu sa redundansom u vremenu za prevazilaženje otkaza", *Zbornik radova V simpozijuma industrijske elektronike*, INDEL 2004, Banja Luka, pp. 146-149, novembar 2004.

[4] S. Đošić, M. Jevtić, M. Damnjanović, "Analysis of possibilities to overcome the transient faults in real-time systems with time redundancy", XLVI International scientific conference on information, communication and energy systems and technologies, ICEST 2011, *Proceedings of Papers*, volume 2, pp 417- 420, Serbia, Niš, June 29 - July 1, 2011.

[5] K. Woonseok, S. Dongkun, Y. Han-Saem, K. Jihong, M. Sang, "Performance Comparison of Dynamic Voltage Scaling Algorithms for Hard Real-Time Systems", *Proceedings of the Eighth IEEE Real-Time and Embedded Technology and Applications Symposium* (RTAS'02*)*, pp. 219 – 228, 2002.

[6] A. S. Ahmadian, M. Hosseingholi, A. Ejlali, "A Control-Theoretic Energy Management for Fault-Tolerant Hard Real-Time Systems", *2010 IEEE International Conference on Computer Design*, pp. 173-178, 2010.

[7] R. M. Santos, J. Santos, J. D. Orozco, "Power saving and fault-tolerance in real-time critical embedded system", *Journal of system Architecture 55*, pp. 90-101, 2009.

[8] Ping Zhu, Fumin Yang, Gang Tu, Wei Luo, "Fault-Tolerant Scheduling for Periodic Tasks based on DVFS", *Proceedings of the 9th International Conference for Young Computer Scientists*, pp. 2186 – 2191, 2008.

[9] Z. Dakai, R. Melhem, D. Mosse, "The Effects of Energy Management on Reliability in Real-Time Embedded Systems", *Proceedings of the 2004 IEEE/ACM International conference on Computer-aided design*, pp. 35-40, 2004.

[10] Sandra Đošić, Milun Jevtić, „Mogućnosti prevazilaženja prolaznih otkaza kod RTS-a sa redundansom u vremenu", ETRAN 2011, *Zbornik radova 55. Konferencije za ETRAN*, EL 2.1-1-4, Banja Vrućica, 6-9. juna 2011.

[11] S. Đošić, M. Jevtić, "Dynamic voltage scaling for real-time systems under fault tolerance constraints", accepted for publication on 28th International Conference on Microelectronics, MIEL2012.

[12] Locke, C. D., Vogel, D. R., Mesler, T. J., "Building a Predictable Avionics Platform in Ada: A Case Study", *Proceedings of IEEE Real-Time Systems Symposium*, pp. 181–189, 1991.

[13] Ying Zhang, Krishnendu Chakrabarty, "Task Feasibility Analysis and Dynamic Voltage Scaling in Fault-Tolerant Real-Time Embedded Systems" *Proceedings of Design, Automation and Test in Europe Conference and Exhibition*, Vol.2, pp. 1170 – 1175, 2004.

Abstract – In this paper we analyze the energy efficiency of fault tolerant real-time systems. We consider real-time systems where the fault tolerance is achieved running the task affected by a transient fault again using time redundancy. The energy efficiency analysis is done using one heuristic-based dynamic voltage and frequency scaling algorithm which we designed. The focus of our analysis is the trade-off between energy consumption and fault tolerance for one real-time task set.

INCREASING THE ENERGY EFFICIENCY OF FAULT TOLERANT REAL-TIME SYSTEM

Sandra Đošić, Milun Jevtić