

POLSKA AKADEMIA NAUK
KOMITET ELEKTRONIKI I TELEKOMUNIKACJI

KWARTALNIK
ELEKTRONIKI I TELEKOMUNIKACJI

ELECTRONICS AND
TELECOMMUNICATIONS
QUARTERLY

TOM 47 — ZESZYT 4

WYDAWNICTWO NAUKOWE PWN
WARSZAWA 2001

RADA REDAKCYJNA

Przewodniczący
Prof. dr hab. inż. STEFAN HAHN
czł. koresp. PAN

Członkowie
prof. dr hab. inż. DANIEL JÓZEF BEM — czł. koresp. PAN, prof. dr hab. inż. MICHAŁ BIAŁKO — czł. rzecz. PAN, prof. dr hab. inż. JAN CHOJCAN, prof. dr hab. inż. MAREK DOMAŃSKI, prof. dr hab. inż. ANDRZEJ HAŁAS, prof. dr inż. WŁADYSŁAW MAJEWSKI, prof. dr hab. inż. JÓZEF MODELSKI, prof. dr inż. JERZY OSIOWSKI, prof. dr hab. inż. EDWARD SĘDEK, prof. dr hab. inż. MICHAŁ TADEUSIEWICZ, prof. dr hab. inż. WIESŁAW WOLIŃSKI — czł. koresp. PAN, prof. dr inż. MARIAN ZIENTALSKI

REDAKCJA
Redaktor Naczelny
prof. dr hab. inż. WIESŁAW WOLIŃSKI

Zastępca Redaktora Naczelnego
doc. dr inż. KRYSTYN PLEWKO

Sekretarz Odpowiedzialny
mgr ELŻBIETA SZCZEPANIAK

ADRES REDAKCJI
00-665 Warszawa, ul. Nowowiejska 15/19 Politechnika, pok. 470
Instytut Telekomunikacji, Gmach im. prof. JANUSZA GROSZKOWSKIEGO

Dyżury Redakcji: środy i piątki, godz. 14–16
tel/fax (022) 660 77 37

Telefony domowe: Redaktora Naczelnego: 812 17 65
Zast. Red. Naczelnego: 826 83 41
Sekretarza Odpowiedzialnego: 633 92 52

WYDAWNICTWO NAUKOWE PWN SA
Warszawa, ul. Miodowa 10

Ark. wyd. 14,75 Ark. druk. 10,75	Podpisano do druku w październiku 2001 r.
Papier offset. kl. III 80 g. B-1	Druk ukończono w listopadzie 2001 r.

SKŁAD I ŁAMANIE: PHOTOTEXT, Warszawa, ul. Chmielna 120
DRUK I OPRAWA: Zakład Poligraficzny Oja Druk, ul. Sporna 2F, Warszawa

Szanowni Autorzy

„Kwartalnik Elektroniki i Telekomunikacji” — Electronics and Telecommunications Quarterly jest kontynuatorem tradycji powstałego 47 lat temu kwartalnika p.t. „Rozprawy Elektrotechniczne”.

Kwartalnik jest czasopismem Komitetu Elektroniki i Telekomunikacji Polskiej Akademii Nauk. Wydawany jest przez Wydawnictwo Naukowe PWN SA w Warszawie. Kwartalnik jest czasopismem naukowym, na którego łamach są publikowane artykuły i komunikaty prezentujące wyniki oryginalnych prac teoretycznych i doświadczalnych, a także przeglądowych. Związane są one z szeroko rozumianymi dziedzinami współczesnej elektroniki, telekomunikacji, mikroelektroniki, optoelektroniki, radiotechniki i elektroniki medycznej.

Autorami publikacji są wybitni naukowcy, znani specjaliści o wieloletnim doświadczeniu, a także młodzi badacze — głównie doktoranci.

Artykuły charakteryzują się oryginalnym ujęciem zagadnienia, interesującymi wynikami badań, krytyczną oceną teorii lub metod, omówieniem aktualnego stanu, lub postępu danej gałęzi techniki oraz omówieniem perspektyw rozwojowych. Sposób pisania matematycznej części artykułów zgodny jest z wytycznymi IEC (International Electronics Commission) oraz ISO (International Organization of Standardization).

Wszystkie publikowane w Kwartalniku artykuły są recenzowane przez znanych krajowych specjalistów, co zapewnia że publikacje te są uznawane jako autorski dorobek naukowy. Opublikowanie w kwartalniku wyników prac naukowych zrealizowanych w ramach „GRANTów” Komitetu Badań Naukowych spełnia więc jeden z wymogów stawianych tym pracom.

Czasopismo dociera do wszystkich zajmujących się elektroniką i telekomunikacją krajowych ośrodków naukowych oraz technicznych, a także szeregu instytucji zagranicznych. Jest ponadto prenumerowane przez liczne grono specjalistów i biblioteki.

Każdy Autor otrzymuje bezpłatnie 20 egzemplarzy nadbitek swojego artykułu, co ułatwia przesłanie go do indywidualnych wybranych przez Autora osób i instytucji w kraju lub za granicą. Ułatwia to dodatkowo fakt, że w Kwartalniku są publikowane artykuły w języku angielskim.

Nadesłane do redakcji artykuły są publikowane w terminie około pół roku, w przypadku sprawnej współpracy Autora z Redakcją. Wytyczne dla Autorów dotyczące formy publikacji są zamieszczone w zeszytach Kwartalnika, można je także otrzymać w siedzibie Redakcji.

Artykuły można dostarczać osobiście, lub pocztą pod adresem zamieszczanym na stronie redakcyjnej w każdym zeszycie.

Redakcja

Wybrane zagadnienia analizy i projektowania rozproszonych systemów komputerowych czasu rzeczywistego

MIROSŁAW GAJER

*Katedra Automatyki AGH
Al. Mickiewicza 30, 30-059 Kraków
e-mail: mgajer@ia.agh.edu.pl*

*Otrzymano 2001.02.23
Autoryzowano 2001.07.27*

W artykule dokonano przeglądu rozwiązań stosowanych w analizie i projektowaniu współczesnych systemów komputerowych czasu rzeczywistego. W artykule wprowadzono podstawowe pojęcia i koncepcje związane z dziedziną przetwarzania danych w czasie rzeczywistym. Szczególną uwagę zwrócono na te zagadnienia, które posiadają szczególne znaczenia z punktu widzenia niezawodności i bezpieczeństwa działania systemów czasu rzeczywistego. W artykule zamieszczono także krótką dyskusję rozwiązań zaproponowanych w literaturze przedmiotu. W artykule poruszono zagadnienia dotyczące systemów operacyjnych dla rozproszonych systemów komputerowych czasu rzeczywistego. Poruszono także zagadnienia związane z synchronizacją w takich systemach. Artykuł zakończono dyskusją o potencjalnych zastosowaniach rozproszonych systemów czasu rzeczywistego.

Słowa kluczowe: systemy czasu rzeczywistego, systemy operacyjne, szeregowanie i alokacja zadań, odporność na błędy, zastosowania.

1. WPROWADZENIE

System komputerowy czasu rzeczywistego może zostać zdefiniowany jako system, którego główną miarą wydajności jest to, czy spełnia on ściśle określone ograniczenia czasowe (Task Deadlines). Systemy takie mogą zostać sklasyfikowane w kategoriach ostrych (Hard Real-Time) i słabych (Soft Real-Time) ograniczeń czasowych.

W ostatnich kilku latach można było zaobserwować wzrost liczby różnorodnych procesów występujących głównie w przemyśle i transporcie, które są zbyt złożone lub zbyt szybkozmienne, aby mogły podlegać bezpośredniemu nadzorowi człowieka. Jako przykłady można wymienić tutaj reaktory atomowe, instalacje chemiczne, sieci dystrybucji energii elektrycznej, samoloty, statki kosmiczne oraz linie produkcyjne, które wykorzystują roboty. W aplikacjach z dziedziny automatycznej kontroli, komputer jest

wpięty w pętlę sprzężenia zwrotnego sterowanego procesu. Zgodnie z elementarnymi zasadami teorii sterowania jakość sterowania jest bardzo czuła na opóźnienia powstające w torze sprzężenia zwrotnego. Zatem jeżeli opóźnienie w torze sprzężenia zwrotnego, które odpowiada bezpośrednio czasowi wykonania danego zadania obliczeniowego przekroczy pewien punkt graniczny (Deadline), wówczas cały proces może wymknąć się spod kontroli. Takie aplikacje, w których naruszenie przez komputer pewnych określonych dla zadań obliczeniowych limitów czasowych może być powodem katastrofy, utraty życia ludzkiego lub poważnych strat finansowych są znane jako systemy z ostrymi ograniczeniami czasowymi (Hard Real-Time Systems).

Dla porównania istnieją takie aplikacje, jak transmisja mowy lub obrazów, które również posiadają ograniczenia czasowe, ale nie są one krytyczne. Dla przykładu można zdefiniować sytuację awaryjną w systemie transmisji mowy powstającą w przypadku nie przybycia pakietów sygnałów w ściśle określonych limitach czasowych. Jednakże w większości zastosowań takie załamanie funkcjonowania systemu nie niesie za sobą żadnych konsekwencji natury katastroficznej. Systemy takie są zwane systemami ze słabymi ograniczeniami czasowymi (Soft Real-Time Systems).

Wyzwania, którym musi stawić czoło projektant systemów czasu rzeczywistego są całkiem odmienne od tych, z którymi musi się uporać projektant systemów komputerowych ogólnego przeznaczenia (General-Purpose Computers). Po pierwsze w systemach czasu rzeczywistego czas jest znacznie ważniejszym parametrem. W istocie, niezawodność takich systemów może zostać w sposób obiektywny zdefiniowana, jako zdolność do dostarczenia pewnej ilości wyników operacji obliczeniowych w danym przedziale czasu, tak aby nie zostały naruszone ograniczenia czasowe. Powyższe ograniczenia czasowe są wynikiem umieszczenia komputera w pętli sprzężenia zwrotnego kontrolowanego procesu. Szczegółowe omówienie tego zagadnienia można znaleźć w pracy [1].

Po drugie, środowisko pracy systemu komputerowego jest zwykle bardziej nieprzyjazne. Systemy komputerowe ogólnego przeznaczenia są zwykle umieszczane w biurach lub salach komputerowych, które charakteryzują się zwykle małymi przedziałami wahań temperatury i wilgotności. Dla kontrastu, systemy czasu rzeczywistego muszą często pracować w środowiskach odznaczających się znacznymi wahaniami temperatury, oddziaływaniami mechanicznymi (takimi jak wysokie przeciążenia grawitacyjne w niektórych zastosowaniach kosmicznych i militarnych) oraz wpływem promieniowania zarówno cząstek elementarnych, jak i elektromagnetycznego.

W rezultacie wydajność obliczeniowa systemów czasu rzeczywistego musi być znaczenie bardziej ostrożnie modelowana. Systemy, które mogą spowodować utratę życia ludzkiego w przypadku, kiedy zawiodą muszą często uzyskać certyfikat, potwierdzający ich zgodność z pewnymi minimalnymi standardami. Zgromadzenie dostatecznej liczby danych eksperymentalnych, w celu ustalenia czy system spełnia bardzo wysokie wymagania niezawodności zajęłoby zbyt wiele czasu. Zatem badania niezawodności takich systemów muszą zostać wykonane w oparciu o odpowiednie modele.

2. CHARAKTERYSTYKA SYSTEMÓW KOMPUTEROWYCH CZASU RZECZYWISTEGO

Wszystkie najważniejsze pojęcia związane z systemami czasu rzeczywistego zostały dobrze zdefiniowane w pracy [2]. Podsumowując zamieszczone tam rozważania, można powiedzieć, że istnieją trzy podstawowe składowe oraz cała gama ich współzależności, które odróżniają systemy czasu rzeczywistego od systemów komputerowych ogólnego przeznaczenia. Po pierwsze, w systemach czasu rzeczywistego czas jest najbardziej cennym zasobem podlegającym zarządzaniu. Wszystkie zadania obliczeniowe muszą zostać przydzielone do procesorów i następnie tak uszeregowane aby nie zostały naruszone ich ograniczenia czasowe. Wiadomości wymieniane pomiędzy współdziałającymi zadaniami czasu rzeczywistego muszą być wysłane, bądź otrzymane w ściśle określonych przedziałach czasowych. Prawidłowa praca systemu czasu rzeczywistego zależy nie tylko od poprawności logicznej dostarczonych wyników, ale także od czasu, w którym wyniki te zostaną dostarczone. Po drugie, niezawodność systemów czasu rzeczywistego jest niezwykle istotnym parametrem, ponieważ jakakolwiek awaria takiego systemu może być przyczyną katastrofy, poważnych strat ekonomicznych, a nawet utraty życia ludzkiego. Wreszcie środowisko, w którym system komputerowy pracuje jest aktywną składową dla każdego systemu czasu rzeczywistego. Przykładowo, nie ma sensu rozpatrywanie komputera pokładowego jako niezależnego, samodzielnego systemu komputerowego, ponieważ łącznie z samolotem stanowi on pewną niepodzielną i współdziałającą ze sobą całość.

Aplikacje czasu rzeczywistego składają się zwykle ze zbioru współpracujących ze sobą zadań. Zadania te są przeważnie wywoływane i aktywowane w regularnych odstępach czasu oraz posiadają ograniczenia czasowe, przed upływem, których muszą zakończyć swoje wykonywanie. Po każdym wywołaniu, zadanie bada stan systemu, dokonuje pewnych obliczeń oraz jeżeli jest to konieczne wysyła pewne komendy, mające na celu zmianę, bądź wyświetlenie stanu systemu. Dla przykładu, w aplikacji związanej z dziedziną motoryzacji, zadanie może odczytywać wartość ciśnienia w układzie hamulcowym samochodu oraz indywidualne prędkości obrotowe wszystkich kół, następnie dokonywać pewnych obliczeń, mających na celu określenie, czy któreś z kół nie zostało zablokowane i tym samym czy nie nastąpił poślizg oraz w przypadku wystąpienia poślizgu podejmować akcję przeciwoślizgową, polegającą na zmianie pozycji odpowiednich zaworów układu hamulcowego. Podobnie w aplikacji związanej z dziedziną lotnictwa, zadanie może monitorować aktualne położenie przepustnicy silnika, dokonywać obliczeń wyznaczających aktualne położenie samolotu, a następnie dokonywać zmiany siły ciągu silnika poprzez zmianę ilości wtryskiwania paliwa.

Opisane powyżej zadania i inne podobne do nich nazywane są zadaniami periodycznymi, bądź okresowymi (Periodic Task). Cechą wspólną wszystkich zadań periodycznych jest to, że są one krytyczne ze względu na czas obliczeń, w tym sensie, że system nie może poprawnie funkcjonować bez zakończenia ich przed upływem ograniczeń czasowych. Przykładowo, w aplikacji związanej z pojazdem samochodowym, jeżeli zadanie nie podejmie akcji przeciwoślizgowej po upływie krótkiego odcinka czasu od

momentu, w którym koło zostało zablokowane, jest rzeczą bardzo prawdopodobną, że samochód wpadnie w poślizg, który w rezultacie może zakończyć się wypadkiem. Podobnie w aplikacji związanej z samolotem, jeżeli położenie przepustnicy nie zostanie skorygowane na czas, samolot może się roztrzaskać na jakichś przeszkodach terenowych, powodując utratę życia znajdujących się w nim osób. Tak więc system komputerowy czasu rzeczywistego musi zapewnić, że wszystkie ograniczenia czasowe zadań krytycznych ze względu na czas będą zachowane, niezależnie od innych zdarzeń, które mogą wystąpić w systemie.

Jest rzeczą oczywistą, że nie wszystkie zadania przybywają do systemu czasu rzeczywistego w regularnych odstępach czasu. Pewne zadania są aktywowane jedynie w przypadku zaistnienia pewnych zdarzeń. Zadania takie nazywane są zadaniami aperiodycznymi (Aperiodic Tasks). Przykładowo, zadanie mające na celu rekonfigurację systemu, może być aktywowane tylko w przypadku wykrycia jakiegoś poważnego błędu w pracy systemu. Ponieważ zdarzenia nie zawsze muszą pojawiać się w regularnych interwałach czasowych, odpowiadające im zadania, także nie są aktywowane w regularnych przedziałach czasu. Jeżeli zdarzenie wymagające obsługi jest krytyczne ze względu na czas, wówczas odpowiadające im zadanie aperiodyczne będzie posiadało ograniczenie czasowe, przed upływem, którego będzie musiało zakończyć swoje wykonywanie. Z kolei jeśli zdarzenie nie jest krytyczne ze względu na czas obsługi, odpowiadające mu zadanie nie będzie posiadało ograniczenia czasowego, ale i tak musi zostać ono obsłużone tak szybko, jak to jest tylko możliwe, nie powodując przy tym naruszenia ograniczeń czasowych nałożonych na pozostałe zadania.

Bazując na powyższej dyskusji, w systemach czasu rzeczywistego, ograniczenia czasowe zadań mogą zostać sklasyfikowane jako: ostre (Hard), mocne (Firm) oraz słabe (Soft). Przy czym mówi się, że ograniczenie czasowe jest ostre, jeżeli konsekwencje jego nie dotrzymania mogą mieć charakter katastroficzny. Zadania periodyczne posiadają zwykle ograniczenia czasowe należące do tej kategorii. Z kolei mówi się, że ograniczenie czasowe jest mocne, jeżeli wyniki dostarczone przez odpowiadające mu zadanie przestają być użyteczne z chwilą przekroczenia przydzielonego dla tego zadania limitu czasu, ale konsekwencje naruszenia ograniczeń czasowych nie są zbyt poważne w skutkach. Ograniczenia czasowe wielu zadań aperiodycznych można zaliczyć do tej kategorii. Na przykład w pracy [3] opisano przeprowadzanie transakcji w czasie rzeczywistym w systemie baz danych. Natomiast ograniczenia czasowe, które nie są ani ostre, ani mocne, zwane są słabymi. Użyteczność wyników dostarczonych przez zadanie ze słabym ograniczeniem czasowym maleje wraz z upływem czasu, który minął od chwili naruszenia tego ograniczenia czasowego.

W związku z powyższym powstaje naturalne pytanie, z czego wynikają ograniczenia czasowe oraz skąd wiadomo, że dane ograniczenie czasowe należy do kategorii ograniczeń ostrych, mocnych, czy też słabych. Ograniczenia czasowe wynikają z danej aplikacji. Dla przykładu w pracy [4] rozważono system ochrony powietrznej, którego zadaniem jest nieustanna obserwacja danego fragmentu przestrzeni powietrznej i wykrywanie pojawiających się w niej nieprzyjacielskich pocisków rakietowych. Zgodnie z charakterem powyższej aplikacji, ograniczenia czasowe są takie, że nieprzyjacielski

pocisk raketowy musi ulec zniszczeniu po upływie najpóźniej 15 sekund od chwili wykrycia faktu naruszenia przez niego przestrzeni powietrznej. Fakt ten powoduje narzucenie dalszych ograniczeń czasowych na pozostałe zadania, których celem jest detekcja pocisku, jego identyfikacja, przygotowanie do odpalenia pocisku raketowego, mającego na celu zniszczenie intruza oraz odpalenie tego pocisku. Dla przykładu, w niniejszej aplikacji uwarunkowania czasowe są takie, że nieprzyjacielski pocisk musi zostać zidentyfikowany najpóźniej po upływie 0,2 sekundy po jego wykryciu i jeżeli jest to konieczne pocisk mający na celu jego zniszczenie musi zostać przygotowany do odpalenia najpóźniej w ciągu 5 sekund po identyfikacji i odpalony najpóźniej po upływie 0,5 sekundy po zakończeniu czynności przygotowawczych. Zdefiniowane ograniczenia czasowe dla powyższych zadań powodują narzucenie dalszych ograniczeń czasowych na odpowiadające im podzadania, a te z kolei narzucają ograniczenia czasowe na ich podzadania itd.

Powyższy przykład rzuca również światło na inny ważny problem związany z aplikacjami czasu rzeczywistego. Ponieważ każdy nieprzyjacielski pocisk, który naruszył przestrzeń powietrzną musi zostać bezwzględnie zniszczony, zachowanie się komputerowego systemu sterowania w czasie rzeczywistym musi być przewidywalne. To znaczy w trakcie projektowania takiego systemu powinno być możliwym wykazanie, że wszystkie ograniczenia czasowe narzucone na poszczególne zadania będą zachowane, o ile tylko pewne założenia odnośnie warunków pracy systemu będą spełnione. Dla przykładu można przyjąć, że jedynym założeniem poczynionym odnośnie pracy systemu jest całkowita liczba błędów popełnionych przez system w danym przedziale czasu, która musi być mniejsza od pewnej wartości progowej f . Zatem system jest przewidywalny, jeżeli jest możliwym wykazanie, na etapie projektowania, że żadne z ograniczeń czasowych nie zostanie naruszone, pod warunkiem wystąpienia nie więcej niż f błędów. Konieczność zapewnienia determinizmu i niezawodności pracy systemu posiada znaczący wpływ na sposób projektowania systemów czasu rzeczywistego.

Oprócz ograniczeń czasowych i ograniczeń związanych z koniecznością zapewnienia deterministycznego zachowania się systemu, zadania z których zbudowana jest aplikacja czasu rzeczywistego posiadają także inne ograniczenia, takie, które występują również w systemach nie będących systemami czasu rzeczywistego. Przykładowo zadania mogą posiadać:

- ograniczenia zasobów: zadanie może żądać dostępu do pewnych zasobów, innych niż procesor, takich jak: urządzenia wejścia-wyjścia, sieć komunikacyjna, struktury danych, pliki oraz bazy danych
- ograniczenia poprzedzania: zadanie zanim rozpocznie swoje wykonywanie może wymagać dostarczenia rezultatów uzyskanych w wyniku wykonania innych zadań, które będą stanowiły dla niego zbiór danych wejściowych
- ograniczenia niezawodności i wydajności systemu: pewne zadania muszą charakteryzować się określonym z góry stopniem niezawodności, dostępności oraz posiadać żądaną wydajność obliczeniową.

3. DETERMINIZM W SYSTEMACH CZASU RZECZYWISTEGO

Jak już wcześniej zwrócono uwagę, zapewnienie deterministycznego zachowania się systemu jest jednym z kluczowych problemów występujących na etapie projektowania systemów czasu rzeczywistego. Jednakże stopień tego determinizmu może być różny w zależności od typu aplikacji, a także może różnić się pomiędzy poszczególnymi zadaniami w ramach danej aplikacji. W przypadku prostych systemów zapewnienie determinizmu polega na wykazaniu na etapie projektowania systemu, że ograniczenia czasowe zostaną zachowane dla wszystkich zadań ze stuprocentową pewnością. Takie postawienie sprawy wymaga jednakże znajomości charakterystyk wszystkich zadań *a priori*. Przykładowo konieczna jest znajomość *a priori* całkowitej liczby zadań, jak również wymagań odnośnie mocy obliczeniowej i wielkości wykorzystywanych zasobów systemu dla każdego i w każdej chwili czasu. Ponadto konieczna jest dokładna znajomość wszystkich zmian zachodzących w czasie w środowisku systemu, ponieważ środowisko może mieć, w sposób znaczący, wpływ na zachowanie się systemu. Nie trzeba chyba dodawać, że jest rzeczą nieprawdopodobną, aby projektant dysponował tymi wszystkimi informacjami na etapie projektowania systemu.

W przypadku bardziej złożonych systemów znaczenie determinizmu różni się pomiędzy poszczególnymi zadaniami. Dla pewnych zadań krytycznych może wciąż być wymagane zagwarantowanie stuprocentowej pewności, że ich ograniczenia czasowe zostaną zachowane. Zadania periodyczne, posiadające ostre ograniczenia czasowe należą do tej kategorii. Jak pokazano powyżej pełny zbiór charakterystyk tych zadań musi być znany *a priori*. Dla pozostałych zadań wystarczy podanie gwarancji natury probabilistycznej lub określenie deterministycznego zachowania się zadania w trakcie wykonywania danej aplikacji czasu rzeczywistego (Run-Time Deterministic Guaranties). Wyrażenie gwarancje probabilistyczne (Probabilistic Guaranties) może posiadać również wiele znaczeń. W pewnych przypadkach oznacza ono, że tylko dla pewnej części zadań występujących w systemie na pewno nie zostaną naruszone ograniczenia czasowe. Z kolei w pozostałych przypadkach wyrażenie to oznacza, że dla danego zadania istnieje pewien stopień prawdopodobieństwa nie naruszenia jego ograniczeń czasowych. Warto zwrócić uwagę, że w pewnych przypadkach oba powyższe sformułowania są równoznaczne.

Natomiast deterministyczne zachowanie się zadania w trakcie jego wykonywania oznacza, że w momencie aktywacji danego zadania sam system określa, czy zadanie to może zostać zakończone przed upływem jego ograniczenia czasowego, nie powodując jednocześnie naruszenia ograniczeń czasowych pozostałych zadań. Jeżeli dotrzymanie ograniczeń czasowych dla wszystkich zadań może być zapewnione, wówczas dane zadanie jest akceptowane oraz system udziela stuprocentowej gwarancji na dochowanie jego ograniczenia czasowego. Z kolei jeśli nie ma możliwości dochowania ograniczeń czasowych, zadanie nie jest akceptowane przez system. Podsumowując, na etapie projektowania nie można stwierdzić, dla których z zadań będą zachowane ograniczenia czasowe, jednakże już podczas pracy z systemem, dla każdego z zadań można stwierdzić, czy jego ograniczenie czasowe nie zostanie naruszone. Ten typ gwarancji jest często

wykorzystywany dla zadań aperiodycznych napływających w sposób dynamiczny do systemu oraz w celu zapewnienia równomiernego obciążenia wszystkich jednostek obliczeniowych systemu.

Należy mieć świadomość faktu, że udzielenie gwarancji na nienaruszenie ograniczeń czasowych możliwe jest jedynie w przypadku, gdy wszystkie charakterystyki zadań, takie jak czas wykonywania obliczeń i moment aktywacji zadania, dane są *a priori*. W praktyce jest rzeczą trudną uzyskanie dokładnej informacji na temat charakterystyk zadań, zatem przyjmowane są wartości odpowiadające najgorszemu przypadkowi. Wartości takie uzyskiwane są albo na drodze rozważań teoretycznych, albo też są one wynikiem szeroko przeprowadzanych symulacji lub testów systemów. Uzyskane w ten sposób wartości mogą różnić się od rzeczywistych wartości odpowiadających najgorszemu z możliwych scenariuszy, w pewnych rzadkich przypadkach. Jednakże projektanci systemu w dalszym ciągu zmuszeni są wykorzystywać uprzednio przyjęte wartości dla najgorszego z przypadków, ponieważ nie ma innej alternatywy. W pracy [5] opisaną powyżej sytuację nazwano naruszeniem specyfikacji (Specification Violation) oraz zaproponowano implementację systemu, rejestrującego przypadki wystąpienia naruszenia specyfikacji. Dane zebrane przez taki system kontrolny, mogą być później użyte w celu modyfikacji przyjętych wartości odpowiadających najgorszemu z przypadków.

4. ARCHITEKTURY SYSTEMÓW KOMPUTEROWYCH CZASU RZECZYWISTEGO

Zagadnienia projektowania architektur systemów czasu rzeczywistego należy rozpatrywać na dwóch płaszczyznach. Po pierwsze pod uwagę musi zostać wzięty poziom poszczególnych jednostek obliczeniowych, a następnie poziom całego systemu. Na poziomie jednostek obliczeniowych, każdy z procesorów musi dostarczyć odpowiedniej mocy obliczeniowej oraz zapewnić wysoki stopień determinizmu podczas wykonywania zadań należących do aplikacji czasu rzeczywistego, obsługi przerw oraz współpracy ze światem zewnętrznym. W tym celu należy uczynić operacje takie jak: wykonanie instrukcji, dostęp do pamięci i przełączanie kontekstu bardziej przewidywalnymi. Aby te drobnoziarniste operacje stały się bardziej przewidywalne, w systemach czasu rzeczywistego bardzo rzadko korzysta się z pamięci wirtualnych, ponieważ tryb stronicowania pamięci może być przyczyną powstania zupełnie nieprzewidywalnych i bardzo długich opóźnień w dostępie do pamięci. Z tych samych powodów w systemach czasu rzeczywistego unika się raczej stosowania pamięci typu *cache*, ponieważ nieokreśloność aktualnego występowania w pamięci *cache* danego fragmentu kodu programu, bądź danych powoduje powstanie nieprzewidywalnych opóźnień w dostępie do pamięci. Jednakże w praktyce uniknięcie stosowania pamięci *cache* może okazać się bardzo trudne, ponieważ systemy czasu rzeczywistego są często budowane z wykorzystaniem współczesnych mikroprocesorów ogólnego przeznaczenia, w których powszechnie stosowane są wielopoziomowe pamięci *cache*, mające na celu zwiększenie ich średniej wydajności

obliczeniowej. Podobnie zastosowanie techniki przetwarzania potokowego w przypadku pobierania instrukcji kodu programu oraz danych, a także wykorzystywanie mechanizmów predykcji operacji skoku (ang. branch prediction), które są również powszechnie stosowane we współczesnych mikroprocesorach powszechnego użytku, sprawia, że deterministyczne zachowanie się systemu na poziomie poszczególnych jednostek przetwarzających jest trudne do zagwarantowania.

Z kolei na poziomie systemu, problemy związane z komunikacją pomiędzy poszczególnymi jednostkami obliczeniowymi oraz konieczność zapewnienia wysokiej odporności systemu na błędy (ang. fault tolerance), sprawiają, że przewidywalne zachowanie się systemu jest trudne do osiągnięcia. Powyższe problemy są niemożliwe do uniknięcia, ponieważ zapewnienie wysokiej mocy obliczeniowej i dużego stopnia niezawodności jest bardzo często rzeczą konieczną w rozproszonych systemach komputerowych czasu rzeczywistego. Zagadnienia związane z rozproszonymi systemami czasu rzeczywistego oraz stosowane powszechnie w takich systemach rozwiązania, zostały poddane poniżej krótkiej dyskusji.

4.1. ZAGADNIENIA ZWIĄZANE Z ARCHITEKTURĄ WYSOKIEGO POZIOMU

Na najwyższym poziomie architektury, rozproszony system komputerowy czasu rzeczywistego, może być rozpatrywany, jako zbiór jednostek przetwarzających, komunikujących się ze sobą poprzez sieć wzajemnych połączeń. Zgodnie z tym, co pokazano w pracach [6], [7] i [8], każda jednostka przetwarzająca może być z kolei również systemem wieloprocessorowym, zbudowanym z procesorów obsługujących daną aplikację, system, sieć, a także jednostka taka może zawierać segment wspólnej, dzielonej pamięci (ang. shared memory segment) oraz urządzenia wejścia i wyjścia (ang. I/O interface). Podczas gdy procesor obsługujący aplikację może być procesorem ogólnego przeznaczenia, procesory związane z siecią i z systemem są zwykle projektowane specjalnie, zgodnie z określonymi potrzebami (ang. custom-designed), ponieważ dostarczają one wsparcia, o wysokim stopniu specjalizacji, dla konkretnych aplikacji czasu rzeczywistego. Przykłady tego typu rozwiązań można znaleźć w pracy [7]. Podsystem pamięci także często musi być projektowany ze szczególną uwagą, tak aby mógł zapewnić szybką i niezawodną komunikację pomiędzy procesorami danego węzła systemu. Przykładowo na poziomie pojedynczego węzła systemu rozproszonego, podsystem pamięci może posiadać wbudowany mechanizm komórek kontaktowych (ang. mailbox facility), w celu zapewnienia efektywnej komunikacji między poszczególnymi procesorami systemu.

Z kolei poszczególne węzły systemu muszą być połączone przez odpowiednią sieć komunikacyjną. Jak można zaobserwować w pracach [6], [9] i [10] we wcześniejszych i małych systemach sieć połączeń była zaprojektowaną specjalnie magistralą przesyłową (ang. custom-designed broadcast bus) ze znaczną redundancją, mającą na celu zapewnienie wymaganego stopnia niezawodności. Jednakże w nowocześniejszych systemach sieć połączeń jest albo charakteryzującą się dużą szybkością przesyłu danych siecią z krążącym żetonem (ang. token ring), albo też stosowana jest strategia połączeń punkt

do punktu (ang. point-to-point) o odpowiednio dobranej topologii. Dla przykładu opisany w pracy [11] system o nazwie „Spring”, zrealizowany na Uniwersytecie w Massachusetts, wykorzystuje odznaczającą się dużą szybkością sieć optyczną o nazwie „Scramment”. Z kolei opisany w pracach [7] i [12] projekt o nazwie „HARTS”, zrealizowany na Uniwersytecie w Michigan wykorzystywał sieć połączeń typu punkt do punktu zwaną C-kształtną heksagonalną topologią siatkową (ang. C-wrapped hexagonal mesh topology).

Niezależnie od zastosowanej aktualnie topologii sieć połączeń powinna odznaczać się skalowalnością, łatwością implementacji i niezawodnością. Ponadto sieć taka powinna umożliwiać efektywną komunikację, zarówno w przypadku połączeń tylko pomiędzy dwoma węzłami (ang. one-to-one), jak i pomiędzy danym węzłem i wszystkimi pozostałymi (ang. one-to-many). Przykładowo dla C-kształtnej heksagonalnej topologii siatkowej, zaimplementowanej w systemie „HARTS”, opracowano specjalny algorytm, w celu obliczania najkrótszej ścieżki, łączącej dwa dowolne węzły systemu. Informacja dotycząca wszystkich najkrótszych ścieżek została w łatwy sposób zakodowana przy użyciu trzech liczb całkowitych, dołączonych do każdej z wiadomości, jako jej część, dzięki czemu węzły pośredniczące w transmisji nie muszą wykonywać zbyt wielu działań. Z kolei w pracy [13] pokazano w jaki sposób algorytm zestawiający połączenie (ang. routing algorithm) może w pełni wykorzystać zaawansowane techniki przełączania (ang. switching techniques), takie jak wirtualne przecięcie (ang. virtual cut-through) oraz łączenie tunelowe (ang. wormhole-routing), dzięki którym przesyłane pakiety nie zawsze muszą być buforowane w węzłach pośredniczących, zanim zostaną przekazane do następnego węzła leżącego na ich linii transmisji. Podobnie przesyłanie wiadomości może również zostać zorganizowane w sposób wydajny i odporny na powstające ewentualnie błędy, dzięki użyciu rozdzielnich ścieżek (ang. disjoint paths), łączących dowolne dwa węzły systemu. Opisane powyżej właściwości są bardzo istotne, ponieważ zapewnienie niezawodnej oraz posiadającej ograniczenia czasowe wymiany informacji, jest sprawą kluczową w przypadku rozproszonego wykonywania jakiejkolwiek aplikacji czasu rzeczywistego.

4.2. ZAGADNIENIA ZWIĄZANE Z ARCHITEKTURĄ NISKIEGO POZIOMU

Zagadnienia związane z architekturą niskiego poziomu koncentrują się zwykle wokół zagadnień przetwarzania pakietów, zestawiania połączeń (ang. routing) oraz kontrolą przepływu danych i powstałych ewentualnie błędów (ang. error and flow control). W rozproszonych systemach komputerowych czasu rzeczywistego pojawiają się jeszcze dodatkowe zagadnienia, związane z koniecznością zapewnienia dochowania ograniczeń czasowych, zarządzania czasem (ang. time management) oraz innymi zasobami systemu. Ponieważ obsługa zadań niskiego poziomu powoduje opóźnienie zadań związanych z konkretną aplikacją, węzły w rozproszonych systemach czasu rzeczywistego posiadają zwykle specjalny procesor, przewidziany do obsługi tych zadań. Procesory takie oznaczane są skrótem NP i nazywane są procesorami sieciowymi (ang. network processors). Głównie zadanie spełniane przez procesory sieciowe polega na dostarczeniu

wiadomości pochodzącej z zadania, będącego jej źródłem do jej miejsca przeznaczenia. W szczególności, jeżeli zadanie związane z daną aplikacją, pragnie wysłać wiadomość, wówczas dostarcza ono procesorowi sieciowemu wszystkich koniecznych informacji na temat odbiorcy tej wiadomości oraz lokalizacji danych związanych z tą wiadomością, a następnie procesor sieciowy przejmuje dalszą kontrolę nad całym procesem, zapewniając, że wiadomość dotrze do miejsca swego przeznaczenia w pewny i uwarunkowany czasowo sposób. Jak pokazano w pracy [14] procesor sieciowy jest w pełni odpowiedzialny za usługi dostarczane przez warstwę transportową, sieci oraz warstwę łącza danych.

Na poziomie warstwy transportowej sieci procesor sieciowy odpowiedzialny jest za zestawienie połączeń pomiędzy węzłami źródła i przeznaczenia. Musi on także zapewnić obsługę powstałych ewentualnie błędów oraz zarządzić retransmisję przekłamaną wiadomości. Z kolei na poziomie warstwy sieci, procesor sieciowy musi podejmować decyzję do wyboru połączeń podstawowych, bądź alternatywnych, następnie alokować odpowiednie pasmo przepustowości (ang. bandwidth), tak aby wiadomość mogła być dostarczona na czas, a także dzielić informację na mniejsze bloki danych i segmenty oraz ponownie łączyć przesłane pakiety w węźle przeznaczenia. W przypadku połączeń realizowanych punkt do punktu, procesor sieciowy musi pozwalać na implementację oraz dokonywać wyboru odpowiedniej metody przełączania pakietów, jak na przykład wirtualne tunelowanie (ang. virtual cut-through wormhole routing), odbieranie pakietów w węźle i posyłanie ich dalej (ang. store-and-forward) oraz przełączanie linii przesyłania pakietów (ang. circuit switching), jak opisano to w pracy [7]. Z kolei w pracach [15], [16] i [17] omówiono sieć wykonaną w postaci okręgu z krążącym żetonem (ang. token ring), gdzie procesor sieciowy musiał dobierać odpowiednie parametry protokołu przesyłania wiadomości, w taki sposób aby mogło być zagwarantowane spełnienie ograniczeń czasowych dla wszystkich przesyłanych wiadomości. Z kolei na poziomie warstwy łącza danych, procesor sieciowy jest odpowiedzialny za zapewnienie wiadomościom dostępu do sieci. Procesor ten musi zapewnić organizację wiadomości w ramki (ang. framing), ich synchronizację oraz odpowiednią sekwencję przesyłanych pakietów.

Procesor sieciowy musi także zapewnić obsługę przerw występujących na wielu poziomach, aby mógł on sprawnie zarządzać wiadomościami o różnym stopniu ważności (ang. different priority levels). Przy czym rozwiązania sprzętowe w obrębie takiego procesora powinny dostarczać wymaganej liczby poziomów przerw, tak aby pilniejszym wiadomościom można było nadać wyższy priorytet niż wiadomościom mniej ważnym. Procesor sieciowy musi również implementować mechanizmy zarządzania buforami (ang. buffer management policies), które mają na celu maksymalizację wykorzystania przestrzeni buforowej, przy jednoczesnym zagwarantowaniu dostępności buforów dla wiadomości o najwyższym priorytecie. Analogicznie, jeżeli pewne nie krytyczne wiadomości wykorzystują inne zasoby systemu, które są aktualnie potrzebne wiadomościom bardziej krytycznym, wówczas procesor sieciowy musi dostarczyć środków koniecznych do wyłączenia takich zasobów, w celu ich wykorzystania przez wiadomości o większym stopniu ważności.

Od procesora sieciowego często jest także wymagane nadzorowanie stanu sieci, ze względu na jej obciążenie (ang. traffic load) oraz detekcję błędów połączenia (ang. link failures). Stopień obciążenia sieci posiada wpływ na zdolność procesora sieciowego do przesłania w czasie rzeczywistym wiadomości do innych procesorów, podczas gdy występowanie błędów połączenia wpływa na stopień niezawodności systemu. Procesor sieciowy może mieć także za zadanie nadzorowanie obciążenia systemu przez komputer główny (ang. host) oraz wykorzystywanie informacji o obciążeniu systemu, w celu wyrównywania obciążenia poszczególnych jednostek (ang. load balancing and sharing) oraz migracji zadań.

4.3. ZAGADNIENIA ZWIĄZANE Z ARCHITEKTURĄ URZĄDZEŃ I/O

Dokonując przeglądu literatury poświęconej rozproszonym systemom komputerowym czasu rzeczywistego można stwierdzić, że większość prac z tej dziedziny koncentruje się głównie na opracowywaniu rozwiązań dotyczących sieci połączeń komunikacyjnych oraz podstawowych zasad i algorytmów przetwarzania danych i komunikacji w takich systemach. Niewiele informacji można natomiast znaleźć na temat rozwiązań w dziedzinie podsystemów wejścia i wyjścia, które przecież w systemach czasu rzeczywistego spełniają niezwykle ważną rolę. Jest bowiem rzeczą oczywistą, że nawet najbardziej wydajny system komputerowy czasu rzeczywistego, nie będzie mógł przetwarzać danych szybciej, niż może je pozyskać ze swoich układów akwizycyjnych. W przypadku systemów czasu rzeczywistego urządzeniami wejścia i wyjścia są różnego rodzaju czujniki (ang. sensors), urządzenia wykonawcze (ang. actuators) oraz urządzenia, których zadaniem jest wyświetlanie i zobrazowanie stanu systemu (ang. display). Jest to sytuacja zupełnie odmienna, od systemów komputerowych ogólnego przeznaczenia, gdzie rolę omówionych powyżej urządzeń spełniają zwykle taśmy i dyski magnetyczne. Ze względu na odmienne wymagania czasowe i niezawodnościowe rozwiązania spotykane w systemach komputerowych ogólnego przeznaczenia najczęściej nie nadają się do zastosowania w systemach czasu rzeczywistego.

W przypadku implementacji w rozproszonym systemie komputerowym czasu rzeczywistego scentralizowanych urządzeń I/O mogą pojawić się problemy związane z ich dostępnością. Zatem urządzenia I/O muszą być zrealizowane w wersji rozproszonej oraz zarządzanie nimi powinno zostać powierzone relatywnie prostym, ale niezawodnym sterownikom. Ponadto w celu zwiększenia dostępności tych urządzeń oraz wydajności ich pracy, urządzenia I/O powinny posiadać możliwość wielokrotnego dostępu (ang. multiple acces paths).

W pracy [18] opisano jedno z możliwych rozwiązań zaimplementowanych w systemie „HARTS”, w celu zapewnienia wielodostępności urządzeń I/O. W rozwiązaniu tym urządzenia I/O zostały zebrane w grupy (ang. clusters) do których prawa dostępu udziela specjalny kontroler. Kontroler posiada zbiór dwukierunkowych łączy (ang. full-duplex links) umożliwiających dostęp do pewnych węzłów systemu rozproszonego. W celu ograniczenia liczby połączeń dla każdego z kontrolerów, przy jednoczesnym

zapewnieniu wielokrotnego dostępu, kontroler posiada połączenie z trzema węzłami systemu. Ponieważ każdy z kontrolerów może być osiągalny przez trzy węzły, w celu obsługi żądań pochodzących z układów I/O zaproponowano różnorodne protokoły zarządzania kontrolerem. W przypadku statycznym jednemu z węzłów powierzana jest pierwszoplanowa odpowiedzialność za zarządzanie kontrolerem, z zastrzeżeniem, że pozostałe węzły mogą przejąć kontrolę w przypadku awarii rozważanego węzła. Natomiast w przypadku dynamicznym, wszystkie trzy węzły uczestniczą w zarządzaniu kontrolerem przy wykorzystaniu bardziej skomplikowanego protokołu.

W pracach [19] i [20] zaproponowano alternatywne rozwiązanie problemu połączenia kontrolerów z węzłami systemu. W tym przypadku kontroler O/I został podłączony do tylko jednego z węzłów. Jednakże rozmieszczenie kontrolerów urządzeń I/O zostało zorganizowane w taki sposób, że każdy z węzłów, który nie posiada połączenia z kontrolerem jest bezpośrednim sąsiadem węzła mającego dostęp do kontrolera. W celu zapewnienia większej niezawodności systemu zaproponowano również schematy rozmieszczenia kontrolerów w taki sposób, aby każdy z węzłów, który nie posiada połączenia z kontrolerem był bezpośrednim sąsiadem j węzłów mających dostęp do kontrolera. Przy czym j jest parametrem projektowym, od którego zależy stopień niezawodności systemu.

Pomimo, że omówione skrótowo powyższe zagadnienia stanowią znaczny krok naprzód w dziedzinie rozwiązań dotyczących architektur urządzeń I/O systemów czasu rzeczywistego, w kwestii tej pozostaje wciąż wiele do zrobienia.

5. SYSTEMY OPERACYJNE CZASU RZECZYWISTEGO

W przeciwieństwie do większości tradycyjnych systemów operacyjnych, determinizm jest jednym z głównych wymagań czynionych odnośnie systemów operacyjnych czasu rzeczywistego. Z tego powodu wiele podstawowych zasad w oparciu, o które konstruowane są systemy operacyjne ogólnego przeznaczenia (ang. general-purpose operating systems) nie znajduje zastosowania w przypadku systemów operacyjnych czasu rzeczywistego. Na przykład w przypadku systemów operacyjnych czasu rzeczywistego nie jest zbyt ważną sprawą implementacja rozbudowanego systemu zarządzania plikami, pamięcią wirtualną, czy też zapewnienie wysokiego stopnia zabezpieczenia przed niepowołanym dostępem zasobów systemu. Natomiast, w systemach czasu rzeczywistego, jest sprawą istotną zapewnienie możliwości szybkiego przełączania kontekstu wykonywanych zadań (ang. context switching), szybkiej i przewidywalnej obsługi przerw (ang. interrupt handling), a także umożliwienie implementacji algorytmów szeregowania zadań, uwzględniających zarówno ograniczenia czasowe poszczególnych zadań, jak i występujące pomiędzy nimi zależności obliczeniowe.

Wszystkie systemy operacyjne, które znajdują zastosowanie we współczesnych systemach czasu rzeczywistego można podzielić na trzy kategorie: systemy operacyjne tworzone specjalnie dla potrzeb konkretnych aplikacji (ang. proprietary kernels), systemy operacyjne powszechnego użytku z dobudowanymi właściwościami umożliwiającymi

cymi ich pracę w czasie rzeczywistym (ang. commercial operating systems with real-time extensions) oraz systemy operacyjne zorientowane badawczo (ang. research-oriented operating systems). Systemy operacyjne tworzone z myślą o danej aplikacji są niewielkie, ale za to odznaczają się dużą szybkością działania. W systemach takich determinizm jest zapewniony dzięki obsłudze tylko tych zadań podstawowych (ang. primitives), dla których może być wykazane, że posiadają ograniczony czas wykonywania. Systemy takie są głównie przewidziane dla małych wbudowanych (ang. embedded) aplikacji, sprawujących kontrolę nad przyrządami pomiarowymi, urządzeniami peryferyjnymi (ang. intelligent peripherals) oraz pewnymi prostymi procesami. Z kolei komercyjne systemy operacyjne z dobudowanymi rozszerzeniami czasu rzeczywistego oferują powszechnie znane interfejsy, przy jednoczesnym zapewnieniu umiarkowanych możliwości implementacji mechanizmów typowych dla czasu rzeczywistego. W ogólnym przypadku systemy takie są wolniejsze oraz ich działanie jest w znacznie mniejszym stopniu przewidywalne, niż ma to miejsce w przypadku systemów zaprojektowanych specjalnie pod kątem konkretnych aplikacji czasu rzeczywistego. Jednakże ze względu na powszechną znajomość oferowanych przez takie systemy interfejsów, nadają się one szczególnie dobrze do tworzenia dużych aplikacji, w których konsekwencje przekroczenia pewnej niewielkiej liczby ograniczeń czasowych nie są zbyt poważne. W przeciwieństwie do obu omówionych powyżej systemów przy konstruowaniu systemów operacyjnych zorientowanych badawczo, główne działania koncentrują się wokół zagadnień specyficznych dla systemów czasu rzeczywistego. W takich systemach istnieje bardzo ograniczona możliwość implementacji mechanizmów, na których podczas tworzenia systemu nie zostały skoncentrowane główne wysiłki.

6. KOMUNIKACJA W SYSTEMACH CZASU RZECZYWISTEGO

Dla przykładu rozważony zostanie system czasu rzeczywistego, składający się z trzech współpracujących ze sobą zadań T_1 , T_2 i T_3 . Niech zadanie T_1 będzie odpowiedzialne za odczytanie aktualnej wartości dostarczanej przez czujniki (ang. sensors) pewnego urządzenia pomiarowego oraz za przekazanie jej do zadania T_2 . Z kolei zadanie T_2 niech będzie odpowiedzialne za wykonanie obliczeń zgodnie z modelem sterowanego procesu i przesłanie ich rezultatów do zadania T_3 . Wreszcie, niech zadanie T_3 odpowiedzialne będzie za kontrolowanie stanu urządzeń wykonawczych (ang. actuators), bazując na rezultatach dostarczonych przez zadanie T_2 . Ponadto zostało założone, że wszystkie rozważane operacje muszą być wykonywane w sposób periodyczny, a okresem równym R sekund. Ponieważ czujniki, jednostki obliczeniowe oraz urządzenia wykonawcze nie muszą wcale znajdować się blisko siebie, w sensie odległości fizycznej, omówione powyżej zadania mogą być wykonywane w trzech różnych węzłach systemu rozproszonego. Zatem zadania T_1 i T_2 muszą przesyłać wiadomości zaraz po zakończeniu swego wykonywania, a także zadanie T_3 musi zakończyć swoje wykonywanie przed upływem ograniczenia czasowego $D \leq R$.

Ten prosty przykład uświadamia ważność ograniczeń czasowych narzuconych na komunikację w systemach czasu rzeczywistego. Aby zapewnić zakończenie zadania T_3 przed upływem D jednostek czasu, nie tylko wykonywanie zadań T_1 i T_2 musi zostać zakończone w odpowiednim czasie, ale także informacje pomiędzy zadaniami T_1 i T_2 oraz T_2 i T_3 muszą zostać przesłane przed upływem ograniczenia czasowego. Nie zakończenie przesyłania informacji w przewidzianym przedziale czasu spowoduje opóźnienie startu, a co za tym idzie wykonania kolejnych zadań, a to z kolei może stać się przyczyną naruszenia przez aplikację przewidzianego dla niej ograniczenia czasowego. W rozważanym powyżej przykładzie jakiegokolwiek niedopuszczalne opóźnienie występujące w dostarczaniu informacji z zadania T_1 do T_2 , będzie powodem opóźnienia startu i wykonywania zadań T_2 i T_3 oraz stanie się, z wysokim prawdopodobieństwem, przyczyną naruszenia przez zadanie T_3 jego ograniczenia czasowego. Ponieważ w większości aplikacji czasu rzeczywistego konsekwencje naruszenia ograniczeń czasowych mogą być bardzo poważne, jest rzeczą niezwykle istotną zaprojektowanie podsystemu komunikacyjnego (ang. communication subsystem) w taki sposób, aby wszystkie operacje wymiany informacji zachodziły bez naruszenia przypisanych im ograniczeń czasowych.

7. ODPORNOŚĆ NA BŁĘDY W SYSTEMACH CZASU RZECZYWISTEGO

Odporność na błędy (ang. fault tolerance) może zostać zdefiniowana, jako zdolność systemu do poprawnej pracy, w tym do dostarczenia określonych usług, nawet w przypadku wystąpienia błędów. Podstawowa trudność jaka występuje podczas projektowania systemów czasu rzeczywistego, polega na tym, że wymagania czasowe występujące w systemach czasu rzeczywistego są przeciwstawne w stosunku do wymagań niezawodnościowych. Ponadto nie można traktować stawianych przed systemem wymogów niezawodnościowych i wymogów poczynionych odnośnie czasu obliczeń w sposób całkowicie od siebie niezależny, ponieważ w systemach czasu rzeczywistego poprawność uzyskanych wyników zależy nie tylko od ich poprawności logicznej, ale także od tego czy zostały one dostarczone w ramach ściśle określonego limitu czasowego. Zatem system czasu rzeczywistego może przestać funkcjonować poprawnie z powodu wystąpienia błędów w działaniu jego struktury sprzętowej, błędów programu lub też z powodu niedotrzymania zdeterminowanych przez środowisko jego pracy reżimów czasowych. Tak więc system czasu rzeczywistego może być uważany za taki, który musi dostarczyć oczekiwanych usług, w zadanym czasie, nawet w sytuacji wystąpienia błędów. Fakt nie dotrzymania ograniczenia czasowego może być potencjalnie równie niebezpieczny jak awaria systemu lub wygenerowanie przez zadanie błędnych wyników, gdyż w każdym z wymienionych powyżej przypadków cyfrowy system sterowania może utracić stabilność.

Jeżeli logiczna poprawność działania systemu może zależeć od ograniczeń czasowych, nałożonych na pewne jego komponenty, rozdzielenie funkcjonalnej specyfikacji (ang. functional specification) systemu od jego specyfikacji czasowej (ang. timing specification) jest zadaniem bardzo trudnym do realizacji. Ponadto konieczność

dochowania ograniczeń czasowych oraz dążenie do zapewnienia wysokiego stopnia niezawodności systemu mogą posiadać zupełnie przeciwstawne wymagania. Na przykład często wywoływane diagnostyczne programy, sprawdzające stan systemu oraz rozbudowane procedury obsługi błędów (ang. error recovery routines) powodują oczywiście podniesienie stopnia niezawodności systemu, ale jednocześnie zwiększają ryzyko nie otrzymania ograniczeń czasowych przez zadania związane z daną aplikacją.

W przypadku, gdy specyfikacja systemu wymaga wykonania pewnych usług w ściśle określonym czasie, wówczas niezdolność systemu do spełnienia ograniczeń czasowych będzie postrzegana jako wystąpienie błędu bądź awarii (ang. failure) systemu. Jednakże zastosowanie prostego podejścia polegającego na zastosowaniu istniejących już metod, stosowanych przy projektowaniu systemów, odznaczających się dużym stopniem niezawodności oraz traktowanie wystąpienia naruszeń ograniczeń czasowych jako błędów, nie zawsze spełnia potrzeby systemów czasu rzeczywistego. Podstawowa różnica polega na tym, że systemy czasu rzeczywistego muszą zachowywać się w sposób przewidywalny, nawet w sytuacji wystąpienia błędów. Tak więc odporność systemu na pojawiające się błędy oraz nałożone na pracę systemu wymagania czasowe muszą być rozpatrywane łącznie i jednocześnie w fazie projektowania systemów. Wyzwanie, jakie jest stawiane przed projektantem systemu czasu rzeczywistego polega na włączeniu do specyfikacji systemu wymagań czasowych i niezawodnościowych, na każdym poziomie abstrakcji, oraz zaadoptowaniu metodologii projektowania, która pozwalałaby na zagwarantowanie przewidywalnego zachowania się systemu, nawet w przypadku wykrycia błędów, utraty części informacji pochodzącej ze środowiska systemu, rekonfiguracji oraz jego powrotu do stanu prawidłowego funkcjonowania (ang. recovery) po zaistnieniu sytuacji awaryjnej. Formalna specyfikacja wymagań niezawodnościowych oraz ich wpływ na dotrzymanie ograniczeń czasowych, stanowi dziedzinę wymagającą jeszcze badań. W szczególności określenie ograniczeń czasowych, w ramach których system może poprawnie funkcjonować na podstawie znajomości jego zasobów, jest bardzo trudnym zadaniem.

7.1. REDUNDACJA ZASOBÓW SYSTEMU I DODATKOWE OBLICZENIA KONTROLNE

Metodologie projektowania systemów o wysokim stopniu niezawodności były często charakteryzowane przez stopień redundancji czasowej i redundancji zasobów dostępnych dla systemu (ang. time and space redundancy). W przypadku systemów, nie będących systemami czasu rzeczywistego, czas obliczeń traktowany jest, jako posiadające stosunkowo niewielką wartość zasoby, stąd też większość wykorzystywanych metod koncentrowała się głównie wokół optymalizacji rozmiaru zasobów wykorzystywanych przez system. W przypadku środowiska czasu rzeczywistego, główna tendencja polega na zwiększaniu dostępnych dla systemu zasobów, ponieważ dotrzymanie ograniczeń czasowych posiada zasadnicze znaczenie dla zapewnienia poprawnego działania systemu. Chociaż czynniki dotyczące czasu obliczeń i rozmiaru zasobów stanowią bazę dla większości metodologii projektowania systemów o wysokim stopniu niezawodności,

nie jest rzeczą do końca jasną, czy powyższy paradygmat jest właściwy dla czynionych prób określenia stopnia niezawodności systemów czasu rzeczywistego. W szczególności redundancja musi być brana pod uwagę w kontekście zarówno przewidywalnego zachowania się (ang. predictability), jak i niezawodności systemu (ang. dependability). Przykładowo, tolerowanie przejściowych błędów (ang. transient faults) poprzez podejmowanie prób wznowienia obliczeń, jest postępowaniem całkowicie dopuszczalnym, jednakże tylko pod warunkiem dochowania ograniczeń czasowych. Takie samo założenie dotyczy postępowania podczas powracania systemu do ponownego działania po wystąpieniu błędu, gdzie wykorzystywanych jest kolejno kilka różnych wersji tych samych modułów programowych, z nadzieją, że któryś z nich wywoła pożądany efekt. Jednakże w przypadku, gdy czas obliczeń jest bardzo cennym zasobem, podczas projektowania systemów musi zostać zastosowane zupełnie odmienne podejście. W szczególności dokładność i jakość produkowanych przez system wyników obliczeń, może stać się kolejnym czynnikiem, branym pod uwagę w procesie projektowania. Podejście takie stanowi przesłankę dla konstrukcji systemów czasu rzeczywistego, charakteryzujących się łagodną degradacją stopnia sprawności działania, ponieważ w systemach takich wystąpienie błędu powoduje celowe zmniejszenie jakości świadczonych przez system usług, w celu zapewnienia dotrzymania ograniczeń czasowych. Chociaż, ogólne metody, które polegają na redukcji liczby mniej ważnych zadań, poprzez ich skasowanie, należą do omawianej powyżej kategorii, jednakże jakość działania systemu (ang. quality) musi zostać, w tym przypadku, zdefiniowana w kategoriach szczegółowej semantyki związanej z konkretną aplikacją. Na przykład, systemy sterowania w czasie rzeczywistym (ang. real-time control systems) charakteryzowane są przez zmienne o zmieniających się w sposób ciągły wartościach. Zatem wartości te mogą zostać przybliżone lub mogą być estymowane, w przypadku gdy czynnik czasowy nie pozwala na wykonanie precyzyjnych obliczeń. Zaproponowane w pracy [21] podejście, opierające się na obliczeniach nieprecyzyjnych (ang. imprecise computations), stanowi jedną z metod postępowania, w których dokładność obliczeń została celowo zmniejszona, tak aby mogły być dochowane ograniczenia czasowe, gdyż w rozważanym przypadku dokładność obliczeń rosła monotonicznie z czasem ich wykonywania. Ponadto zwiększanie redundancji zasobów wykorzystywanych przez system również posiada, w przypadku systemów czasu rzeczywistego, swoje potencjalne wartości graniczne, ponieważ wprowadza dodatkowe narzuty czasowe, związane z zarządzaniem bardziej rozbudowanym systemem, co zostało pokazane w pracy [22].

7.2. ZARZĄDZANIE STOPNIEM REDUNDANCJI SYSTEMU

Chociaż postęp poczyniony w dziedzinie konstrukcji systemów rozproszonych i równoległych stwarza możliwość zapewnienia dotrzymania ograniczeń czasowych, przy jednoczesnym zapewnieniu wysokiego stopnia niezawodności pracy tych systemów, jednakże wykorzystanie, związanej w naturalny sposób z charakterem tych systemów, wewnętrznej redundancji nie jest pozbawione dodatkowych kosztów. Dla

przykładu narzuty związane z synchronizacją oraz brak determinizmu, wywołany opóźnieniami komunikacyjnymi, stanowią przyczynek do rosnącej złożoności systemów, które mają zachowywać się w sposób przewidywalny czasowo. Tak więc celowe zarządzanie stopniem redundancji systemu, pozostaje wciąż otwartym problemem badawczym. Przykładowo zbiór identycznych programów usługowych (ang. servers) uruchomionych jednocześnie na wielu procesorach zapewnia wysoki stopień niezawodności systemu w przypadku wystąpienia sytuacji awaryjnych. Jednakże, jak zostało to przedstawione w pracy [22], prawidłowe zarządzanie systemem o tak dużej redundancji wymaga rozwiązania problemu wzajemnej synchronizacji programów usługowych, określenia kolejności akceptacji żądań wykonania usług oraz uwzględnienia kosztów detekcji pojawienia się błędu i ponownego powrotu systemu do stanu poprawnego funkcjonowania. Jak widać w tym przypadku zwiększenie stopnia redundancji systemu powoduje wzrost narzutów czasowych. Narzuty związane z zarządzaniem systemem o dodatkowej redundancji muszą być brane pod uwagę, w sposób niezwykle precyzyjny, tak aby mogły być udzielone pewne gwarancje, odnośnie dotrzymania ograniczeń czasowych w systemie.

Zarządzanie stopniem redundancji systemu w sposób przewidywalny związane jest ściśle z wymogami stawianymi przez teorię szeregowania zadań dla systemów czasu rzeczywistego. Konieczność spełnienia nałożonych na pracę systemu ograniczeń czasowych, wymusza takie szeregowanie kolejności wykorzystywania zasobów systemu, aby zachowanie się systemu w czasie było zrozumiałe, przewidywalne i możliwe do opanowania. Większość z istniejących algorytmów szeregowania dopuszcza wykorzystanie w danej chwili czasu tylko jednego z zasobów oraz pomija w ogóle zagadnienia związane z niezawodnością działania systemu. Wymóg dotrzymania ograniczeń czasowych, nawet w obecności wystąpienia błędów, narzuca dodatkowe wymagania na zastosowane algorytmy szeregowania zadań. W celu zapewnienia przewidywalnego zachowania się i wysokiego stopnia niezawodności złożonych systemów czasu rzeczywistego, muszą zostać zastosowane specjalne techniki alokacji zasobów. Powyższym zagadnieniom została poświęcona praca [23].

7.3. NADZOROWANIE PRZEBIEGU ZADAŃ W TRAKCIE ICH WYKONYWANIA

W fazie projektowania systemów czasu rzeczywistego niejednokrotnie czynione są pewne założenia odnośnie zachowania się systemu i jego środowiska. Powyższe założenia mogą przybrać wiele form, przykładowo mogą mieć postać wartości granicznych, nałożonych na opóźnienia występujące w komunikacji pomiędzy poszczególnymi procesorami, maksymalnych czasów wykonania zadań, limitów czasowych, w ramach których zadania muszą zostać zakończone lub też minimalnych wartości przedziałów czasowych pomiędzy wystąpieniem dwóch różnych zdarzeń. Tak więc projektant systemu musi uporać się z brakiem deterministycznego zachowania się zewnętrznego środowiska, co często odbywa się kosztem uproszczenia problemu, który jest bardzo trudny lub wręcz nie możliwy do rozwiązania w inny sposób. Przypuszczenia czynione odnośnie środo-

wiska systemu, mogą zostać wyrażone jako część formalnej specyfikacji systemu, albo też w postaci dodatkowych wymagań nałożonych na proces szeregowania zadań. Pomimo opracowania formalnych metod weryfikacji poprawności działania systemów oraz postępów w teorii szeregowania zadań dla systemów czasu rzeczywistego, często zachodzi konieczność nadzorowania przebiegu zadań w trakcie ich wykonywania. Spowodowane jest to tym, że nie istnieją absolutnie niezawodne systemy komputerowe czasu rzeczywistego, a interakcja systemu komputerowego z jego środowiskiem powoduje wprowadzenie dodatkowego czynnika zwiększającego stopień braku determinizmu. Ponadto założenia poczynione na etapie projektowania systemu mogą zostać naruszone w trakcie jego pracy, na skutek zaistnienia nieprzewidzianych okoliczności, takich jak przejściowe przeciążenia systemu (ang. transient overloads). Zastosowanie formalnych technik analizy systemów lub wybranych algorytmów szeregowania zadań, powoduje konieczność poczynienia dodatkowych założeń odnośnie systemu i jego interakcji ze środowiskiem, co nie zawsze jest wykonalne na etapie projektowania, skąd wynika konieczność przeprowadzania kontroli wykonywania zadań na bieżąco podczas pracy systemu. Głębszej analizie powyższych zagadnień zostały poświęcone prace [5] i [24].

Nadzorowanie systemu w trakcie jego pracy powoduje konieczność wprowadzenia znaczników czasu (ang. timestamping), gromadzenia informacji o wystąpieniu zdarzeń, analizowania historii podczas rejestracji nowych zdarzeń oraz przesyłu tych informacji do pozostałych elementów systemu. W pracach [25] i [26] opisano wykorzystanie specjalnych urządzeń sprzętowych, w celu zapewnienia nadzorowania procesu wykonywania zadań, bez bezpośredniej ingerencji w pracę systemu.

Oprócz detekcji naruszenia założeń poczynionych na etapie projektowania systemu, nadzorowanie zadań w trakcie ich wykonywania, może zostać wykorzystane w celu wykrycia sytuacji wyjątkowych, jakie występują podczas pracy danej aplikacji czasu rzeczywistego. Przykładowo w pracy [5] opisano sposób funkcjonowania systemu nadzorczego, którego zadaniem była detekcja błędów, będących wynikiem niewłaściwej specyfikacji systemu. Ponadto, oprócz wykrywania sytuacji wyjątkowych, nadzorowanie przebiegu procesu wykonywania zadań może dostarczyć sprzężenia zwrotnego do innych części systemu. Jak pokazano w pracy [25] informacja zebrana przez urządzenia nadzorcze może zostać wykorzystana w celu zapewnienia sprzężenia zwrotnego z operatorem systemu, z zadaniami aplikacyjnymi lub programem szeregującym zadania. Dla przykładu, fakt przekroczenia maksymalnego limitu czasu, jaki został przewidziany na wykonanie danego zadania, może zostać odnotowany przez urządzenia nadzorcze. Sprzężenie zwrotne z programem szeregującym zadania może, z kolei, posłużyć do opracowania systemu, zdolnego do automatycznej adaptacji do zmian środowiska pracy oraz zmieniającego się stopnia jego obciążenia. Tworzenie zadań aplikacyjnych, których celem jest nadzorowanie pracy systemu oraz opracowywanie w tym celu specjalnych systemów operacyjnych, zapewniających sprzężenie zwrotne do pozostałych komponentów systemu, jest obszarem intensywnych prac badawczych. Z powyższą problematyką związane jest ściśle zagadnienie jednoczesnego szeregowania w systemie zadań nadzorujących inne zadania w trakcie ich wykonywania (ang. run-time monitoring tasks) oraz zadań stanowiących dane aplikacje czasu rzeczywistego.

W
globaln
Czas
projek
korzys
munic
niu his
działa
Wprov
cych p
wych
każde
C
zegarc
kiem,
niepra
system
chow
zegarc
przyk
z trze
Z
średn
algor

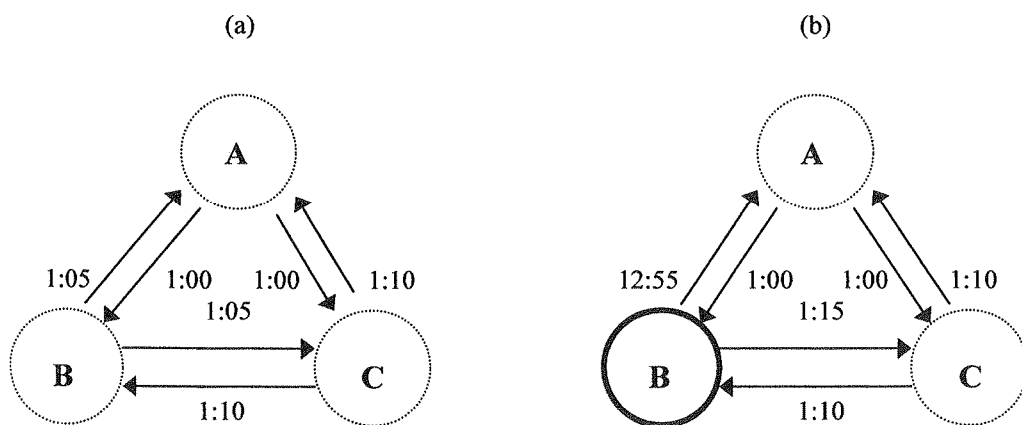
Rys.

8. SYNCHRONIZACJA W SYSTEMACH CZASU RZECZYWISTEGO

W rozproszonych systemach komputerowych konieczność zapewnienia bazy czasu globalnego (ang. global time base) jest wymogiem posiadającym zasadnicze znaczenie. Czas globalny może przyczynić się w stopniu znaczącym do uproszczenia procesu projektowania algorytmów, odznaczających się wysokim stopniem niezawodności, wykorzystywanych dla potrzeb komunikacji pomiędzy procesami (ang. interprocess communication), sprawdzaniu stanu wykonania programów (ang. checkpointing), odtwarzaniu historii wykonania programów, w przypadku powracania systemu do prawidłowego działania po detekcji błędu (ang. rollback recovery) oraz alokacji zasobów systemu. Wprowadzenie czasu globalnego, może także uprościć użycie znaczników sygnalizujących przekroczenie zadanego przedziału czasu (ang. timeouts) lub ograniczeń czasowych (ang. deadlines), co jest niezwykle istotne dla prawidłowego funkcjonowania każdego rozproszonego systemu czasu rzeczywistego.

Czas globalny może zostać wprowadzony poprzez zsynchronizowanie wszystkich zegarów występujących w systemie. Nie stanowi to większego problemu, pod warunkiem, że wszystkie zegary występujące w systemie, nawet te, które działają w sposób nieprawidłowy, zachowują się w sposób niezmienny podczas całego czasu pracy systemu. Jednakże, w przypadku, gdy niektóre z wadliwie działających zegarów zachowują się w sposób całkowicie dowolny, przeprowadzenie synchronizacji wszystkich zegarów może stwarzać poważne problemy. Powyższy problem zostanie zilustrowany na przykładzie przedstawionym na rys. 1. Został tam przedstawiony system składający się z trzech węzłów, z których każdy posiada własny zegar synchronizujący jego pracę.

Zegary są zsynchronizowane poprzez dopasowanie każdego z nich do wartości średniej czasów podawanych przez każdy z zegarów. Ten prosty i intuicyjnie poprawny algorytm daje oczekiwane rezultaty pod warunkiem zachowywania się w sposób stały



Rys. 1. Przykłady wystąpienia błędu podczas synchronizacji zegarów. (a) Wszystkie zegary działają w sposób prawidłowy. (b) W węźle B wystąpił zegar działający wadliwie

każdego z zegarów, jak zostało to przedstawione na rys. 1 (a). Jednakże w przypadku nieprawidłowej pracy jednego z zegarów, gdzie do dwóch pozostałych zegarów podawane są różne, przypadkowe wartości czasów, przeprowadzenie synchronizacji pozostałych zegarów jest niemożliwe. Dla przykładu na rys. 1 (b) zegar w węźle B podaje zafałszowane informacje do zegarów w węzłach A i C, w wyniku czego, żaden z tych zegarów nie wprowadza do podawanego przez siebie czasu żadnej korekty, ponieważ każdy z nich uważa, że podaje właśnie czas średni.

Praca [27] była pierwszym opracowaniem poświęconym problemowi synchronizacji trzech zegarów, w przypadku zachowania się jednego z nich w sposób całkowicie arbitralny. W pracy tej zostało wykazane, że w przypadku podawania przez jeden z zegarów zupełnie dowolnych wartości czasów, nie istnieje żaden algorytm, który byłby w stanie zagwarantować poprawną synchronizację pozostałych zegarów, w przypadku systemu o trzech węzłach. W pracy tej zostało wykazane ponadto, że liczba $3m + 1$ zegarów, jest wystarczająca do przeprowadzenia synchronizacji systemu, w przypadku wystąpienia m zegarów wadliwych. W pracy [28] zostało pokazane, że powyższy warunek jest warunkiem koniecznym i wystarczającym do przeprowadzenia synchronizacji pozostałych zegarów systemu.

Od czasu aplikacji pracy [27], problem synchronizacji zegarów w systemie z pewną liczbą wadliwych zegarów był intensywnie badany, przy czym uzyskane wyniki można znaleźć w pracy [29]. Proponowane w literaturze rozwiązania mogą zostać rozpatrywane w kategoriach sprzętowych lub programowych. Podejście programowe jest bardziej elastyczne i tanie, ale jak zostało pokazane w pracach [30], [31], [32] i [33] wymaga przesyłu dodatkowych informacji, wykorzystywanych tylko i wyłącznie na potrzeby synchronizacji. Podstawowa zasada synchronizacji dokonywanej w sposób programowy polega na tym, że każdy z węzłów posiada zegar logiczny (ang. logical clock), który dostarcza podstawy czasu (ang. time base) dla wszelkich operacji przeprowadzanych w tym węźle. Taki zegar logiczny konstruowany jest w oparciu o zegar sprzętowy (ang. hardware clock), ale posiada znacznie większą ziarnistość czasową (ang. granularity) niż zegar sprzętowy. Poszczególne węzły wymieniają, w sposób okresowy, wartości swoich czasów i dokonują dostrojenia swoich zegarów, w oparciu o wartości dostarczane przez pozostałe zegary. Algorytmy synchronizacji programowej różnią się sposobem, w jaki węzły dokonują wymiany aktualnych wartości swoich zegarów i wyborem rodzaju procedury dostrojenia zegarów.

W przypadku algorytmów zbieżnych do wartości średniej (ang. convergence-averaging algorithms) węzły systemu dokonują wzajemnej synchronizacji w periodycznie występujących momentach czasu. W przypadku gdy zegar danego węzła osiągnął wartość odpowiadającą momentowi synchronizacji, wysyła wartość swojego zegara do pozostałych węzłów oraz czeka, przez czas równy okresowi procesu synchronizacji, na otrzymanie aktualnych wartości zegarów z pozostałych węzłów systemu. Na końcu okresu oczekiwania następuje dostrojenie zegara węzła, zgodnie z wartościami otrzymanymi z pozostałych węzłów, po uwzględnieniu momentów czasowych przybycia tych sygnałów. Dla przykładu w pracy [31] do dostrojenia zegara zaproponowano wykorzystanie średniej arytmetycznej wszystkich sygnałów zegarowych. Ponadto w celu elimi-

nacji w
działają
nywane
kracząc
elimin
i m naj
medial
cych z

Z
cover-
jest w
system
w peri
nięcia
próbę
stałych
otrzym
popraw
uznaje
wiadom
aktual

W
algori
uzgod
one p
niowe

C
wej p
synch
wystę
algor
przes
ne w
i min
warto
syste

przy
Zapr
dym
sync
fazo
wan

nacji wpływu na wartość powyższej średniej sygnałów pochodzących od zegarów działających w sposób wadliwy, otrzymane ze wszystkich węzłów wartości są porównywane z wyliczoną uprzednio pewną wartością progową i te wartości, które przekraczają zadany próg są następnie odrzucane. Z kolei w pracy [32] każdy z węzłów eliminuje wpływ wadliwie działających zegarów, poprzez odrzucenie m największych i m najmniejszych otrzymanych wartości, a jako wartość prawidłowa przyjmowana jest medialna pozostałych wartości, przy czym m jest maksymalną liczbą wadliwie działających zegarów, która może być jeszcze tolerowana.

Z kolei w pracach [30] i [34] została zaproponowana klasa algorytmów (ang. cover-gance non-aeraging algorithms), w których celu synchronizacji zegarów nie jest wykorzystywana wartość średnia czasów podawanych przez wszystkie zegary systemu. W tym przypadku poszczególne węzły również dokonują synchronizacji w okresowo następujących po sobie momentach czasowych. W przypadku osiągnięcia przez zegar danego węzła jednego z takich momentów, węzeł ten podejmuje próbę przeprowadzenia synchronizacji całego systemu, poprzez przesłanie do pozostałych węzłów systemu prośby o przeprowadzenie synchronizacji ich zegarów. Po otrzymaniu takiej wiadomości pozostałe węzły systemu dokonują sprawdzenia jej poprawności i dokonują dostrojenia swoich zegarów do otrzymanej wartości, jeżeli tylko uznają, że jest poprawna. Sposób sprawdzania poprawności i przydatności otrzymanej wiadomości dla potrzeb przeprowadzenia synchronizacji jest zależny od zastosowanego aktualnie algorytmu.

W pracy [31] zaproponowano wykorzystanie algorytmów (ang. consistency-based algorithms), które traktowały przesyłane wartości zegarów jako dane i dokonywały ich uzgodnienia. Podstawowa zaleta tych algorytmów polegała na tym, że nie wymagały one przeprowadzania synchronizacji poszczególnych zegarów, ale ich narzuty obliczeniowe były zdecydowanie większe niż w przypadku pozostałych algorytmów.

Główny problem występujący w przypadku algorytmów synchronizacji programowej polega na tym, że bazują one na wiadomościach przesyłanych w celu dokonania synchronizacji poszczególnych zegarów. W rezultacie maksymalne wartości błędów występujących podczas procesu synchronizacji, gwarantowane przez większość z tych algorytmów, są większe niż różnica pomiędzy maksymalnym i minimalnym czasem przesyłu informacji pomiędzy dwoma dowolnymi węzłami systemu, co zostało pokazane w pracy [35]. Ponieważ, w ogólnym przypadku, różnica pomiędzy maksymalnym i minimalnym czasem przesyłu informacji może być bardzo duża, odpowiadające jej wartości błędów synchronizacji są również duże, a to oznacza, że synchronizacja systemu nie jest ścisła.

Powyższy problem nie występuje w przypadku przeprowadzania synchronizacji przy użyciu, zaproponowanych w pracach [36], [37], [38] i [39] rozwiązań sprzętowych. Zaproponowane w powyższych pracach algorytmy, wykorzystują zainstalowane w każdym z węzłów systemu specjalne urządzenia sprzętowe, w celu uzyskania bardzo ścisłej synchronizacji. Konstrukcja takich urządzeń bazuje na koncepcji pętli synchronizacji fazowej (ang. phaselocked loop). Zegar sprzętowy, w każdym z węzłów, jest zrealizowany w postaci wyjścia generatora sterowanego napięciem (ang. voltage-controlled

oscillator). Napięcie przyłożone do generatora pochodzi z wyjścia detektora fazy (ang. phase detector), którego sygnał jest proporcjonalny do wielkości błędu fazy (ang. phase error) pomiędzy sygnałem zegara, który jest synchronizowany i sygnałem odniesienia, generowanym przez pozostałe zegary sytemu. Na skutek systematycznego dostrajania faz wszystkich zegarów, zegary te zostają zsynchronizowane. Podstawowy problem występujący w algorytmach synchronizacji sprzętowej polega na doborze odpowiedniej strategii selekcji sygnału odniesienia. Jest to zagadnienie ważne, z uwagi na możliwość występowania w systemie zegarów, działających w sposób nieprawidłowy. Niestety intuicyjne rozwiązanie polegające na wyborze mediany policzonej dla wszystkich sygnałów, w przypadku istnienia uszkodzonych zegarów, które zachowują się w sposób całkowicie arbitralny, nie daje oczekiwanych rezultatów.

Jedno z pierwszych rozwiązań sprzętowych, działających poprawnie w obecności wadliwie funkcjonujących zegarów zostało zaproponowane w pracy [38]. Jednakże podstawowy mankament zaproponowanego tam rozwiązania, polega na tym, że dzięki niemu możliwa staje się synchronizacja maksymalnie tylko czterech zegarów. Zaproponowany w pracy [38] algorytm został następnie rozszerzony w pracach [36], [37] i [39], tak że możliwa stała się synchronizacja dowolnej liczby zegarów. Jednakże zaproponowane tam algorytmy posiadały dwa główne ograniczenia. Pierwsze z nich polega na tym, że do funkcjonowania tych algorytmów potrzebna jest pełna sieć połączeń (ang. fully connected network) pomiędzy wszystkimi zegarami systemu, tak aby każdy z zegarów wysyłał swój sygnał do wszystkich pozostałych zegarów występujących w systemie. W przypadku małych systemów nie stanowi to większego problemu. Jednakże w przypadku zwiększania się rozmiarów systemu, całkowita liczba połączeń w sieci połączonej w sposób pełny, byłaby tak ogromna, że niezawodność procesu synchronizacji byłaby zdeterminowana raczej przez częstotliwość awarii tych połączeń, niż przez częstotliwość wystąpień uszkodzeń zegarów. Ponadto powstałyby problemy technicznej realizacji układów sprzętowych o tak dużej liczbie wejść i wyjść. W pracy [37] zaproponowano łączenie węzłów w grupy (ang. clusters), gdzie całkowita liczba połączeń pomiędzy poszczególnymi podjednostkami systemu byłaby z zasady mniejsza niż w przypadku realizacji sieci o pełnych połączeniach. Drugie z ograniczeń rozważanych algorytmów polega na tym, że przyjmują one zaniedbywanie mały czas propagacji sygnałów zegarowych pomiędzy dwoma dowolnymi węzłami. To znowuż może stanowić zasadniczy problem w przypadku bardzo dużych systemów, gdzie fizyczna odległość pomiędzy dwoma zegarami może być dostatecznie duża, aby mogła w rezultacie wywołać opóźnienia propagacji sygnałów, które nie mogą już zostać zaniedbane. Propozycję eliminacji powyższego ograniczenia zaproponowano w pracach [40] i [41], gdzie zaprezentowano algorytmy synchronizacji sprzętowej, zapewniające bardzo dokładną synchronizację, nawet w dużych systemach rozproszonych, przy jednocześnie bardzo małych narzutach czasowych. Jednakże znaczny koszt dodatkowych urządzeń sprzętowych, instalowanych w każdym z węzłów, stanowi skuteczną barierę utrudniającą stosowanie takich rozwiązań w dużych systemach rozproszonych, chyba że zapewnienie ścisłej synchronizacji poszczególnych węzłów systemu jest sprawą absolutnie zasadniczą.

W
propon
stopnie
z węz
gramo
wane
wystap
służą c
wartoś
przypa
pomie
dowol
In
kompr
rozwi
chroni
dopod
na zeg
Na ko
powst
wprov
niż w
ponow
nej l
przes
zaws
odczy
powy
wie
powi
zegar
wzro
bilist
w pr

moż
syste
życi
oraz

W celu redukcji znacznych kosztów rozwiązań sprzętowych, w pracy [42] zaproponowano podejście hybrydowe, gdzie starano się uzyskać kompromis pomiędzy stopniem dokładności procesu synchronizacji, a wymaganiami sprzętowymi w każdym z węzłów. Zastosowane w pracy [42] rozwiązanie polegało na wzbogaceniu programowych algorytmów synchronizacji pewnymi rozwiązaniami sprzętowymi. Zastosowane urządzenia sprzętowe mają za zadanie oszacowanie wartości opóźnień, które wystąpiły w procesie przesyłania informacji. Następnie uzyskane w ten sposób wartości służą do przeprowadzenia korekty, dokonanej na drodze czysto programowej oszacowań wartości opóźnień. W powyższym rozwiązaniu hybrydowym, uzyskany dla najgorszego przypadku stopień synchronizacji systemu, był znacznie lepszy niż wynikający z różnicy pomiędzy maksymalnym i minimalnym czasem przesyłu wiadomości pomiędzy dwoma dowolnymi węzłami.

Inne, zaproponowane w pracach [43] i [44] podejście, mające za zadanie uzyskanie kompromisu pomiędzy dokładnością procesu synchronizacji, a kosztem zastosowanych rozwiązań, nazwane zostało synchronizacją probabilistyczną (ang. probabilistic synchronisation). Powyższe podejście opiera się na znajomości funkcji rozkładu prawdopodobieństwa czasów opóźnień, jakie występują podczas przesyłania wiadomości oraz na zezwoleniu każdemu z węzłów systemu na kilka prób odczytu pozostałych zegarów. Na końcu każdej takiej próby, węzeł oblicza maksymalną wartość błędu, jaka mogła powstać, jeżeli wartość zegara odczytana podczas tej próby została wykorzystana do wprowadzenia poprawki. Jeżeli przybliżona wartość maksymalnego błędu jest większa niż wyspecyfikowana uprzednio wartość progowa, wówczas dany węzeł podejmuje ponowną próbę odczytania zegara innego węzła. W przypadku ograniczenia maksymalnej liczby prób, jakie dany węzeł może poczynić (w celu ograniczenia wielkości przesyłanych wiadomości i związanych z tym narzutów komunikacyjnych) istnieje zawsze niezerowe prawdopodobieństwo zdarzenia polegającego na tym, że dany węzeł odczyta aktualne wskazanie zegara innego węzła z założoną uprzednio precyzją. Wadą powyższego rozwiązania jest możliwość utraty synchronizacji systemu. W przeciwieństwie do zaprezentowanych uprzednio rozwiązań, w podejściu probabilistycznym, odpowiadające najgorszemu z możliwych przypadków opóźnienia przesyłanych wartości zegarów, mogą zostać przyjęte dowolnie małe. Jednakże ich zmniejszanie powoduje wzrost prawdopodobieństwa utraty synchronizacji w systemie. Ponadto podejście probabilistyczne charakteryzuje się mniejszą odpornością na powstałe ewentualne błędy, w przeciwieństwie do omawianych wcześniej rozwiązań.

9. OBSZARY ZASTOSOWAŃ SYSTEMÓW CZASU RZECZYWISTEGO

Istnieje wiele różnorodnych aplikacji systemów czasu rzeczywistego, do których można zaliczyć większość z militarnych systemów obrony, zautomatyzowanych fabryk, systemów nadzoru procesów przemysłowych, systemów ratujących i podtrzymujących życie ludzkie (ang. life-support systems), ponadto systemy ochrony i nadzoru obiektów oraz wiele innych. Powyższe aplikacje często stawiają przed systemem komputerowym

wymagania, polegające na konieczności pracy w pętli sprzężenia zwrotnego, zdolności do inteligentnego podejmowania decyzji oraz realizację w czasie rzeczywistym dostępu do baz danych. Wymienione zagadnienia zostaną przedyskutowane szerzej w kolejnych podpunktach.

9.1. SYSTEMY STEROWANIA W CZASIE RZECZYWISTYM

Komputery cyfrowe są powszechnie używane w systemach sterowania w czasie rzeczywistym, głównie ze względu na ich zwiększoną w ostatnich latach wydajność obliczeniową oraz niezawodność podczas sterowania przebiegiem coraz to bardziej złożonych procesów. Komputer cyfrowy umieszczony w pętli sprzężenia zwrotnego systemu sterowania, wylicza wejściową wartość sygnału, poprzez wykonywanie sekwencji instrukcji, wprowadzając tym samym do sterowanego procesu pewne nieuniknione opóźnienie, zwane opóźnieniem związanym z czasem obliczeń (ang. computation-time delay). Jest to opóźnienie dodatkowe, które sumuje się z opóźnieniem opisywanym w literaturze związanej z dziedziną teorii sterowania. Opóźnienie związane z czasem obliczeń, stanowi pokąsną część opóźnienia, występującego w pętli sprzężenia zwrotnego, do którego zalicza się także opóźnienia wnoszone przez urządzenia pomiarowe i sensoryczne przetworniki A/D i D/A oraz układy wykonawcze. Jednakże wymienione tutaj pozostałe składniki opóźnienia mają zwykle stałe wartości, co znacznie ułatwia zapewnienie poprawnego działania systemu w ich obecności.

Ze względu na zależne od danych pętle programowe, występowanie instrukcji skoków warunkowych oraz nieprzewidywalne opóźnienia jakie występują na skutek dzielenia zasobów podczas wykonywania zaimplementowanych algorytmów sterowania, opóźnienie związane z czasem obliczeń jest zmienną losową, o wartości znacznie mniejszej niż jeden okres próbkowania T_s , jeżeli tylko podczas pracy komputera sterującego danym procesem nie wystąpi żaden błąd. W przypadku wystąpienia jakiegokolwiek awarii, którejś z części systemu lub pochodzącego ze środowiska zakłócenia, np. w postaci elektromagnetycznej, czas jaki musi zostać poświęcony na detekcję i lokalizację tego urządzenia oraz podjęcie odpowiedniej akcji ratunkowej, mającej za zadanie zapewnienie powrotu systemu do stanu normalnego funkcjonowania, musi zostać dodany do całkowitego czasu wykonania programu sterującego danym procesem, co przyczynia się do zwiększenia, w sposób znaczący, opóźnienia związanego z czasem obliczeń. Powyższe zjawisko przyczynia się do poważnej degradacji wydajności obliczeniowej komputerowego systemu sterowania i może w efekcie doprowadzić do powstania katastrofy, jeżeli powstałe opóźnienie przekroczy pewien limit czasowy zwany ostrym ograniczeniem czasowym (ang. hard deadline). Zagadnienie powyższe zostało szerzej omówione w pracy [45].

Wpływ opóźnienia związanego z czasem obliczeń na wydajność obliczeniową systemu oraz stabilność kontrolowanego procesu, analizowany był już przez wielu badaczy. Dla przykładu w pracy [46] zostały podane konieczne warunki stabilności dla systemu liniowego z wnoszonym przez komputer opóźnieniem w pętli sprzężenia

zwrotnego
mierzący
nieliniow
w pracy [4
systemu
modyfika
wymienio
ści pows
badany c
jednakże
wtarzani
ny syste
zwrotnego
nieniem,
stabilnej
jawnego
a wielko
równani
częstotli
ogranicz
pomied

Z k
związan
systemu
czeń cz
wywoła
niepraw
wartośc
ciowych
magnet
maksym
których
jest w g
nego j
wyprov
to ogra
wyprov
wyprov

zwrotnego oraz został przeanalizowany wpływ tego opóźnienia na wartości wskaźników mierzących wydajność obliczeniową systemu. Podobna analiza, przeprowadzona dla nieliniowego systemu sterowania robotem została zaprezentowana w pracy [47]. Z kolei w pracy [48] została przeprowadzona bardziej szczegółowa analiza stabilności cyfrowego systemu sterowania z opóźnieniem w pętli sprzężenia zwrotnego, polegająca na pewnej modyfikacji równania różniczkowego, opisującego stan systemu. Jednakże wszystkie wymienione powyżej analizy przeprowadzane były przy założeniu niezmiennych wartości powstających w systemie opóźnień. Chociaż w pracy [49] problem stabilności był badany dla zmiennych wartości, powstających w pętli sprzężenia zwrotnego opóźnień, jednakże analizę przeprowadzono przy założeniu regularnego i periodycznego powtarzania się pewnych opóźnień wzorcowych. Z kolei w pracy [50] został zamodelowany system sterowania, ze zmiennym opóźnieniem wnoszonym w pętli sprzężenia zwrotnego. Wykorzystano w tym celu równanie różniczkowe ze stochastycznym opóźnieniem, dzięki czemu uzyskano wystarczające warunki stabilności systemu, dla prawie stabilnej częstotliwości próbkowania. Jednakże uzyskany rezultat nie pokazuje żadnego jawnego związku pomiędzy wydajnością obliczeniową systemu i jego stabilnością, a wielkością opóźnienia, ale za to podaje warunki, jakie muszą spełniać współczynniki równania stanu systemu oraz średnia szybkość zmian wielkości opóźnienia dla ustalonej częstotliwości próbkowania. Ponadto w pracy tej przyjęto, że maksymalne opóźnienie ograniczone jest poprzez występujący dla najgorszego przypadku przedział czasu pomiędzy dwoma chwilami próbkowania.

Z kolei w pracy [51] została zaproponowana metoda analizy wpływu opóźnień związanych z obliczeniami na stabilność systemu dla liniowego i niezmiennego w czasie systemu sterowania. W szczególności zostały wyprowadzone wartości ostrych ograniczeń czasowych dla takich systemów sterowania oraz krytyczne wartości opóźnień wywołanych przeprowadzaniem obliczeń, powyżej których system zaczyna działać nieprawidłowo, przy założeniu stacjonarności funkcji rozkładu prawdopodobieństwa wartości czasów opóźnień. Powyższe założenia odpowiadają charakterystykom przejściowych błędów w pracy komputera, wywołanych przykładowo zakłóceniami elektromagnetycznymi. Dynamika systemu była następnie modyfikowana, zgodnie z przyjętą maksymalną wartością opóźnienia NT_s , oraz probabilistyczną funkcją rozkładu opóźnień, których częstotliwości występowania są mniejsze bądź równe NT_s , gdzie N zmieniane jest w granicach od 1 do aktualnie maksymalnego dopuszczalnego opóźnienia, oznaczonego jako DT_s . Zmodyfikowane równanie stanu było następnie testowane w celu wyprowadzenia warunków koniecznych dla asymptotycznej stabilności systemu. Ponadto ograniczenia nałożone na stan systemu i sygnały wejściowe zostały użyte w celu wyprowadzenia dozwolonej przestrzeni stanów, z której silne ograniczenia czasowe są wprowadzane jako funkcja czasu i stanu systemu.

9.2. DZIAŁAJĄCE W CZASIE RZECZYWISTYM SYSTEMY SZTUCZNEJ INTELIGENCJI

Po tym jak metody sztucznej inteligencji (ang. artificial intelligence) stały się dostatecznie dojrzałe, wystąpiło, wciąż wzrastające, zainteresowanie w zastosowaniu

tych metod do sterowania złożonymi rzeczywistymi systemami, w których występują silne ograniczenia czasowe. W systemach takich oczekuje się, że sterujący ich pracą komputer będzie udzielał odpowiedzi na sygnały wejściowe w ramach ściśle określonych limitów czasowych, gdyż w przeciwnym przypadku może nastąpić katastrofalne załamanie się pracy systemu. Ponieważ scenariusz możliwych do wystąpienia sytuacji posiada zbyt wiele wariantów, które mogłyby być kolejno przeanalizowane, a konsekwencje niedotrzymania ograniczeń czasowych są tak poważne, samo testowanie systemu nie jest w stanie dostarczyć gwarancji na to, że jego praca będzie poprawna również w warunkach czasu rzeczywistego. Zagadnieniom tym zostały poświęcone prace [52] i [2], gdzie pokazano, że musi zostać formalnie udowodnione, że dany system jest w stanie dotrzymać narzucone przez środowisko ograniczenia czasowe. Niestety w pracy [53] wykazano, że większość technik i heurystyk stosowanych w systemach sztucznej inteligencji poddaje się z wielką trudnością formalnej analizie, która byłaby w stanie dostarczyć odpowiedzi na pytanie odnośnie gwarantowanych czasów odpowiedzi systemu. Nawet jeżeli zostanie wykazane, że wybrane metody sztucznej inteligencji posiadają przewidywalne czasy odpowiedzi, zmienność tych czasów jest w typowych rozwiązaniach zwykle tak wielka, że podanie gwarancji czasowych, w oparciu o najgorszy z możliwych przypadków, spowodowałaby, że zasoby systemu wykorzystywane byłyby w niezwykle małym stopniu, co ze względu na koszty takich systemów jest zupełnie niedopuszczalne. Powyższym zagadnieniom została poświęcona praca [54].

Powyższe przykłady dowodzą istnienia konfliktu pomiędzy naturą metod stosowanych w sztucznej inteligencji, a wymaganiami stawianymi przed systemami pracującymi w czasie rzeczywistym. Podczas gdy metody sztucznej inteligencji charakteryzują się nieprzewidywalnym oraz podlegającym dużym zmianom czasem wykonania obliczeń, w przypadku systemów sterowania w czasie rzeczywistym czas ten musi być stały i dobrze określony. W większość prac z dziedziny zastosowań metod sztucznej inteligencji w systemach czasu rzeczywistego, jak np. prace [55], [56] i [57], uwaga koncentrowana jest wokół ograniczonych technik sztucznej inteligencji, które charakteryzują się zdeterminowanym czasem obliczeń lub też, czego przykładem mogą być prace [58] i [59], konstruowane są systemy, wykorzystujące tylko w bardzo ograniczonym zakresie możliwości tradycyjnych metod sztucznej inteligencji. Z kolei w pracach [60], [61], [62] i [63] podjęte zostały próby stworzenia systemów łączących w sobie możliwości tradycyjnych technik sztucznej inteligencji z wysokim stopniem determinizmu, charakterystycznego dla aplikacji pracujących w czasie rzeczywistym, jednakże w żadnym z tych systemów nie udało się zapewnić stuprocentowych gwarancji, dotyczących czasu wykonania programów.

W celu połączenia niczym nieograniczonych technik sztucznej inteligencji w pracy [64] zaproponowano wykorzystanie architektury komputerowej, oznaczonej skrótem CIRCA (ang. Co-operative Intelligent Real-time Architecture). W rozważanej architekturze podsystem sztucznej inteligencji w sposób samoczynny wyciąga wnioski odnośnie poziomów ważności zadań, które wymagają zastosowania jego skutecznych, acz nieprzewidywalnych metod wnioskowania. W tym samym czasie oddzielny podsystem czasu rzeczywistego wykorzystuje przewidywalne charakterystyki wydajności oblicze-

niowej c
ści, któ
systemu
współpr
postawi
o modu
niczoną
kację z
udzielo

W
sterowa
którymi
otrzyma
Komple
wszystk
oznacza
czasowy
ce) do
efektyw
powinie
wych z
czasowy

Ko
poczyn
czonej
w któr
czynien
temu. P
mogło
około 2
sterują

Ist
temami
czasow
w stop
gramow
rzeczyw
niu duż
mi. Po

niowej dla poszczególnych zadań, w celu przypisania tym zadaniom poziomów ważności, które są wymagane w celu zagwarantowania określonych czasów odpowiedzi systemu. Kluczowym problemem w opisanym powyżej systemie jest zapewnienie takiej współpracy obu systemów, aby każdy z nich mógł skutecznie dążyć do realizacji postawionych przed nim celów. Architektura CIRCA została zrealizowana w oparciu o moduł szeregujący (ang. scheduling module) oraz interfejs, który pozwala na nieograniczoną niczym pracę podsystemu sztucznej inteligencji i jego asynchroniczną komunikację z podsystemem czasu rzeczywistego, bez naruszenia jakichkolwiek gwarancji udzielonych odnośnie czasów odpowiedzi systemu.

W pracy [65] wykazano, że jakość odpowiedzi udzielonej przez inteligentny system sterowania zależy w zdecydowanym stopniu od czterech podstawowych czynników, którymi są: kompletność odpowiedzi, jej precyzja, stopień zaufania co do poprawności otrzymanych wyników oraz dostarczenie odpowiedzi w oczekiwanym przedziale czasu. Kompletność (ang. completeness) odpowiedzi oznacza, że odpowiedzi są udzielane dla wszystkich możliwych stanów wejść systemu. Czasowość (ang. timeliness) odpowiedzi oznacza, że odpowiedzi są udzielane przed upływem związanych z nimi ograniczeń czasowych. Precyzja (ang. precision) odpowiedzi oraz stopień zaufania (ang. confidence) do niej determinują stopień dokładności odpowiedzi, z czym wiąże się stopień efektywności wykorzystania wiedzy systemu. Idealny inteligentny system sterowania powinien zagwarantować, że dla dowolnej z możliwych sekwencji sygnałów wejściowych zostaną udzielone optymalne odpowiedzi, przy zachowaniu wszelkich wymogów czasowych.

Konstruktorzy niektórych systemów starają się dążyć do powyższego ideału poprzez poczynienie założeń o nieograniczoności zasobów systemu, a zwłaszcza jego nieograniczonej mocy obliczeniowej. Przykładowo w pracy [59] rozważono architekturę systemu, w której każdemu z urządzeń wykonawczych przypisano oddzielny procesor. Poczynienie takich założeń nakłada znaczne ograniczenia na stopień skalowalności systemu. Przykładowo w pracach [66] i [52] wykazano, że aby powyższe rozwiązanie mogło zostać wykorzystane do sterowania platformą wiertniczą, należałoby zastosować około 20 000 procesorów, gdyż w takim systemie występuje aż tak duża liczba sygnałów sterujących jego pracą.

9.3. BAZY DANYCH O DOSTĘPIE RELIZOWANYM W CZASIE RZECZYWISTYM

Istnieje wiele aplikacji czasu rzeczywistego, które wymagają współpracy z systemami baz danych i wymagają jednocześnie dochowania ściśle określonych ograniczeń czasowych. Bazy danych o dostępie zrealizowanym w czasie rzeczywistym ułatwiłyby w stopniu znaczącym implementację rozszerzalnych oraz otwartych architektur programowych (ang. extensible and open software architectures) dla systemów czasu rzeczywistego, co pomogłoby aplikacjom czasu rzeczywistego w efektywnym zarządzaniu dużymi porcjami danych i informacji dzielonych pomiędzy poszczególnymi zadaniami. Powyższe idee posiadają fundamentalne znaczenie przy wyborze kierunków, w ja-

kich prowadzone mają być badania nad nowymi generacjami systemów czasu rzeczywistego. Tak więc jest rozwijanie baz danych, będących w stanie wspomagać aplikacje czasu rzeczywistego jest jak najbardziej pożądane.

Podstawowa trudność w zaimplementowaniu technologii baz danych dla systemów czasu rzeczywistego polega na tym, że architektury konwencjonalnych baz danych nie zostały zaprojektowane pod kątem uzyskania wymaganych w systemach czasu rzeczywistego gwarancji na maksymalny czas odpowiedzi. Większość konwencjonalnych systemów baz danych zostało opartych na odczycie informacji z dysków magnetycznych (ang. disc-based) oraz wykorzystuje logowanie poszczególnych transakcji (ang. transaction logging) i dwufazowe domykające protokoły (ang. two-phase locking protocols), w celu zapewnienia przesłania wszystkich porcji informacji we właściwej kolejności (ang. transaction atomicity and serialisability). Powyższe postępowanie prowadzi, w prawdzie, do zachowania integralności danych (ang. data integrity), ale daje w rezultacie duże i trudne do przewidzenia wartości czasów odpowiedzi systemu. Zatem nie można pozwolić sobie na podłączenie konwencjonalnego systemu baz danych z wielokrotnym dostępem (ang. conventional multiuser database system), takiego jak np. ORACLE lub SYBASE do układu, pracującego w czasie rzeczywistym kontrolera sterującego, przykładowo, przebiegiem procesu produkcji w kompleksowo zautomatyzowanej fabryce. Relatywnie wolne oraz zupełnie nieprzewidywalne czasy odpowiedzi baz danych, spowodowały by z pewnością naruszenie istniejących w systemie czasu rzeczywistego ograniczeń czasowych, co zostało wykazane w pracach [67] i [68].

Nieadekwatność konwencjonalnych systemów baz danych dla aplikacji systemów czasu rzeczywistego stała się przyczyną prowadzenia intensywnych prac badawczych w tej dziedzinie, czego wyrazem są między innymi prace [69], [70] i [71].

10. BIBLIOGRAFIA

1. C.M. Krishna, K.G. Shin: *Performance measures for multiprocessor controllers*. Performance'83, Amsterdam, The Netherlands: North-Holland, 1983, pp. 229--250.
2. J.A. Stankovic: *Misconceptions about real-time computing: A serious problem for next generation systems*. IEEE Comput., vol. 21, no. 10, pp. 10-19, Oct. 1988
3. J.R. Haritsa, M.J. Carey, M. Livny: *Data access scheduling in firm real-time database systems*. J. Real-Time Syst., vol. 4, no. 3, pp. 203-241, Sept. 1992
4. J.J. Molini, S.K. Maimon, P. H. Watson: *Real-time system scenarios*. Proc. Real-Time Systems Symp., Dec. 1990, pp. 214-225
5. S. Chodorow, F. Jahanian, M. Donner: *Run-time monitoring of real-time systems*. Proc. Real-Time Systems Symp., Dec. 1991, pp. 74-83.
6. A. Damm, J. Reisinger, W. Schwabl, H. Kopetz: *The real-time operating system of MARS*. ACM Operating Syst. Rev., vol. 23, no. 3, pp. 141-157, July 1989
7. K. G. Shin: *HARTS: A distributed real-time architecture*. IEEE Computers, vol. 24, no. 5, pp. 25-35, May 1991
8. J. A. Stankovic, K. Ramamritham: *The design of Spring kernel*. Proc. Real-Time Systems Symp., Dec. 1987, pp.146-157
9. R. Kiekhäfer, C.J. Walter, A. M. Finn, P. M. Thambidurai: *The MAFT architecture for distributed fault tolerance*. IEEE Trans. Comput., vol. 37, no. 4, Apr. 1988, pp. 398-405

10. T. B. comp
11. J. S. real-
12. M. S. mult
13. D.D. Distr
14. A. L.
15. G. A. time
16. M. I. mess
17. W. 2. netw
18. K.G. Com
19. H. L. On l
20. P. R. Sym
21. J. L. com
22. C. M. syst
23. C. M. Cor
24. F. Fau
25. D. I. time
26. J. F. syst
27. M. pp.
28. D. on
29. P. sys
30. J. Pri
31. L. Co
32. J. Inf
33. T. Jul
34. T. Jul
35. P. for

10. T. B. Smith, J. H. Lala: *Development and evaluation of a fault-tolerant multiprocessor (FTMP) computer: FTMP principles of operation*. NASA, May 1983
11. J. Stankovic: *SpringNet: A scalable architecture for high performance, predictable, distributed, real-time computing*. Tech. Rep. Univ. Of Massachusetts, Oct. 1991, pp. 74–91
12. M. S. Chen, K. G. Shin, D. D. Kandlur: *Addressing, routing and broadcasting in hexagonal mesh multiprocessors*. IEEE Trans. Comput., vol. 39, no. 1, Jan. 1990, pp. 10–18
13. D.D. Kandlur, K. G. Shin, D. Ferrari: *Real-time communication in multi-hop networks*. Proc. Distributed Computing Systems, May 1991, pp. 300–307
14. A. L. Tannenbaum: *Computer Networks*. Englewood Cliffs, Prentice-Hall, 1981
15. G. Agrawal, B. Chen, W. Zhao, S. Davari: *Guaranteeing synchronous message deadlines with the timed token protocol*. Proc. Distributed Computing Systems, June 1992, pp. 468–475
16. M. Hamdaoui, P. Ramanathan: *Selection of timed token protocol parameters to guarantee message deadlines*. Univ. Of Wisconsin-Madison, Nov. 1992
17. W. Zhao, K. Ramamriotham, J. A. Stankovic: *Synchronous bandwidth allocation in {FDDI} networks*. Proc. ACM Multimedia'93, 1993, pp. 31–38.
18. K.G. Shin, G. L. Dykema: *Distributed {I/O} architecture for {HARTS}*. Proc. 17th Int. Symp. On Computer Architectures, June 1990, pp. 332–342
19. H. L. Chen, N. F. Tzeng: *Fault-tolerant resource placement in hypercube computers*. Proc. Int. Conf. On Parallel Processing, Aug. 1991, pp. 517–524.
20. P. Ramanathan, S. Chalasani: *Resource placement in k-ary n-cubes*. Proc. Int. Parallel Processing Symp., pp. II-133-II-140, Aug. 1992
21. J. Liu, K. J. Lin, W. K. Shih, A. Yu, A. Y. Chung, W. Zhao: *Algorithms for scheduling imprecise computations*. Computer, vol. 24, no. 5, May 1991, pp. 58–69
22. C. M. Krishna, K. G. Shin, R. W. Butler: *Synchronisation and fault-masking in redundant real-time systems*. Dig. Papers, FTCS-14, June 1984, pp. 152–157
23. C. M. Krishna, K. G. Shin: *On scheduling tasks with a quick recovery from failure*. IEEE Trans. Computers, vol. C-35, no. 5, May 1986, pp. 448–455
24. F. Jahanian, A. Goyal: *A formalism for monitoring real-time constraints at run-time*. Proc. Fault-Tolerant Computing Symp. (FTCS-20), June 1990, pp. 148–155
25. D. Haban, K. G. Shin: *Application of real-time monitoring for scheduling tasks with random execution times*. IEEE Trans. Software Eng., vol. 16, no. 12, Dec. 1990, pp. 1374–1389
26. J. P. Tsai, K. Y. Fang, H. Y. Chen: *A noninvasive architecture to monitor real-time distributed systems*. Comput., vol. 23, no. 3, Mar. 1990, pp. 11–23
27. M. H. Graham: *Issues in real-time data management*. J. Real-Time Syst., vol. 4, no. 3, Sept. 1992, pp. 185–202
28. D. Dolev, J. Halpern: *On the possibility and impossibility of achieving clock synchronisation*. Proc. on Theory of Computing, Apr. 1984, pp. 504–511
29. P. Ramanathan, K. G. Shin, R. W. Butler: *Fault-tolerant clock synchronisation in distributed systems*. IEEE Computers, vol. 23, no. 10, Oct. 1990, pp. 33–42
30. J. Y. Halpern, B. Simons, R. Strong, D. Dolev: *Fault-tolerant clock synchronisation*. Proc. Principles of Distributed Computing, 1984, pp. 89–102
31. L. Lamport, P. M. Melliar-Smith: *Synchronising clocks in the presence of faults*. J. Assoc. Comput. Mach., vol. 32, no. 1, Jan. 1985, pp. 52–78
32. J. Lundelius-Welch, N. Lynch: *A new fault-tolerant algorithm for clock synchronisation*. Informat. And Computat., vol. 77, no. 1, 1988, pp. 1–36
33. T. K. Srikanth, S. Toueg: *Optical clock synchronisation*. J. Assoc. Comput. Mach., vol. 34, no. 3, July 1987, pp. 626–645
34. T. K. Srikanth, S. Toueg: *Optimal clock synchronisation*. J. Assoc. Comput. Mach., vol. 34, no. 3, July 1987, pp. 626–645
35. P. Ramanathan, D. D. Kandlur, K. G. Shin: *Hardware assisted software clock synchronisation for homogeneous distributed systems*. IEEE Trans. Comput., vol. 39, no. 4, Apr. 1990, pp. 514–524

36. J. L. Kessels: *Two designs of a fault-tolerant clocking system*. IEEE Trans. Comput., vol. C-33, no. 10, Oct. 1984, pp. 912-919
37. C. M. Krishna, K. G. Shin, R. W. Butler: *Synchronisation and Fault-masking in redundant real-time systems*. Dig. Papers, FTCS-14, June 1984, pp. 152-157
38. T. B. Smith: *Fault-tolerant clocking system*. Proc. Fault-Tolerant Computing Symp., 1981, pp. 262-264
39. N. Vasanthavada, P. N. Marinos: *Synchronisation of fault-tolerant clocks in the presence of malicious failures*. IEEE Tran. Comput., vol. 37, no. 4, April 1988, pp. 440-448
40. K. G. Shin, P. Ramanathan: *Clock synchronisation of a large multiprocessor system in the presence of malicious faults*. IEEE Trans. Comput., vol. C-36, no. 1, Jan. 1987, pp. 2-12
41. K. G. Shin, P. Ramanathan: *Transmission delays in hardware clock synchronisations*. IEEE Trans. Comput., vol. 37, no. 11, Nov. 1988, pp. 1465-1467
42. P. Ramanathan, D. D. Kandlur, K. G. Shin: *Hardware assisted software clock synchronisation for homogeneous distributed systems*. IEEE Trans. Comput., vol. 39, no. 4, Apr. 1990, pp. 514-524
43. F. Cristian: *Probabilistic clock synchronisation*. Tech. Rep. RJ 6432, IBM Almaden Research Center, Sept. 1988
44. A. Olson, K. G. Shin: *Probabilistic clock synchronisation in large distributed systems*. Proc. 11th Int. Conf. On Distributed Computer Systems, May 1991, pp. 290-297
45. K. G. Shin, C. M. Krishna, Y. H. Lee: *A unified method for evaluating real-time computer controller and its application*. IEEE Trans. Automat. Contr., vol. AC-30, no. 4, pp. 357-366
46. A. Gosiewski, A. Olbrot: *The effect of feedback delays on the performance of multivariable linear control systems*. IEEE Trans. Automat. Contr., vol. AC-25, no. 4, Aug. 1980, pp. 729-734
47. K. G. Shin, X. Cui: *Effects of computing time delay on real-time control systems*. Proc. 1988 American Control Conf., 1988, pp. 1071-1076
48. Z. V. Rekasius: *Stability of digital control with computer interruption*. IEEE Trans. Automat. Contr., vol. AC-32, np. 4, Apr. 1986, pp. 356-359
49. K. Hari, Y. Satoh: *Stability of a system with variable time delay*. IEEE Trans. Automat. Contr., vol. AC-25, no. 3, June 1980, pp. 552-554
50. A. Belleisle: *Stability of systems with nonlinear feedback through randomly time-varying delays*. IEEE Trans. Automat. Contr., vol. AC-20, no. 1, Feb. 1975, pp. 67-75
51. K. G. Shin, H. Kim: *Derivation and application of hard deadlines for real-time control systems*. IEEE Trans. System, Man, Cybern., vol. 22, no. 6, Nov./Dec. 1992, pp. 1403-1413
52. T. J. Laffey, P. A. Cox, J. L. Schmidt, S. M. Kao, J. Y. Read: *Real-time knowledgebased systems*. AI Mag., vol. 9, no. 1, 1988, pp. 27-45
53. E. H. Durfee: *A cooperative approach to planning for real-time control*. Proc. Workshop on Innovative Approaches to Planning, Scheduling and Control, Nov. 1990, pp. 277-283
54. C. J. Paul, A. Acharya, B. Black, J. K. Strosnider: *Reducing problem-solving variance to improve predictability*. Commun. ACM, vol. 34, no. 8, Aug. 1991, pp. 81-93
55. M. Boddy, T. Dean: *Solving time-dependant planning problems*. Proc. Int. Joint Conf. On Artificial Intelligence, Aug. 1989, pp. 979-984
56. E. J. Horvitz: *Reasoning about beliefs and actions computational resource constraints*. Proc. Workshop on Uncertainty in AI, 1987
57. R. E. Korf: *Real-time search for dynamic planning*. Proc. AAAI Spring Symp. On Planning in Uncertain, Unpredictable, Channing Environments, 1990
58. P. E. Agre, D. Champain: *Pengi: An implementation of a theory of activity*. Proc. Nat. Conf. On Artificial Intelligence, Morgan Kaufmann: 1987, pp. 268-272
59. R. A. Brooks: *Arobust layered control system for a mobile robot*. IEEE J. Robotics Automat., vol. RA-2, no. 1, March. 1986, pp. 14-22
60. R.C. Arkin: *Integrating behavior al, perceptual and world knowledge in reactive navigation*. Robotics and Autonomous Systems 6, 1990, pp. 105-122
61. S. Hanks, R. J. Firby: *Issues architectures for planning and execution*. Proc. Workshop on Innovative Approaches to Planning, Scheduling and Control, Nov. 1990, pp. 59-70

62. D. P. Miller, E. Gat: *Exploiting known topologies to navigate with low-computation sensing*. Proc. SPIE Sensor Fusion Conf., Nov. 1990
63. R. Simons: *An architecture for coordinating planning, sensing and action*. Proc. Workshop on Innovative Approaches to Planning, Scheduling and Control, Nov. 1990, pp. 292–297
64. D. J. Musliner, E. H. Durfee, K. G. Shin: *CIRCA: A cooperative intelligent real-time control architecture*. IEEE Trans. Systems, Man, Cybern., vol. 23, no. 6, Nov./Dec. 1993
65. V. R. Lesser, J. Palvina, E. Durfee: *Approximate processing in real-time problem solving*. AI Mag., vol. 9, no. 1, 1988, pp. 49–61
66. J. E. Arnold: *Experiments with the subsumption architectures*. Conf. On Artificial Intelligence Applications, 1989, pp. 93–100
67. L. Sha, R. Rajkumar, J. P. Lehoczky: *Concurrency control for distributed real-time databases*. SIGMOD Rec., vol. 17, no. 1, Mar. 1988, pp. 82–98
68. M. Singhal: *Issues and approaches to design of real-time database systems*. SIGMOD Rec., vol. 17, no. 1, Mar. 1988, pp. 19–33
69. M. H. Graham: *Issues in real-time data management*. J. Real-Time Syst., vol. 4, no. 3, Sept. 1992, pp. 185–202
70. V. B. Lortz, K. G. Shin: *MDARTS: A multiprocessor data base architecture for real-time systems*. Tech. Rep. CSE-TR-155-93, CSE Div., Dep. Elec. Eng. Comput. Sci., Univ. Of Michigan, Ann Arbor, MI, Mar. 1993
71. K. Ramamrithan: *Real-time databases*. IEEE Trans. Comput., vol. 39, no. 4, Apr. 1990, pp. 514–524

M. GAJER

THE CHOSEN PROBLEMS OF ANALYSIS AND DESIGN OF DISTRIBUTED REAL-TIME SYSTEMS

Summary

The paper surveys the state of the art in real-time computing. In the paper some basic concepts related with real-time computing field are introduced. Further the basic key issues in the design of real-time systems are identified. Solutions proposed in literature for tackling these issues are also briefly discussed. In the paper the different types of distributed real-time systems architectures is discussed. The operation systems for distributed real-time systems are also discussed. The synchronisation issues in real-time systems are also taken into account. The article is terminated by the discussion of potential application of real-time systems.

Keywords: Real-time systems, operating systems, task scheduling and allocation, fault tolerance, applications

a
c
c
z
z
s
c
b
z
l
S

A
i w jak
się, że
wszelk
rzeczy
wystę
rys. 1.
relacje

Zagadnienia przydziału i planowania zadań w systemach czasu rzeczywistego

MIROSŁAW GAJER

*Katedra Automatyki AGH
Al. Mickiewicza 30, 30-059 Kraków
e-mail: mgajer@ia.agh.edu.pl*

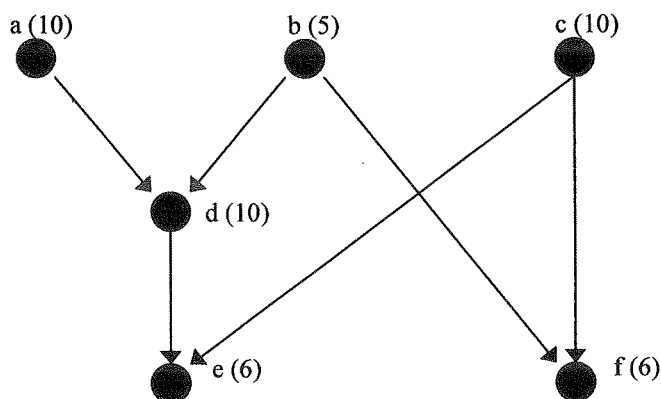
*Otrzymano 2001.02.23
Autoryzowano 2001.07.25*

W artykule dokonano przeglądu rozwiązań stosowanych współcześnie w dziedzinie alokacji i szeregowania zadań dla systemów czasu rzeczywistego. Zamieszczono zwięzłe charakterystyki czterech podstawowych schematów postępowania podczas szeregowania zadań dla systemów czasu rzeczywistego. Omówiono podejścia oparte na: statycznym szeregowaniu z tablicą zadań, szeregowaniu z wyłączeniem priorytetowym, szeregowaniu dynamicznym z planowaniem oraz szeregowaniu dynamicznym ukierunkowanym na jak najlepsze działanie systemu. Dokonano także przeglądu technik bezpośrednio związanych z problematyką przydziału i szeregowania zadań, takich jak: szeregowanie z zapewnieniem odporności systemu na błędy oraz techniką obliczeń nieprecyzyjnych. Rozważano zagadnienia szeregowania zadań zarówno w systemach jednoprocessorowych, jak i wieloprocessorowych. Na zakończenie artykułu podano warunek na szeregowalność zbioru niezależnych zadań okresowych.

Słowa kluczowe: system czasu rzeczywistego, alokacja i szeregowanie zadań, odporność na błędy, obliczenia nieprecyzyjne

1. WPROWADZENIE DO ZAGADNIEŃ SZEREGOWANIA ZADAŃ

Alokacja zadań i ich szeregowanie jest procesem polegającym na określeniu gdzie i w jakiej chwili czasu każde z zadań może rozpocząć wykonywanie, przy czym zakłada się, że dany jest zbiór wszystkich zadań występujących w systemie, jak i określone są wszelkie dostępne zasoby systemu. Dla przykładu rozważona zostanie aplikacja czasu rzeczywistego, składająca się z sześciu zadań *a*, *b*, *c*, *d*, *e*, *f*, pomiędzy którymi występują relacje poprzedzania i, które posiadają ograniczenia czasowe, jak pokazano na rys. 1. Na rysunku tym czarne punkty reprezentują zadania, a strzałki odzwierciedlają relacje poprzedzania, jakie zachodzą pomiędzy tymi zadaniami. Z rysunku widać, że



Rys. 1. Przykład aplikacji czasu rzeczywistego

zadania a i b muszą zostać zakończone, zanim zadanie d zostanie rozpoczęte, ponieważ istnieją strzałki skierowane zarówno od zadania a do d , jak i od b do d .

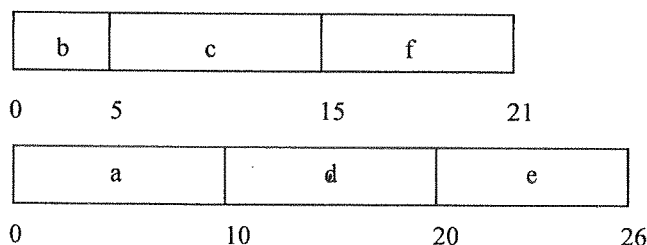
Każde z zadań posiada skojarzoną z nim wagę, która reprezentuje czas potrzebny do jego wykonania. Analizując ograniczenia czasowe widać, że zadania e i f muszą zakończyć swoje wykonywanie (zgodnie z ograniczeniami czasowymi), odpowiednio po 31 i 16 jednostkach czasu. Przy czym założono, że wszystkie zadania są gotowe do wykonania w czasie 0 i wzięto pod uwagę relacje poprzedzania zachodzące pomiędzy zadaniami.

Na rys. 2(a) i (b) pokazano dwa możliwe uszeregowania zadań rozpatrywanej aplikacji czasu rzeczywistego dla systemu komputerowego składającego się z dwóch procesorów. Na rys. 2(a) zadania b , c , f zostały przypisane do procesora pierwszego, natomiast zadania pozostałe do procesora drugiego. Na pierwszym z procesorów zadanie b wykonuje się począwszy od czasu 0 do 5, zadanie c wykonuje się począwszy od czasu 5 do 15 itd. Dla drugiego procesora sytuacja wygląda analogicznie. Jednakże przy takiej alokacji i uszeregowaniu zadań, zadanie f nie zakończy swojego wykonywania po upływie 16 jednostek czasu. Zatem w tym przypadku ograniczenia czasowe systemu zostaną naruszone. Z kolei alokacja i uszeregowanie zadań zaprezentowane na rys. 2(b) spełniają zarówno relacje poprzedzania zadań, jak i narzucone na poszczególne zadania ograniczenia czasowe. Zatem szeregowanie zadań z rys. 2(b) jest odpowiednie dla aplikacji czasu rzeczywistego. Problem alokacji i szeregowania zadań dla aplikacji czasu rzeczywistego polega na znalezieniu odpowiedniego uszeregowania, jak na rys. 2(b) przy zadanych wszystkich parametrach aplikacji, jak na rys. 1.

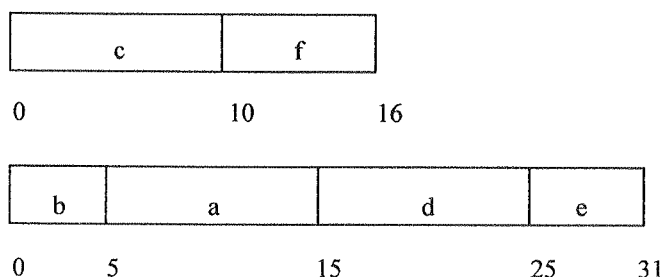
Problem szeregowania pojawia się w wielu dziedzinach nauki związanych przykładowo z przetwarzaniem równoległym, badaniami operacyjnymi, syntezą systemów, jednakże w związku z systemami czasu rzeczywistego istnieją pewne dodatkowe, kluczowe ograniczenia, które sprawiają, że problem ten staje się w swej istocie odmienny.

W większości systemów nie związanych z dziedziną czasu rzeczywistego, głównym celem szeregowania jest minimalizacja całkowitego czasu wymaganego do wykonania

wszystkich
stego
ne za
rys. 2
z rys
przyk
nych
czaln
nia c
I
doko
jedyn
zadan
szere
przy
wani
pozo
szere
rzeci
pom



Rys. 2(a). Szeregowanie i alokacja zadań dla systemu czasu rzeczywistego z naruszeniem ograniczeń czasowych



Rys. 2(b). Szeregowanie i alokacja zadań dla systemu czasu rzeczywistego z zachowaniem ograniczeń czasowych

wszystkich zadań związanych z daną aplikacją. Natomiast w systemach czasu rzeczywistego, głównym celem jest zagwarantowanie dotrzymania narzuczonych na poszczególne zadania ograniczeń czasowych. Dla przykładu wszystkie zadania przedstawione na rys. 2(a) zakończą swoje wykonywanie po czasie 26 jednostek, podczas gdy zadania z rys. 2(b) zostaną zakończone dopiero po 31 jednostkach czasu. Tak więc pierwszy przykład szeregowania zadań byłby, z pewnością, preferowany w systemach nie związanych z dziedziną czasu rzeczywistego, podczas gdy tylko drugi przykład jest dopuszczalny dla aplikacji czasu rzeczywistego, ponieważ w pierwszym przypadku ograniczenia czasowe nałożone na pewne z zadań nie są dochowane.

Podziału algorytmów szeregowania zadań dla aplikacji czasu rzeczywistego można dokonać na wielu płaszczyznach. Pewne algorytmy szeregowania znajdują zastosowanie jedynie do zadań periodycznych, podczas gdy pozostałe są przewidziane wyłącznie dla zadań aperiodycznych. Istnieje bardzo mało algorytmów dających się zastosować do szeregowania obu typów zadań, ponieważ podejścia zastosowane w tych dwóch różnych przypadkach różnią się diametralnie od siebie. Analogicznie pewne algorytmy szeregowania mogą być wykorzystywane jedynie do zadań wywłaszczalnych, podczas gdy pozostałe wymagają zadań nie podlegających wywłaszczeniu. Na postać algorytmów szeregowania mają także wpływ inne charakterystyki zadań tworzących aplikację czasu rzeczywistego, takie jak: ostrość ograniczeń czasowych, krytyczność zadań, zależności pomiędzy zadaniami, ograniczenia zasobów systemu i inne.

Algorytmy szeregowania różnią się również znacząco w zależności od typu architektury systemu komputerowego, dla którego zostały zaimplementowane. Pewne algorytmy są przewidziane dla systemów jednoprocessorowych, podczas gdy pozostałe dla wieloprocessorowych. Algorytmy przewidziane dla systemów wieloprocessorowych mogą także różnić się w zależności od tego, czy są to systemy ze wspólną pamięcią, czy też systemy, w których komunikacja pomiędzy poszczególnymi jednostkami obliczeniowymi została oparta na przesyłaniu wiadomości. Struktura sieci połączeń między procesorami może również posiadać wpływ na postać algorytmów szeregowania zadań.

Wreszcie mogą wystąpić różnice w celach, ku którym dążą poszczególne algorytmy szeregowania zadań. Większość algorytmów przyjmuje, że wszystkie zadania, podlegające szeregowaniu, posiadają ostre ograniczenia czasowe. Jakkolwiek w pracy [1] zaproponowano podejście, w którym przyjmuje się, że zadanie składa się z części obowiązkowej i części opcjonalnej. Część obowiązkowa musi zostać zakończona przed upływem narzuconego na zadanie ograniczenia czasowego, natomiast część opcjonalna może ulec zakończeniu, ale nie musi.

2. ALGORYTMY SZEREGOWANIA ZADAŃ DLA SYSTEMÓW JEDNOPROCESOROWYCH

W pracy [2] autorzy zaproponowali dwie klasy algorytmów szeregowania zadań dla systemów jednoprocessorowych. Pierwszy typ algorytmu oznaczony skrótem RMS (ang. Rate Monotonic Scheduling) opierał się na zasadzie statycznego przydziału priorytetów do zadań, natomiast drugi EDF (ang. Earliest Deadline First) implementował mechanizmy dynamicznego przydziału priorytetów, w trakcie wykonywania zadań. Oba typy algorytmów odnosiły się do zadań, na które zostały narzucone pewne dodatkowe ograniczenia w postaci ich niezależności, wyłączalności i periodyczności. Ponadto przyjęto, że wszystkie zadania podlegające szeregowaniu posiadają ostre ograniczenia czasowe. W pracy tej podano prostą charakterystykę zbioru zadań, podlegających szeregowaniu przy użyciu tych dwóch powyższych typów algorytmów. Wykazano także, że algorytm RMS jest optymalny spośród wszystkich algorytmów ze statycznym przydziałem priorytetów, w tym sensie, że jeśli jakieś zadanie jest szeregowalne przez dowolny z algorytmów ze statycznym przydziałem priorytetów, to jest również szeregowalne przez algorytm RMS. Podobnie wykazano, że algorytm EDF jest optymalny, spośród wszystkich algorytmów z dynamicznym przydziałem priorytetów, w przypadku szeregowania na systemie jednoprocessorowym zadań o wymienionych wcześniej charakterystykach.

Zaproponowane w pracy [2] algorytmy szeregowania operowały na bardzo restrykcyjnym zbiorze zadań. Jednakże po opublikowaniu tej pracy zaproponowano liczne rozszerzenia tych algorytmów, tak aby mogły zostać one zaimplementowane do szeregowania zadań, na które nałożono pewne dodatkowe ograniczenia w postaci zależności pomiędzy zadaniami. Przykładowo zaproponowano rozwiązania zapewniające kontrolowany dostęp do wspólnych zasobów, a także pewne modyfikacje tych algorytmów

uwzględniające w procesie szeregowania również zadania aperiodyczne i sporadyczne. Wykazano także, że rezultaty uzyskane w wyniku rozwoju teorii szeregowania zadań dla systemów jednoprocessorowych, mogą być wykorzystane do modelowania bardziej złożonych sytuacji, takich jak rozproszone szeregowanie wiadomości w sieciach komputerowych z wielokrotnym dostępem; czego przykładem może być sieć FDDI.

3. ALGORYTMY SZEREGOWANIA ZADAŃ DLA SYSTEMÓW WIELOPROCESOROWYCH

Problemy występujące w przypadku szeregowania zadań w systemach wieloprocessorowych są znacząco odmienne od zagadnień analogicznych, ale występujących w systemach jednoprocessorowych. W przypadku szeregowania zadań dla systemów wieloprocessorowych problem polega nie tylko na określeniu czasu, w którym dane zadanie będzie wykonywane, ale także na wskazaniu jednostki obliczeniowej, która to zadanie będzie wykonywać. Tak więc zagadnienia alokacji i szeregowania zadań, w tym przypadku muszą być rozpatrywane łącznie. W związku z tym pojawiają się zagadnienia związane z dostępnością potrzebnych zasobów w obrębie jednostki, w ramach której dane zadanie będzie wykonywane, koniecznością rozwiązywania konfliktów podczas komunikacji poszczególnych jednostek poprzez sieć połączeń i wiele innych. Wszystkie te czynniki sprawiają, że problem szeregowania zadań w systemach wieloprocessorowych, ze swej natury staje się znacznie trudniejszy do rozwiązania.

Podobnie jak w przypadku systemów jednoprocessorowych, szeregowanie zadań w systemach wieloprocessorowych może być statyczne lub dynamiczne. W przypadku algorytmów statycznych przydział zadań dla poszczególnych procesorów oraz przedziały czasu, w których dane zadania są wykonywane, są ściśle zdeterminowane *a priori*. Algorytmy te zostały szczegółowo omówione w pracy [3]. Niestety sam problem przydziału zadań do jednostek obliczeniowych jest bardzo złożony, nawet jeśli pominie się nałożone na zadania ograniczenia czasowe. W większości przypadków zagadnienie przydziału zadań należy do klasy problemów NP-zupełnych. Przykładowo w pracy [4] pokazano, że znalezienie optymalnego przydziału dla zadań z podanym grafem zależności komunikacyjnych do czterech lub więcej jednostek obliczeniowych, charakteryzujących się różnymi prędkościami obliczeń, jest problemem NP-zupełnym. Tak więc większość istniejących rozwiązań z tej dziedziny polega na podjęciu próby znalezienia rozwiązań suboptymalnych przy użyciu heurystyk.

Najczęściej wykorzystywane heurystyki należą do kategorii oznaczonej skrótem LS (ang. List Scheduling). W metodzie tej reguła heurystyczna wykorzystywana jest najpierw do uporządkowania wszystkich zadań występujących w systemie. Zadania poddawane są procesowi szeregowania we wcześniejszym określonym porządku oraz są one najczęściej przydzielane i szeregowane w ramach tej jednostki obliczeniowej, która potrafi zagwarantować ich najkrótszy czas wykonania. Przykładowo w pracach [3], [5] i [6] zadania zostały uporządkowane ze względu na wzrastający przedział czasu, zwany (ang. Laxity), przez jaki dane zadanie może oczekiwać na rozpoczęcie wykonywania,

tak aby nie zostały naruszone jego ograniczenia czasowe, a następnie tak uzyskane uporządkowanie zostało wykorzystane do szeregowania tych zadań. W pracach [7] i [8] zaproponowano inne metody znajdowania statycznego uszeregowania zadań, oparte odpowiednio na metodach podziału i ograniczeń (ang. Branch-and-Bound) oraz algorytmach genetycznych.

Jak pokazano w pracy [3], pomimo trudności w wyszukiwaniu optymalnego rozwiązania, algorytmy statyczne są wykorzystywane do szeregowania zadań periodycznych z ostrymi ograniczeniami czasowymi. Główna zaleta takiego podejścia polega na tym, że jeśli rozwiązanie zostanie już znalezione, wtedy można być pewnym, że wszystkie ograniczenia czasowe nałożone na zadania zostaną dotrzymane. Jednakże podejście to nie może być zastosowane do zadań aperiodycznych, dla których czasy rozpoczęcia wykonywania i ograniczenia czasowe zwykle nie są znane a priori. Szeregowanie takich zadań w systemach wieloprocesorowych wymaga zastosowania algorytmów scentralizowanych i rozproszonych. W przypadku scentralizowanym wszystkie zadania przybywają do centralnego procesora, gdzie następnie są przydzielane do pozostałych procesorów systemu. Przykłady scentralizowanych algorytmów przydziału i szeregowania zadań dla systemów wieloprocesorowych można znaleźć w pracach [9], [10] i [11]. Główna zaleta podejścia scentralizowanego polega na tym, że tylko centralny procesor musi znać obciążenie pozostałych procesorów systemu, w celu określenia czy dochowanie ograniczeń czasowych dla przybywających do systemu zadań może być zagwarantowane.

W przypadku rozproszonych algorytmów szeregowania zadań, zadania przybywają w sposób od siebie niezależny do każdej z jednostek obliczeniowych systemu. W momencie, w którym zadanie obliczeniowe przybywa do danego procesora, lokalny program szeregujący, zaimplementowany na tym procesorze musi określić, czy w ramach tego procesora zadanie można zaszeregować w ten sposób, aby nie zostały naruszone jego ograniczenia czasowe. Operacja ta nosi nazwę planowania transferu zadań (ang. Transfer Policy). Zadanie może zostać przyjęte do wykonywania w ramach danej jednostki obliczeniowej, jeśli możliwe jest dla tego zadania zagwarantowanie dotrzymania ograniczeń czasowych, bez niebezpieczeństwa naruszenia ograniczeń czasowych zaszergowanych wcześniej zadań. Jeżeli warunek ten nie może być spełniony, wówczas lokalny program szeregujący podejmuje próbę znalezienia jednostki obliczeniowej, która jest w stanie zagwarantować dochowanie ograniczeń czasowych dla danego zadania. Operacja ta zwana jest planowaniem umiejscowienia zadania (ang. Location Policy). Algorytmy szeregowania zadań dla środowiska rozproszonego zostały zaproponowane między innymi w pracach [12], [13], [14], [15] i [16]. Algorytmy te różnią się między sobą głównie ze względu na zastosowaną strategię przydziału jednostki obliczeniowej do dynamicznie napływających zadań. Różnice te bazują głównie na zastosowanych metodach pozyskiwania i gromadzenia informacji o stanie pozostałych jednostek systemu. Operacje te zwane są planowaniem informacji (ang. Information Policy).

4. METRYKI STOSOWANE W CELU POMIARU WYDAJNOŚCI SYSTEMÓW CZASU RZECZYWISTEGO

Szeregowanie zadań polega na alokacji zasobów systemu oraz przydziału czasu pracy jednostek obliczeniowych do poszczególnych zadań w taki sposób, aby spełnione były pewne wymagania dotyczące wydajności systemu. Spośród wszystkich zagadnień związanych z tematyką systemów czasu rzeczywistego, szeregowanie zadań obliczeniowych jest prawdopodobnie dziedziną, w której badania prowadzone są najbardziej intensywnie. Powodem takiego stanu rzeczy jest powszechnie uznana zasada, według której podstawowym problemem, wymagającym rozwiązania w systemach czasu rzeczywistego jest zagwarantowanie dochowania ograniczeń czasowych przez wszystkie zadania obliczeniowe. Ponadto szeregowanie zadań jest dobrze zdefiniowanym i stawiającym przed badaczami poważne wyznania problemem. W punkcie kolejnym wszystkie dotychczasowe osiągnięcia w dziedzinie szeregowania zadań dla systemów czasu rzeczywistego zostaną podzielone na kilka kategorii ze względu na pewne fundamentalne zasady, które legły u podstaw konstrukcji związanych z nimi algorytmów. Jednakże wcześniejszego omówienia wymagają powszechnie stosowane różne metryki, służące do pomiaru wydajności systemów czasu rzeczywistego (ang. performance metrics).

Rodzaj metryki, w oparciu o który podejmowane są decyzje o szeregowaniu zadań, zależy od aktualnego obszaru zastosowań danego systemu czasu rzeczywistego. Przykładowo dla statycznego systemu nie będącego systemem czasu rzeczywistego metryka może zostać zdefiniowana jako całkowita długość uszeregowania zadań w czasie, a zatem głównym celem projektanta takiego systemu będzie jej minimalizacja. W przypadku dynamicznych systemów czasu rzeczywistego metryka może być zdefiniowana, jako czas odpowiedzi na zdarzenia w systemie i w tym przypadku głównym celem projektanta systemu będzie również jej minimalizacja mierzonej wartości. W systemach takich metryka może być także zdefiniowana jako przepustowość systemu, czyli ilość zadań, które mogą być wykonane w jednostce czasu, w tym jednak przypadku projektant systemu musi dążyć do maksymalizacji mierzonej wartości. Jednakże zarówno w statycznych, jak i dynamicznych systemach czasu rzeczywistego głównym celem, ku któremu należy dążyć jest dotrzymanie ograniczeń czasowych. Fakt ten powoduje wprowadzenie zupełnie odmiennych metryk dla systemów czasu rzeczywistego.

Różnorodność metryk, jakie zostały zaproponowane dla systemów czasu rzeczywistego, jest spowodowana zróżnicowaniem typów tych systemów, ze względu na ich powiązania z aplikacjami ze świata rzeczywistego oraz odmiennością wymagań nałożonych na każdy z takich systemów. Fakt ten staje się często przyczyną trudności jakie występują w przypadku podjęcia próby porównania różnych algorytmów szeregowania zadań. Przyczyną kolejnych trudności jest fakt wykorzystywania w praktyce różnych typów charakterystyk zadań. Z zadaniami można powiązać takie wielkości, jak czas wykonania obliczeń, wymagania odnośnie zasobów systemu, stopień ważności poszczególnych zadań (czasami zwany również priorytetem zadań, lub ich krytycznością), relacje poprzedzania zadań, wymagania odnośnie komunikacji zadań i oczywiście ich ograniczenia czasowe. Jeżeli dane zadanie jest periodyczne, istotną wielkością go

charakteryzującą staje się jego okres. Jeżeli natomiast zadanie jest aperiodyczne, ważną wielkością staje się jego ograniczenie czasowe. Zadanie periodyczne może również posiadać ograniczenie czasowe, przed upływem którego musi zakończyć swoje wykonywanie. To ograniczenie czasowe może być równe okresowi zadania, chociaż nie jest to wymóg konieczny. Zarówno zadania periodyczne, jak i aperiodyczne mogą posiadać ograniczenia odnośnie czasu rozpoczęcia ich wykonywania.

W systemach czasu rzeczywistego rozważane są następujące metryki, służące do pomiaru jakości pracy i wydajności systemu. W przypadku statycznych systemów czasu rzeczywistego należy poszukiwać takiego statycznego uszeregowania zadań, które będzie zapewniać zachowanie ograniczeń czasowych dla wszystkich zadań. W przypadku istnienia wielu dopuszczalnych wariantów szeregowania zadań, drugorzędna metryka, w której dokonywana będzie maksymalizacja średniego czasu wcześniejszego zakończenia zadań (ang. *average earliness*), może być wykorzystana celem dokonania wyboru optymalnego rozwiązania. Jeżeli natomiast zadań nie da się uszeregować w taki sposób, aby były spełnione wszystkie ograniczenia czasowe, wówczas może być wykorzystana kolejna metryka w celu minimalizacji średniego czasu opóźnienia zadań (ang. *average tardiness*), co umożliwi dokonanie wyboru optymalnego rozwiązania. W ogólności, w dynamicznych systemach czasu rzeczywistego nie można *a priori* zagwarantować dochowania wszystkich ograniczeń czasowych, zatem do analizy takich systemów może zostać wykorzystana metryka, w której maksymalizacji podlegać będzie liczba zadań dla których spełnione będą ograniczenia czasowe.

Problemem pokrewnym doborowi metryki pomiaru jakości i wydajności pracy systemu czasu rzeczywistego jest wyznaczenie stopnia determinizmu zachowania się systemu, w przypadku wyboru konkretnej metody szeregowania zadań. To znaczy dla każdego z zadań należy obliczyć prawdopodobieństwo, że nie zostanie naruszone jego ograniczenie czasowe.

Godnym odnotowania uwagi jest fakt, że problem szeregowania zadań rozpatrywany jest, z różnej perspektywy, przez dwa odmienne środowiska badawcze. W dziedzinie badań operacyjnych poszukiwania koncentrują się wokół problemów planowania prac, mających podlegać wykonaniu (ang. *job-shop*) i przepływu zasobów (ang. *flow-shop*), przy czym ograniczenia czasowe są lub nie są w tym wypadku uwzględnione. Jako przykłady można w tym miejscu wymienić zagadnienia alokacji mocy produkcyjnych, planowanie kolejności wykonywania prac, planowanie porządku, w jakim wykorzystane zostaną maszyny oraz inne podobne zagadnienia, będące przedmiotem badań operacyjnych. Typ zasobów rozważanych przez uczonych w dziedzinie badań operacyjnych (maszyny, stanowiska pracy w fabryce itp.) oraz to, w jaki sposób prace wykorzystują te zasoby (np. dana praca może wymagać użycia wszystkich maszyn w określonym porządku), jest zupełnie odmienny od zasobów rozważanych przez informatyków (np. cykle maszynowe procesora, wymagana wielkość bloku pamięci itp.). Ponadto prace w danej chwili wykorzystują zwykle tylko jedną maszynę. Także aktywności na obszarze fabryki posiadają znacznie większą ziarnistość czasową, niż zadania rozważane przez informatyków. Metryki wykorzystywane w dziedzinie badań operacyjnych służą takim celom jak: minimalizacja maksymalnego kosztu, minimali-

zacja ca
nia pos
nionych
runkowa
informa
mienion
intensy
Z techn
można

Jak
nego za
uwagę
uszereg
wykona
w celu
czasow
schemu
dokonu
to czy
przepr
jednost
można

• A
przypa
istnien
tablicy
momen

•
driven
analiza
przypa
system
highes

•
proach
liwość
napływ

zacja całkowitego czasu wykonania danych prac, minimalizacja sumy czasów wykonania poszczególnych prac, minimalizacja opóźnień oraz minimalizacja liczby opóźnionych prac. Także techniki wykorzystywane w dziedzinie badań operacyjnych ukie-
runkowane są raczej na rozwiązania statyczne, podczas gdy techniki wypracowane przez informatyków koncentrują się głównie wokół rozwiązań dynamicznych. Pomimo wymienionych powyżej różnic, pewne abstrakcyjne zagadnienia, będące przedmiotem intensywnych studiów obu społeczności naukowych, posiadają wiele cech wspólnych. Z technikami szeregowania zadań, wykorzystywanymi w dziedzinie badań operacyjnych można zapoznać się w pracach [17], [18], [19], [20] i [21].

5. PODSTAWOWE ZASADY STOSOWANE PRZY SZEREGOWANIU ZADAŃ W SYSTEMACH CZASU RZECZYWISTEGO

Jak już wspomniano w jednym z poprzednich punktów zapewnienie deterministycznego zachowania się systemu, jest najważniejszym zagadnieniem, które należy brać pod uwagę na etapie projektowania systemów czasu rzeczywistego. Analiza możliwości uszeregowania zadań (ang. schedulability analysis) oraz sprawdzenie możliwości ich wykonania (ang. feasibility checking) w czasie rzeczywistym, muszą być dokonane w celu określenia, czy dla poszczególnych zadań zostaną zachowane ograniczenia czasowe. Pewne podstawowe zasady wykorzystywane przy szeregowaniu zadań (ang. scheduling paradigms) mogą zostać sformułowane, w zależności od tego czy system dokonuje samoczynnie analizy istniejących możliwości uszeregowania zadań. Jeżeli tak, to czy jest to analiza przeprowadzana statycznie, czy dynamicznie oraz czy w rezultacie przeprowadzonej analizy powstaje plan, zgodnie z którym zadania są przydzielane do jednostek obliczeniowych w odpowiednich chwilach czasu. Bazując na powyższym można wyróżnić następujące klasy algorytmów alokacji i szeregowania zadań:

- Algorytmy statyczne z tablicą zadań (ang. static table driven approaches). W tym przypadku przeprowadzana jest statyczna analiza zależności pomiędzy zadaniami oraz istniejących możliwości ich uszeregowania. Powstały w ten sposób plan zadań w postaci tablicy, jest wykorzystywany w czasie pracy systemu w celu podjęcia decyzji, w którym momencie czasu dane zadanie może rozpocząć swoje wykonywanie.

- Algorytmy statyczne z wyłączeniem priorytetowym (ang. static priority-driven preemptive approaches). Również w tym przypadku dokonywana jest statyczna analiza możliwości uszeregowania zadań, ale w przeciwieństwie do poprzedniego przypadku, nie jest konstruowany żaden plan wykonywania zadań. W czasie pracy systemu wykonywane jest to zadanie, które posiada aktualnie najwyższy priorytet (ang. highest priority first).

- Algorytmy dynamiczne oparte na planowaniu (ang. dynamic planning-based approaches). W przeciwieństwie do obu omówionych uprzednio klas algorytmów, możliwość wykonania zadania sprawdzana jest w czasie pracy systemu, to znaczy, że napływające do systemu w sposób dynamiczny zadania są akceptowane do wykonania,

tylko wtedy, jeżeli zostanie wykazana możliwość ich zakończenia przed upływem ich ograniczeń czasowych. O takich zadaniach mówi się, że są one gwarantowane (ang. guaranteed) ze względu na dochowanie ich ograniczeń czasowych. Jednym z rezultatów przeprowadzonej analizy możliwości wykonania zadań jest plan, na podstawie, którego system podejmuje decyzję odnośnie momentów czasowych, w których dane zadanie może rozpocząć swoje wykonywanie.

- Algorytmy dynamiczne oparte na najlepszym działaniu systemu (ang. dynamic best effort approaches). W przypadku niniejszym możliwość wykonania zadań przed upływem ich ograniczeń czasowych nie jest sprawdzana. System stara się czynić wszystko, co jest w jego mocy, w celu dotrzymania ograniczeń czasowych wszystkich zadań. Ponieważ jednak nie są w tym przypadku udzielane żadne gwarancje, dane zadanie może zostać wyłączone w trakcie jego wykonywania.

Należy zwrócić jeszcze uwagę, że pomimo wyróżnienia czterech powyższych klas algorytmów alokacji i szeregowania zadań, istnieją rozwiązania posiadające cechy i stanowiące kombinację kilku rozważonych uprzednio schematów. Poniżej zostaną w sposób skrótowy scharakteryzowane rozważane w tym miejscu poszczególne klasy algorytmów.

5.1. PODEJŚCIE STATYCZNE Z TABLICĄ ZADAŃ

Algorytmy statyczne z tablicą zadań stosowane są do szeregowania zadań, które albo są zadaniami periodycznymi, albo też zostały przetransformowane do zadań periodycznych przy wykorzystaniu opracowanych w tym celu odpowiednich technik. Na podstawie danych charakterystyk zadań tworzona jest tablica, przy użyciu jednej z wiele możliwych technik, np. poszukiwań heurystycznych (ang. search heuristics), w której zostają określone momenty rozpoczęcia wykonywania poszczególnych zadań oraz czasy ich trwania. Podczas pracy systemu zadania są przydzielane zgodnie z powyższą tablicą. Jest to rozwiązanie w wysokim stopniu przewidywalne, ale zarazem bardzo nieelastyczne, gdyż jakakolwiek zmiana zadań, bądź ich charakterystyk, powoduje konieczność ponownego przeliczenia powyższej tablicy.

5.2. PODEJŚCIE STATYCZNE Z WYŁĄSZCZENIEM PRIORYTETOWYM

Algorytmy statyczne z wyłączeniem priorytetowym tradycyjnie były wykorzystywane w systemach nie będących systemami czasu rzeczywistego. W przypadku takich systemów wszystkie zadania posiadają priorytety, które mogą być przydzielane statycznie, bądź dynamicznie w dowolnej chwili czasu, przy czym zawsze wykonywane jest zadanie posiadające aktualnie najwyższy priorytet. Bazując na wartościach priorytetów poszczególnych zadań może zostać dokonane ich wyłączenie (ang. pre-emption). W przypadku, gdy wykonywane jest zadanie o niskim priorytecie i jednocześnie do systemu przybywa pewne zadanie posiadające wyższy priorytet, aktualnie wykonywane zadanie podlega wyłączeniu, a następnie nowo przybyłe zadanie jest wykonywane. Jeżeli priorytety przypisywane są do zadań systematycznie, w taki sposób, że brane są

pod uwagę ich ograniczenia czasowe, wówczas powstały plan uszeregowania zadań może być wykorzystany w systemach czasu rzeczywistego. W pracy [2] wykazano, że w przypadku implementacji algorytmu przydziału priorytetów typu RM (ang. rate monotonic), można wyznaczyć wartości graniczne maksymalnego wykorzystania zasobów systemu i jeżeli tylko nie zostaną one przekroczone, zawsze istnieje taki plan uszeregowania zadań, że wszystkie ograniczenia czasowe występujące w systemie zostaną dochowane.

Z kolei w pracy [22] zaprezentowano algorytm planowania cyklicznego (ang. cyclic scheduling), stosowany w dużych dynamicznych systemach czasu rzeczywistego (ang. large-scale dynamic real-time systems), stanowiący kombinację dwóch poprzednio omówionych klas statycznych algorytmów szeregowania zadań, odpowiednio z tablicą zadań i wyłączeniem priorytetowym. W tym przypadku zadaniom przypisywane są okresy należące do zbioru okresów harmonicznym. W ramach każdego z okresów, zadania są przydzielane zgodnie z tabelą, która podaje kolejność w jakiej zadania będą wykonywane. Podejście to jest bardziej elastyczne od algorytmów statycznych z tablicą zadań, ponieważ czasy rozpoczęcia zadań nie są tutaj określone oraz możliwe jest w tym przypadku określenie *a priori* dopuszczalnych wartości granicznych wykorzystania zasobów systemu, o ile tylko maksymalne wymagania zadań czynione odnośnie zasobów systemu są znane dla każdego z cykli. Jednakże w celu określenia tych wymagań należy poczynić pewne założenia co do najbardziej pesymistycznych przypadków, jakie mogą zajść podczas wykonywania tych zadań. W wielu aktualnie wykorzystywanych aplikacjach, zamiast przyjmowania wartości dla najgorszego przypadku, zaufanie do opracowanej metody szeregowania cyklicznego, jest uzyskiwana dzięki dobrze opracowanym i intensywnie przeprowadzanym symulacjom typowych scenariuszy pracy systemu. Jednakże jak wykazano w pracy [23] podejście takie jest kosztowne i niestety nie jest w pełni odporne na błędy.

5.3. PODEJŚCIE DYNAMICZNE OPARTE NA PLANOWANIU

Główną zaletą algorytmów dynamicznych opartych na planowaniu jest to, iż łączą one główne zalety dynamicznych algorytmów szeregowania z wysokim stopniem determinizmu, zapewnianym przez algorytmy sprawdzające możliwość wykonania zadań w czasie rzeczywistym. W przypadku takich algorytmów natychmiast po przybyciu zadania do systemu, ale przed momentem rozpoczęcia jego wykonywania, podejmowane są wysiłki mające na celu stworzenie planu uszeregowania zadań, uwzględniającego zarówno zadania wcześniej zagwarantowane (ang. guaranteed tasks), jak i zadania, które aktualnie wpłynęło do systemu. W przypadku gdy próba zaszergowania takiego zadania się nie powiodła, a została ona dokonana dostatecznie wcześniej przed upływem jego ograniczenia czasowego, wówczas przeważnie istnieje dostateczny przedział czasu na podjęcie pewnych działań alternatywnych. Rozwiązanie powyższe pozwala na zapewnienie determinizmu odnośnie zachowania się poszczególnych zadań oraz zapewnia kontrolę dostępu zadań do zasobów systemu.

5.4. PODEJŚCIE DYNAMICZNE UKIERUNKOWANE NA JAK NAJLEPSZE DZIAŁANIE SYSTEMU

W przeciwieństwie do omówionego powyżej dynamicznego algorytmu opartego na planowaniu, zastosowanie podejścia opartego jedynie na wywłaszczeniu priorytetowym, w przypadku użycia ograniczeń czasowych jako priorytetów, powoduje, że dane zadanie może zostać wywłaszczone w dowolnym momencie swego wykonywania. W tym przypadku, zanim zostanie przekroczony przydzielony limit czasu lub zanim zostanie zakończone zadanie, bez względu na to co nastąpi najpierw, nie jest wiadomo, czy ograniczenia czasowe systemu będą zachowane. Jest to główny mankament dynamicznych algorytmów opartych na najlepszym działaniu systemu. Jedynie gruntowna analiza zachowania się systemu szeregującego zadania dla najgorszego z możliwych przypadków, pozwala, oczywiście tylko z pewnym stopniem prawdopodobieństwa, na uniknięcie sytuacji, w których ograniczenia czasowe nie mogłyby być dochowane. Ze stosowanymi w praktyce metodami analizy najgorszego przypadku można zapoznać się w pracy [24]. Jednakże zaproponowane tam metody mogą być stosowane jedynie do zadań o bardzo prostych charakterystykach.

6. PRZYKŁADY ALGORYTMÓW SZEREGOWANIA ZADAŃ W SYSTEMACH CZASU RZECZYWISTEGO

Różnorodność metryk stosowanych w celu pomiaru wydajności systemu, podejść stosowanych do szeregowania zadań oraz zasobów systemu, wykorzystywanych przez zadania jest powodem znacznego zróżnicowania stosowanych w praktyce algorytmów szeregowania zadań. Jednakże, w pracach [25] i [26] wykazano, że w przypadku systemów czasu rzeczywistego z ostrymi ograniczeniami czasowymi, dla większości problemów szeregowania zadań, znalezienie rozwiązania dokładnego jest niemożliwe, ze względu na ich zbyt wielką złożoność obliczeniową. Na przykład problem szeregowania zadań o jednostkowych czasach wykonania oraz z arbitralnie ustaloną relacją poprzedzania dla dwóch procesorów i jednego wspólnie używanego zasobu jest problemem NP-pełnym. Algorytmy dostarczające rozwiązania tego problemu w czasie wielomianym istnieją tylko w przypadku, gdy zbiór relacji poprzedzania jest pusty, ale za to dana jest arbitralnie ustalona liczba zasobów. Natomiast dla trzech procesorów i jednego dzielonego zasobu, nawet w przypadku, gdy zbiór relacji poprzedzania jest pusty, problem jest NP-pełny. Uogólnione wersje powyższych problemów są z reguły NP-pełne, chociaż dla ograniczonej liczby przypadków istnieją także rozwiązania wielomianowe. Podsumowując można powiedzieć, że szeregowanie zadań przy ograniczonych zasobach systemu jest problemem NP-pełnym, a obecność relacji poprzedzania powoduje dodatkowe jego skomplikowanie. Jednakże, jak zostanie to pokazane w dalszej części niniejszego rozdziału w wielu praktycznych przypadkach, zastosowanie odpowiednich heurystyk, bądź algorytmów przybliżonych pozwoliło na efektywne szeregowanie zadań o podanych w pełni charakterystykach, dotyczących zarówno wymagań odnośnie zasobów systemu, jak i relacji wzajemnego poprzedzania.

6.1. ALGORYTMY STATYCZNE Z TABLICĄ ZADAŃ

Podejścia bazujące na algorytmach statycznych z tablicą zadań, podyktowane są głównie faktem konieczności wcześniejszej alokacji (ang. preallocation) zasobów, koniecznych dla zachowania ograniczeń czasowych zadań krytycznych, tak aby terminowe zakończenie tych zadań mogło być określone *a priori*. Zadania takie są zwykle szeregowane w sposób statyczny, tak aby ich ograniczenia czasowe były zachowane nawet w warunkach najgorszego przypadku (ang. worst case conditions). Z oczywistych powodów przyjmuje się, że zadania takie są zadaniami periodycznymi. Gdyby tak nie było, zadania takie zawsze można sprowadzić do zadań periodycznych, przyjmując najgorszy przypadek czasu ich ponownego przybycia. Zgodnie z tym, co pokazano w pracy [27], dla zadań periodycznych istnieje uszeregowanie zadań, gwarantujące dotrzymanie wszystkich ograniczeń czasowych, tylko wtedy, gdy można wykazać, że takie uszeregowanie istnieje dla okresu równego najmniejszej wspólnej wielokrotności okresów wszystkich zadań, oznaczanej skrótem LCM (ang. least common multiple). Dla danego zbioru zadań periodycznych, typowy algorytm szeregowania zadań, przewidziany dla systemu wieloprocesorowego lub rozproszonego, próbuje przypisać podzadania danych zadań do odpowiednich procesorów lub węzłów systemu oraz stworzyć plan uszeregowania dla okresu równego najmniejszej wspólnej wielokrotności okresów zadań. W czasie pracy systemu, zbiór zadań jest wykonywany periodycznie zgodnie z powyższym planem ich uszeregowania dla każdego okresu, równego LCM jednostek czasu.

W przypadku, gdy zadania posiadają proste charakterystyki, tablica określająca kolejność ich wykonywania może zostać utworzona dzięki wykorzystaniu techniki oznaczonej skrótem EDF (ang. earliest-deadline-first), opartej na wykonywaniu najpierw tych zadań, które znajdują się najbliżej swych ograniczeń czasowych lub też szeregując jako pierwsze do wykonania te zadania, które posiadają najkrótsze okresy (ang. shortest-period-first technique). Jednakże, oprócz ograniczeń związanych z wielkością okresów ich wykonania, zadania mogą posiadać ograniczenia odnośnie zasobów systemu, a także mogą być na ich zbiorze określone relacje poprzedzania, wyłączości, komunikacji i replikacji. W większości takich przypadków znalezienie planu uszeregowania zadań jest problemem NP-zupełnym, jakkolwiek pracy [28] zaproponowano pewne heurystyki dające jego przybliżone rozwiązanie. Większość algorytmów stosowanych w tej dziedzinie, w celu znalezienia wykonywalnego planu uszeregowania zadań wykorzystuje metodę podziału i ograniczeń (ang. branch-and-bound).

Dla przykładu, w pracy [29] problem szeregowania był badany dla modelu zadań, w którym zadania zostały podzielone na podzadania, na zbiorze których zostały określone relacje wyłączości i poprzedzania. Uzyskane rozwiązania nadawały się do zastosowania w systemach wieloprocesorowych. W przypadku istnienia relacji wyłączości pomiędzy dwoma podzadaniami s_1 i s_2 , wykonywanie zadania s_1 nie może zostać przerwane przez zadanie s_2 i na odwrót. Relacje wyłączości mogą zostać użyte w celu modelowania konfliktów powstałych podczas dostępu do zasobów. W takim przypadku algorytm wykorzystuje technikę podziału i ograniczeń, gdzie początkowy plan uszeregowania zadań, który bazuje na uporządkowaniu zadań zgodnie z ich ograniczeniami

czasowymi, jest następnie w każdym kolejnym kroku modyfikowany, w celu zredukowania maksymalnych opóźnień zadań (ang. tasks lateness). Jeżeli zostanie znaleziony taki plan uszeregowania zadań, dla którego maksymalne opóźnienie (ang. maximum lateness) jest równe zero, bądź jest ujemne, wówczas wszystkie istniejące w systemie ograniczenia czasowe będą zachowane. Jeżeli plan szeregowania zadań, gwarantujący ich wykonanie przed upływem ograniczeń czasowych nie zostanie znaleziony, wtedy algorytm tworzy uszeregowanie zadań, charakteryzuje się najmniejszym maksymalnym opóźnieniem.

Z kolei algorytm opisany w pracy [30] uwzględnia wymagania, poczynione odnośnie komunikacji i replikacji zadań oraz jest on głównie stosowany w przypadku systemów rozproszonych. Działanie powyższego algorytmu polega na grupowaniu podzadań danych zadań, w zależności od liczby koniecznych komunikacji, przeprowadzanych pomiędzy parą współdziałających podzadań oraz czasów wykonania tych podzadań. Podzadania, które zostały zgrupowane (ang. clustered) przypisywane są do tego samego węzła w systemie, dzięki czemu eliminowane są dodatkowe narzuty czasowe związane z ich komunikacją. Wykonalny plan uszeregowania zadań zostaje ustalony poprzez zastosowanie techniki poszukiwań heurystycznych, która bierze pod uwagę różne charakterystyki zadań, a w szczególności czasy wykonania podzadań, narzuty komunikacyjne, ich ograniczenia czasowe oraz zdefiniowane na ich zbiorze relacje poprzedzania. Komunikacje przeprowadzane w systemie pomiędzy podzadaniami podlegają również szeregowaniu. Algorytm taki został zaprojektowany dla zadań, których podzadania muszą być wykonywane przez różne jednostki obliczeniowe systemu, w celu spełnienia wymagań replikacyjnych. Ponadto całkowite wymagania obliczeniowe narzucone na podzadania danych zadań, mogą być takie, że pojedynczy węzeł systemu może nie być w stanie ich wykonać przed upływem czasu równego okresowi zadania. Jakkolwiek, dzięki dystrybucji podzadań, możliwej poprzez wykorzystanie tkwiących w obrębie danego zadania możliwości zrównoleglenia obliczeń, spełnienie wymagań odnośnie okresowości zadania może stać się możliwe. Może także zajść taka sytuacja, w której wszystkie zasoby wymagane poprzez pewną grupę podzadań, mogą nie być dostępne w żadnym innym węźle systemu. W pracy [3] podane są także przykłady algorytmów opartych jedynie na technice podziału i ograniczeń, które mogą znaleźć zastosowanie do szeregowania zadań z ograniczeniami poprzedzania i komunikacji. Jednakże w tym przypadku, w przeciwieństwie do rozwiązania zaproponowanego w pracy [30] wszystkie podzadania danych zadań są szeregowane do wykonania tylko przez jeden węzeł systemu.

Głównym kryterium stosowanym w przypadku statycznego szeregowania zadań periodycznych jest determinizm, polegający na skonstruowaniu takiego planu uszeregowania zadań, aby spełnione były wszystkie wymagania poczynione odnośnie czasów wykonania zadań, ich poprzedzania itp. W przypadku statycznych algorytmów alokacji i szeregowania zadań, momenty czasu oraz jednostki obliczeniowe systemu, które będą wykonywały dane zadania, zostają ściśle określone dla wszystkich możliwych przypadków zadania. Natomiast w przypadku, w którym w systemie występują zarówno zadania periodyczne, jak i nieperiodyczne, korzystnym jest podjęcie na etapie statycznego

szeregow
przed up
czny za
modyfika
szego sz
okresowy

Algo
szości sy
temów n
w zależn
CPU-bou
systemów
zadań og
w sposób
wym zos
skoncent
procesor
tetowym
Monoton
wartości
zadaniom
że ten s
algorytm
łatwy do
skrótom
nym prz
im bliżej
priorytet

Meta
priorytet
nia prog
może by
my bazu
zarówno
do priory
mogą ul
systemu
występow
czasowy

zreduko-
zależiony
maximum
systemie
antujący
y, wtedy
malnym

e odnoś-
zypadku
powaniu
eprowa-
nia tych
ne są do
narzuty
i zostaje
erze pod
podzadań,
zbiore
adaniemi
a zadań,
owe sys-
ania ob-
edynczy
równego
wykorzy-
eń, speł-
że zajęć
podzadań,
są także
re mogą
komuni-
owanego
nia tylko

ia zadań
szerego-
e czasów
alokacji
óre będą
przypad-
o zadania
tycznego

szeregowania, pewnych kroków mających na celu zapewnienie możliwości wykonania, przed upływem ograniczeń czasowych, przybywających do systemu w sposób dynamiczny zadań aperiodycznych. W pracy [31] zaprezentowano algorytm dynamicznej modyfikacji statycznych planów uszeregowania zadań, w celu umożliwienia wydajniejszego szeregowania zadań nieokresowych, z jednoczesnym uwzględnieniem zbioru okresowych zadań krytycznych.

6.2. ALGORYTMY STATYCZNE Z WYWŁASZCZENIEM PRIORYTETOWYM

Algorytmy statyczne z wywłaszczeniem priorytetowym są wykorzystywane w większości systemów z podziałem czasu (ang. time-sharing systems). W przypadku systemów nie będących systemami czasu rzeczywistego priorytety zadań ulegają zmianie, w zależności od tego, czy są one bardziej związane z jednostką przetwarzającą (ang. CPU-bound), czy też z urządzeniami I/O (ang. I/O-bound). Natomiast w przypadku systemów czasu rzeczywistego, przydział priorytetów jest związany z przypisanymi do zadań ograniczeniami czasowymi, przy czym przydział ten może być dokonywany w sposób statyczny lub dynamiczny. Algorytmy oparte na wywłaszczeniu priorytetowym zostały przeanalizowane od strony formalnej w pracy [2]. W pracy tej autorzy skoncentrowali się na problemie szeregowania zadań periodycznych na pojedynczym procesorze, a następnie zaproponowali dwa typy algorytmów z wywłaszczeniem priorytetowym. Pierwszy z rozważanych algorytmów oznaczony skrótem RM (ang. Rate Monotonic) dokonuje przydziału priorytetów do zadań w sposób statyczny, bazując na wartościach ich okresów. Przy czym wyższe wartości priorytetów są przypisywane zadaniom posiadającym krótsze okresy. Ponadto w pracy niniejszej zostało wykazane, że ten sposób przydziału priorytetów jest optymalny pośród wszystkich możliwych algorytmów statycznego przydziału priorytetów. Ponadto sposób ten jest intuicyjnie łatwy do zrozumienia. Drugi z rozważanych w pracy [2] algorytmów, oznaczony skrótem EDF (ang. Earliest-Deadline-First) jest algorytmem bazującym na dynamicznym przydziale priorytetów. W tym przypadku zadanie uzyskuje tym wyższy priorytet, im bliżej swego ograniczenia czasowego się znajduje. Powyższa strategia przydziału priorytetów jest również intuicyjnie jasna.

Metody statycznego przydziału priorytetów wydają się być atrakcyjne, ponieważ priorytety zostają przypisane do zadań tylko jeden raz przed rozpoczęciem wykonywania programu i nie muszą już być zmieniane w jego trakcie. Jednakże algorytm RM może być zastosowany jedynie do szeregowania zadań okresowych. Natomiast algorytmy bazujące na dynamicznym przydziale priorytetów mogą znaleźć zastosowanie zarówno do szeregowania zadań okresowych, jak i aperiodycznych. W przeciwieństwie do priorytetów przypisanych statycznie, wartości priorytetów przypisanych dynamicznie mogą ulec zmianie w trakcie wykonywania programu, w przypadku przybycia do systemu zadania o wcześniejszym ograniczeniu czasowym. Fakt ten jest przyczyną występowania w przypadku dynamicznego przydziału priorytetów większych narzutów czasowych w trakcie pracy systemu (ang. run-time overheads).

Główna zaleta obu opisanych powyżej strategii dynamicznego przydziału priorytetów polega na tym, że dla zadań periodycznych zawsze istnieją pewne wielkości graniczne dopuszczalnego stopnia wykorzystania zasobów systemu. Na przykład w przypadku zastosowania strategii przydziału priorytetów typu RM, zbiór n niezależnych zadań może zostać uszeregowany, w przypadku ich wykonywania na pojedynczym procesorze, w taki sposób, że dla wszystkich zadań będą zawsze spełnione ograniczenia czasowe, jeżeli tylko dla dużych wartości n stopień wykorzystania procesora nie będzie większy niż $\ln 2$. Praca [32] podaje sposób znajdowania większych, a zarazem lepszych wartości powyższej granicy, w przypadku dokładniejszego sprecyzowania strategii monotonicznego przydziału priorytetów. Ponadto jeżeli dla wszystkich zadań ich okresy są harmonikami najmniejszego z okresów, wówczas graniczna dopuszczalna wartość wykorzystania zasobów systemu wynosi $1,00$. Dla kontrastu w przypadku dynamicznego algorytmu EDF, graniczna dopuszczalna wartość wykorzystania zasobów systemu wynosi zawsze $1,00$.

W przypadku algorytmów klasy EDF, w celu ustalania wartości priorytetów zadań może być wykorzystana wartość maksymalnego przedziału czasu, przez jaki dane zadanie może czekać na rozpoczęcie swego wykonywania, z jednoczesną gwarancją nie przekroczenia jego ograniczenia czasowego (ang. task laxity). Algorytm oparty na powyższym podejściu oznaczany jest skrótem LLF (ang. Least-Laxity-First). W ogólności dowolna funkcja, określona na zbiorze parametrów zadań, może zostać użyta w celu przydziału priorytetów.

Algorytmy szeregowania zadań, wykorzystujące strategię RM przydziału priorytetów zostały użyte między innymi podczas misji kosmicznego statku „Apollo”, jednakże praca [2] była pierwszą publikacją, która zawierała formalną charakterystykę i analizę strategii RM. Z dalszym rozwinięciem teorii strategii RM, uwzględniającym dzielone zasoby systemu, występowanie także zadań aperiodycznych oraz zadań o różnych poziomach ważności można zapoznać się w pracy [33].

W ogólnym przypadku analiza konkretnego uszeregowania zadań, a zwłaszcza możliwość sprawdzenia jej realizowalności, w przypadku algorytmów z wywłaszczeniem priorytetowym jest łatwiejsza, w kategoriach teoretycznej optymalności i złożoności obliczeń, jednakże stosowane w tym przypadku metody, na ogół nie biorą pod uwagę kosztów, związanych z przydziałem zadań, bądź też przyjmują, że są one zanedbywane małą stałą. Jednakże w rzeczywistych systemach przydział zadań jest procesem znacznie bardziej skomplikowanym, zwłaszcza jeżeli weźmie się pod uwagę wywłaszczanie priorytetowe zadań, przełączanie kontekstu oraz wznowienie uprzednio wywłaszczonego zadania. Mechanizm przydziału zadań musi również uwzględniać przybyłe do systemu przerwania. Złożoność procesu przydziału zadań w przypadku algorytmów szeregowania zadań bez wywłaszczeń priorytetowych zależy od tego, czy występujące w systemie zdania są od siebie niezależne oraz czy istnieją ograniczenia odnośnie zasobów systemu. Dyskutowane w punkcie kolejnym algorytmy oparte na planowaniu, typowo nie posiadają właściwości wywłaszczania priorytetowego, co częściowo motywowane jest chęcią zredukowania liczby niepotrzebnych wywłaszczeń.

Dzi
w traci
czasowy
cję plan
wszystk
gwaranc
szego p
w syste
zadanie
przekroc
gorithm
najgorsz
sowe, o
wykorzy
zadań o
mencie
związane
jednostk
powiedz
stek ob
zwalają
wego.

W
nych, d
czenia o
wymaga
może w
wyłączeni
jest czas
kich za
Przeszu
planu u
się rozs
się jedn
jest w
zadań.
stanowi
w syste
odpowi
najwyż
Spośród

6.3. ALGORYTMY DYNAMICZNE OPARTE NA PLANOWANIU

Działanie dynamicznych algorytmów szeregowania zadań polega na sprawdzaniu, w trakcie pracy systemu, możliwości wykonania zadań, przed upływem ich ograniczeń czasowych. Terminowe wykonanie zadania zostaje zagwarantowane poprzez konstrukcję planu wykonania zadań, który zapewnia dochowanie ograniczeń czasowych przez wszystkie zadania gwarantowane (ang. *guaranteed tasks*). Podczas udzielania zadaniu gwarancji czynione są liczne założenia odnośnie czasu wykonania zadania dla najgorszego przypadku, potrzebnych zasobów systemu oraz natury mogących pojawić się w systemie błędów. Jeżeli poczynione założenia są spełnione, w momencie, kiedy zadanie zostanie zagwarantowane, można być pewnym, że jednocześnie nie zostanie przekroczone jego ograniczenie czasowe. Algorytm gwarantujący (ang. *guarantee algorithm*) musi brać pod uwagę wiele czynników, takich jak: czas wykonania zadania dla najgorszego przypadku, zapotrzebowanie odnośnie zasobów systemu, ograniczenia czasowe, obecność zadań periodycznych i zadań wywłaszczalnych, relacje poprzedzania, wykorzystywane w przypadku szeregowania grup zadań, wielokrotne poziomy ważności zadań oraz odporność systemu na błędy. W przypadku systemu rozproszonego, w momencie przybycia zadania do jednej z jednostek przetwarzających, program sterujący, związany z tą jednostką, stara się zagwarantować wykonywanie zadania przez tę jednostkę, przed upływem jego ograniczenia czasowego. Jeżeli próba taka się nie powiedzie, program szeregujący stara się następnie ustalić, która z pozostałych jednostek obliczeniowych systemu posiada dostateczną nadwyżkę mocy obliczeniowej, pozwalającej na terminowe zakończenie zadania przed upływem jego ograniczenia czasowego.

W pracy [34] opisano algorytm gwarantujący wykonanie zadań niewywłaszczalnych, dla których znane są momenty przybycia do jednostki obliczeniowej, ich ograniczenia czasowe lub wartości okresów, czasy wykonania dla najgorszego przypadku oraz wymagania poczynione odnośnie zasobów systemu. W powyższym przypadku zadanie może wykorzystywać zasoby w trybie dzielonym (ang. *shared mode*), albo też w trybie wyłącznym (ang. *exclusive mode*) oraz przyjęto, że czas użytkowania zasobów równy jest czasowi wykonania zadania. Pełny plan gwarantujący terminowe wykonanie wszystkich zadań jest konstruowany przy użyciu metody heurystycznej, w następujący sposób. Przeszukiwanie drzewa możliwych rozwiązań (ang. *search tree*) rozpoczynane jest od planu uszeregowania, który jest zbiorem pustym, następnie algorytm szeregujący stara się rozszerzyć plan uszeregowania, przez dodanie jednego zadania więcej, przesuwając się jednocześnie o jeden poziom wyżej w drzewie poszukiwań. Procedura ta powtarzana jest w sposób iteracyjny, aż do momentu utworzenia pełnego planu uszeregowania zadań. W trakcie poszukiwań, funkcja heurystyczna (ang. *heuristic function*) H , która stanowi syntezę wszystkich charakterystyk zadań, mających wpływ na ich szeregowanie w systemie czasu rzeczywistego, jest wykorzystywana, w celu dokonania wyboru odpowiedniej gałęzi w drzewie poszukiwań. Wartość funkcji H jest liczona dla co najwyżej k zadań, które pozostały do uszeregowania na danym poziomie poszukiwań. Spośród tych k zadań, zadanie, któremu przypisana została najmniejsza wartość funkcji

H , zostaje wybrane, w celu dodania do częściowo już utworzonego planu uszeregowania. Jeżeli w trakcie trwania tego procesu, okaże się, że częściowy plan uszeregowania zadań jest niewykonalny, ze względu na przekroczenie ograniczeń czasowych pewnych zadań, wówczas zawsze istnieje możliwość cofnięcia się o kilka poziomów i kontynuowania poszukiwań. W pracy [34] pokazano także, że jeżeli wartość k jest stałą, a w większości zastosowań praktycznych k jest wartością małą w porównaniu do liczności zbioru zadań n , wówczas złożoność obliczeniowa algorytmu poszukiwań jest liniowo proporcjonalna do n . Ponieważ złożoność algorytmu jest proporcjonalna do n , algorytm ten jest programowalny, to znaczy posiada on ustaloną wartość czasu wykonania dla najgorszego przypadku, który można zredukować, ograniczając liczbę rozwinięć funkcji H dozwolonych dla każdego kroku algorytmu. W rozważanej pracy podano również sposób wyboru optymalnej wartości k .

W pracy [35] wykazano, że dynamiczny algorytm szeregujący, który nie zna *a priori* czasów przybycia do systemu zadań, nie jest w stanie zagwarantować optymalnej wydajności systemu. Spośród dynamicznych algorytmów szeregowania zadań, ten algorytm uzany jest za lepszy, który dla danej liczby zbiorów zadań, dla których istnieje dopuszczalny plan wykonania, potrafi znaleźć jak największą liczbę planów uszeregowania zadań, gwarantujących ich wykonanie przed upływem ograniczeń czasowych. W pracach [36] i [11] wykazano w wyniku przeprowadzenia intensywnych badań symulacyjnych algorytmów szeregujących zadania, że ze względu na wydajność obliczeniową systemu, najbardziej korzystne jest użycie heurystyki, która wykorzystuje kombinację informacji na temat ograniczeń czasowych zadań oraz ich wymagań odnośnie zasobów systemu. Algorytm tej klasy znalazł zastosowanie także w opisanym w pracy [37] systemie operacyjnym Spring. Z kolei w pracy [38] zaproponowano nałożenie na algorytm poszukiwań heurystycznych ograniczenia w postaci maksymalnej długości planu uszeregowania zadań (ang. schedule length), uzyskanej dzięki wykonaniu specjalnego algorytmu minimalizującego tę długość.

W pracach [39], [40] i [41] zaproponowano kilka metod postępowania w przypadku dynamicznego szeregowania zadań dla współpracujących ze sobą jednostek obliczeniowych. Poniżej zostaną krótko omówione cztery klasy algorytmów, opisane w pracy [41], różniące się sposobem, w jaki jednostka obliczeniowa traktuje zadania, które nie mogą być zagwarantowane lokalnie. W przypadku algorytmu szeregowania losowego (ang. random-scheduling algorithm), zadanie, dla którego nie został znaleziony dopuszczalny plan uszeregowania zostaje wysłane do innej jednostki przetwarzającej wybranej w sposób czysto losowy. Z kolei w przypadku algorytmów z ukierunkowanym adresowaniem (ang. focused-addressing algorithm), zadanie jest wysyłane do tej jednostki obliczeniowej, co do której można czynić uzasadnione przypuszczenia, że posiada dostateczną nadwyżkę zasobów oraz dysponuje dostatecznie dużą wolną mocą obliczeniową, pozwalającą na ukończenie tego zadania przed upływem jego ograniczenia czasowego. Trzeci typ algorytmu zwany algorytmem opartym na licytacji (ang. bidding algorithm), przed wysłaniem zadania zbiera oferty (ang. bids) pochodzące od wszystkich jednostek systemu, a następnie dane zadanie wysyłane jest do tej jednostki, która zadeklarowała posiadanie największych dostępnych zasobów. Natomiast czwarty typ algorytmu zwany

algorytmu
stosowa
opartych
porówna
cznych.
forma v
algorithm
dowolne
ideal alg
licytacji
poszcze
liczenio
wania ro
znacznie
elastycz
żadnej
w przyp
algorytm
wiek róż
opartego
komunil
wydajno
oraz alg
Zachow
było cał
load) by
dku umi
szeregow
ogranicz
symulac
dla zad
we wsz
wykazał
pozostał
cią. Zag
towane

. Alg
w wielu
każdego

egowa-
owania
wnych
i kon-
t stałą,
niu do
wań jest
a do n ,
rykona-
zwinieć
podano

nie zna
ptymal-
dań, ten
których
planów
i choro-
h badań
ość ob-
rzystuje
ymagań
pisanym
onowano
ymalnej
konaniu

zypadku
liczenio-
acy [41],
nie mogą
go (ang.
szczalny
ej w spo-
owaniem
liczenio-
stateczną
ową, po-
asowego.
gorithm),
jednostek
klarowała
nu zwany

algorytmem elastycznym (ang. flexible algorithm) stanowi połączenie ze sobą technik stosowanych w przypadku algorytmów z ukierunkowanym adresowaniem oraz algorytmów opartych na licytacji. Zaprezentowane w pracy [41] algorytmy poddane zostały, w celu porównania ich skuteczności licznym testom, z uwzględnieniem dwóch przypadków granicznych. Pierwszym przypadkiem granicznym był algorytm, w którym nie istniała żadna forma współpracy pomiędzy jednostkami przetwarzającymi system (ang. noncooperative algorithm), a zadanie nie mogące zostać zagwarantowane lokalnie, wysyłane było do dowolnej innej jednostki. Drugi przypadek graniczny stanowił algorytm doskonały (ang. ideal algorithm), którego działanie było identyczne jak w przypadku algorytmu opartego na licytacji, ale z założeniem zerowych wartości czasów trwania komunikacji pomiędzy poszczególnymi jednostkami przetwarzającymi systemu. Fakt polepszenia wydajności obliczeniowej systemu z ostrymi ograniczeniami czasowymi, w przypadku stosowania szeregowania rozproszonego, został potwierdzony, w każdych warunkach obciążenia systemu, przez znacznie lepsze wartości wyników uzyskanych w przypadku zastosowania algorytmów elastycznych, w porównaniu z pierwszym granicznym typem algorytmu nie zapewniającym żadnej współpracy pomiędzy jednostkami systemu. Wydajność obliczeniowa systemu w przypadku zastosowania algorytmu elastycznego była także lepsza niż w przypadku algorytmów z ukierunkowanym adresowaniem i algorytmów opartych na licytacji. Jakkolwiek różnica w wartości wydajności obliczeniowej systemu w przypadku użycia algorytmu opartego na licytacji oraz algorytmu elastycznego, w przypadku małych wartości opóźnień komunikacyjnych, była zaniedbywanie mała. Sytuacja wyglądała identycznie, odnośnie wydajności systemu, w przypadku porównania algorytmu o ukierunkowanym adresowaniu oraz algorytmu elastycznego, w przypadku dużych wartości opóźnień komunikacyjnych. Zachowanie się algorytmu szeregowania losowego, w porównaniu z algorytmem elastycznym było całkiem dobre, zwłaszcza w przypadkach, w których obciążenie systemu (ang. system load) było niewielkie, albo też było ono znaczne, ale nierównomiernie rozłożone. W przypadku umiarkowanych wartości obciążenia systemu, wydajność obliczeniowa dla algorytmów szeregowania losowego spadała o kilka procent, co w przypadku systemów z ostrymi ograniczeniami czasowymi może być wielkością znaczącą. Omówione powyżej badania symulacyjne wykazały, że żaden z typów algorytmów poszukiwania jednostki obliczeniowej dla zadania, które nie może zostać zagwarantowane lokalnie, nie jest lepszy od pozostałych, we wszystkich możliwych sytuacjach i warunkach pracy systemu. Przeprowadzone testy wykazały, że algorytmy elastyczne zachowują się, w większości przypadków, lepiej niż pozostałe typy algorytmów, jednakże fakt ten jest okupiony ich znacznie większą złożonością. Zagadnienia związane ze stabilnością omówionych typów algorytmów zostały zaprezentowane w pracy [42].

6.4. ALGORYTMY DYNAMICZNE UKIERUNKOWANE NA JAK NAJLEPSZE DZIAŁANIE SYSTEMU

Algorytmy dynamiczne oparte na najlepszym działaniu systemu znalazły zastosowanie w wielu eksploatowanych obecnie systemach czasu rzeczywistego. W systemach takich, dla każdego z zadań obliczana jest wartość priorytetu, bazując na jego charakterystykach.

Następnie system szereguje zadania, wykorzystując w tym celu wartości ich priorytetów. Zaufanie do takich systemów jest osiągane poprzez intensywnie przeprowadzane symulacje ich działania, połączone z rejestracją zadań i wartości przydzielonych im priorytetów.

Często używane w tego typach systemów klasy algorytmów, takie jak EDF (ang. Earliest Deadline First) oraz LLF (ang. Least Laxity First) wykazują optymalne zachowanie, o ile w systemie nie występują przeciążenia (ang. overloads). W pracy [43] zaprezentowano wyniki przeprowadzonych eksperymentów dowodzące, że algorytmy te wykazują znaczną degradację wydajności w przypadku przeciążeń systemu. Ponieważ algorytmy dynamiczne muszą zachowywać się dobrze w różnych warunkach obciążenia systemu, następne zadanie, które zostanie przyjęte do wykonywania, bądź też zostanie odrzucone, musi być wybierane w bardzo ostrożny sposób. Algorytm szeregowania zadań oparty na najlepszym działaniu systemu, opisany w pracy [43], próbuje maksymalizować sumę wartości przypisanych poszczególnym zadaniom, które zostały zakończone w warunkach przeciążenia systemu. Przy czym wartości przypisane zadaniom zależą od czasu zakończenia tych zadań. Zastosowany w rozważanej pracy algorytm, był algorytmem z wyłączeniem priorytetowym. Podczas przeprowadzonych eksperymentów przebadanych zostało wiele różnych typów funkcji przypisujących wartości zadaniom. Między innymi wzięto pod uwagę najkrótszy czas przetwarzania (ang. shortest processing time first), najwcześniejsze ograniczenie czasowe (ang. earliest deadline first), maksymalny dopuszczalny czas oczekiwania zadania na rozpoczęcie wykonywania (ang. least laxity first), zasadę pierwszej kolejności obsługi najwcześniej przybyłego zadania (ang. first come first served), algorytm, który wybiera w sposób losowy następne zadanie do wykonania, jak również algorytm, który stara się przypisać priorytetowi zadania możliwie największą wartość. Oprócz standardowego algorytmu szeregowania zadań o największym priorytecie (ang. highest priority first), zaproponowano również algorytm, który odrzucał zadania, posiadające małą gęstość przypisaną im wartości, tzn. stosunek tej wartości do czasu wykonania zadania, w przypadku dużego prawdopodobieństwa wystąpienia przeciążeń systemu. Zgodnie z oczekiwaniami nowy typ algorytmu znacznie poprawił wydajność systemu w warunkach przeciążeń. Łagodzenie skutków wystąpienia przeciążeń jest w ogólności bardzo złożonym problemem oraz zgodnie z tym co pokazano w pracach [24] i [44] istniejące obecnie rozwiązania tego problemu znajdują się wciąż w stadium początkowym.

Największym mankamentem algorytmów dynamicznych opartych na najlepszym działaniu systemu jest to, iż są one całkowicie nieprzewidywalne oraz suboptymalne. Mówi się, że dynamiczny algorytm szeregowania zadań jest optymalny, jeżeli produkowane przez niego uszeregowanie zadań jest zawsze wykonalne z zachowaniem ograniczeń czasowych, podczas gdy algorytm testowy, np. statyczny algorytm szeregowania posiadający kompletną wiedzę *a priori* na temat charakterystyk tych zadań, też potrafi dostarczyć wykonalnego planu ich uszeregowania. W pracach [45], [46], [47] i [48] wykazano, że dla większości przypadków dopuszczalnych warunków pracy systemów czasu rzeczywistego takie optymalne dynamiczne algorytmy szeregowania zadań nie istnieją. Jakkolwiek w ostatnich latach podjęto wiele prób poprawienia zachowania się

dynamio
mane re
wania p

Prz
paramet
sporady
wielopr
zakończ
nym prz
zaden z
tować s
wykony
wego a
przykła
rezultat
skrótom
pomied
Otrzym
kano w
nych s
dziedzi
zostać
związan
aby mo
przewie

Dw
szczeg
koniecz
podnie
pozosta
szerego
imprec
dokład

W
towały
padku

dynamicznych algorytmów dla najgorszego przypadku warunków pracy systemu. Otrzymane rezultaty mogą stać się pomocne w opracowaniu efektywnej metody rozładowywania przeciążeń systemu.

Przykładowo w pracy [24] dokonano analizy dopuszczalnych wartości granicznych parametrów pracy dynamicznego algorytmu szeregującego z wyłączeniami zadania sporadyczne, zarówno dla systemu jednoprocessorowego, jak i rozproszonego środowiska wieloprocessorowego. W rozważanym modelu, jeżeli zaseregowane zadanie zostało zakończone, zostaje mu przypisana wartość równa czasowi jego wykonania, w przeciwnym przypadku jest mu przypisywana wartość zerowa. W pracy tej zostało wykazane, że żaden z dynamicznych algorytmów szeregowania zadań, nie jest w stanie zagwarantować sumy łącznych wartości przypisanych poszczególnym zadaniom, w trakcie ich wykonywania, większej niż $1/4$ wartości uzyskanej w przypadku zastosowania wzorcowego algorytmu odniesienia, znającego wszystkie charakterystyki zadań *a priori*. Jako przykład wskazujący na nienaruszalność tej wartości granicznej w pracy [24] opisano rezultaty uzyskane w wyniku zastosowania dynamicznego algorytmu oznaczonego skrótem TD₁. W pracy tej uzyskano także pewne wyniki odnośnie relacji jakie zachodzą pomiędzy dopuszczalną wartością przeciążenia systemu, a wartością powyższej granicy. Otrzymane rezultaty uogólniono także dla przypadku dwóch procesorów, gdzie uzyskano wartość graniczną równą $1/2$, która może jednakże zostać przekroczona w pewnych szczególnych przypadkach. Trzeba jednakże podkreślić, że otrzymane w tej dziedzinie rezultaty znajdują się w stadium początkowym, a konieczne prace muszą zostać ukierunkowane na rozwiązanie bardziej interesujących i użytecznych sytuacji związanych z dynamicznym szeregowaniem opartym na najlepszym działaniu systemu, aby mogły znaleźć one zastosowanie w systemach czasu rzeczywistego, które muszą być przewidywalne.

7. INNE ZAGADNIENIA ZWIĄZANE Z SZEREGOWANIEM ZADAŃ

Dwa zagadnienia związane bezpośrednio z problematyką szeregowania zadań są szczególnie ważne w systemach czasu rzeczywistego. Należy do nich odpowiednio: konieczność zapewnienia odporności systemu na ewentualnie powstałe błędy oraz podniesienie wydajności obliczeniowej systemu poprzez wykorzystanie czasu, który pozostał wolny w przypadku wcześniejszego zakończenia zadania. Z problematyką szeregowania zadań związana jest także ściśle dziedzina obliczeń nieprecyzyjnych (ang. imprecise computations) stosowanych w celu skrócenia czasu trwania obliczeń kosztem dokładności dostarczonych wyników.

7.1. SZEREGOWANIE ZAPEWNIAJĄCE ODPORNOŚĆ NA BŁĘDY

W pracy [49] autorzy zaproponowali implementację mechanizmów, które gwarantowałyby dochowanie dla podstawowej wersji zadania ograniczenia czasowego, w przypadku gdy nie wystąpił żaden błąd w systemie. W przypadku powstania takiego błędu

automatycznie byłaby uruchamiana alternatywna, o zmniejszonej dokładności wyników, wersja tego zadania, dla której możliwe byłoby dochowanie ograniczenia czasowego. W przypadku terminowego wykonania podstawowej wersji zadania, wersja alternatywna nie jest już wykonywana, a zaoszczędzony w ten sposób czas jest przekazywany innym zadaniom występującym w systemie. Praca [49] dotyczy tylko zadań periodycznych oraz dopuszcza możliwość wyłączenia każdego z zadań. W pracy tej rozważono także możliwość konstrukcji drzewa, reprezentującego plan szeregowania zadań, przy wykorzystaniu programu szeregującego opartego o tablice zadań.

Powyższa problematyka została rozwinięta w pracy [50], gdzie autorzy skupili się na zagadnieniu szybkiego przełączania pomiędzy pierwotną, a alternatywną wersją zadania, w przypadku wystąpienia błędu. W pracy tej zaproponowano wykorzystanie algorytmu programowania dynamicznego do wyznaczania alternatywnych planów uszeregowania zadań. Następnie wykazano, że wszystkie ostre ograniczenia czasowe zostaną zachowane, pod warunkiem, że liczba powstałych w trakcie wykonywania programu błędów nie przekroczy pewnej ustalonej wartości. Alternatywne plany uszeregowania zadań nie są wykorzystywane w sytuacji bezawaryjnej pracy systemu. Jednakże ich wbudowanie w pierwotne plany uszeregowania zadań powoduje, że zużywają one pewien czas pracy procesora, obniżając tym samym, w pewnym stopniu wydajność obliczeniową systemu. Jest to cena jaką należy zapłacić za bardzo krótki czas wymagany do realizacji przełączenia pomiędzy wersjami zadań. W pracy [50] rozważono także kwestię konieczności rezerwacji pamięci, aby dla każdego z procesorów mógł być przechowywany co najmniej jeden alternatywny plan uszeregowania zadań. W wielu współczesnych systemach czasu rzeczywistego ograniczenia związane z rozmiarami pamięci muszą być wciąż brane pod uwagę. W pracy [50] rozważono także zadania periodyczne, ale dla kontrastu z pracą [49] nie zaproponowano możliwości wykonywania ograniczonych lub mniej dokładnych wersji zadań.

Zaproponowane w pracach [49] i [50] podejścia mające na celu zwiększenie odporności systemu na błędy (ang. fault tolerance) znajdują zastosowanie głównie w statycznych, wbudowanych (ang. embedded) systemach komputerowych czasu rzeczywistego, o bardzo ostrych ograniczeniach czasowych. W takich przypadkach stopień wykorzystania procesora nie jest istotny, bardziej istotne jest zapewnienie realizowalności pierwotnych, bądź alternatywnych planów uszeregowania zadań. Jednakże takie podejścia statyczne nie mogą być stosowane do systemów czasu rzeczywistego nowszych generacji, które muszą być przewidywalne w przypadku reakcji na zachodzące w sposób dynamiczny zmiany środowiska. Ponadto wymagane jest także zastosowanie takich technik, które potrafią połączyć stopień odporności systemu na błędy z dochowaniem ograniczeń czasowych zadań. Posiada to duże znaczenie w przypadku wystąpienia przeciążeń systemu. Dla przykładu możliwe jest połączenie podejścia dynamicznego szeregowania zadań opartego na planowaniu z techniką obliczeń nieprecyzyjnych (ang. imprecise computation).

7.2. SZEREGOWANIE Z ODZYSKIWANIEM ZASOBÓW

Zmienność czasów wykonania zadań może być przyczyną wcześniejszego zakończenia pewnych zadań, niż przewidywał to program szeregujący. Program przydzielają

cy zada
poprzez
niom. T
wania z
bardziej
zmierza
zadań.
nych m
różni s
systemu
niektóre
stosowa
rzystani
procesor
zapropon
w przyp
Praca [4
wykony
zadań, i
nazywa

Z k
procesor
W pracy
algorithm
gwarant
sze nar
z czasem
złożono
mogły z
cie alg
wydajno

W
temu. S
oraz al
Start). W
realizow
siadają
droższy
wykaza
ne, w p
tego. O
ści obl
odpowi

cy zadania (ang. task dispatcher) może zatem podjąć próbę wykorzystania zwolnionego, poprzez wcześniejsze wykonanie zadania czasu, poprzez przydzielenie go innym zadaniom. Takie wolne szczeliny czasowe mogą być na przykład wykorzystane do wykonywania zadań, na które nie zostały nałożone żadne ograniczenia czasowe. Jednakże bardziej cennym z punktu widzenia systemów czasu rzeczywistego jest podejście, które zmierza w kierunku zwiększenia stopnia pewności dotrzymania ograniczeń czasowych zadań. W pracy [51] pokazano, że jeżeli w systemach wieloprocesorowych, pozbawionych możliwości wyłączenia zadań rzeczywisty czas wykonania danego zadania różni się znacznie od czasu przewidzianego dla najgorszego przypadku, w pracy systemu mogą wystąpić różne anomalie, które mogą spowodować naruszenie przez niektóre z zadań ich ograniczeń czasowych. W szczególności nie jest możliwe zastosowanie programu przydzielającego zadania, który zapewniałby stuprocentowe wykorzystanie czasu pracy procesora. Zawsze musi istnieć pewien czas rezerwowy, w którym procesor nie jest wykorzystywany. Aby uniknąć powyższych anomalii w pracy [52] zaproponowano algorytm, który uwzględniał relacje poprzedzania zadań, tak aby w przypadku ich wykonywania równoległego, było zachowane ich uporządkowanie. Praca [52] była motywowana chęcią zagwarantowania, że jeżeli procesor będzie wykonywał alternatywną wersję zadania, to nie zostanie naruszony plan uszeregowania zadań, nawet w przypadku wcześniejszego zakończenia tego zadania. Zagadnienie to nazywane bywa zagadnieniem stabilizacji (ang. stabilisation problem).

Z kolei praca [53] poświęcona została zagadnieniom wykorzystania wolnego czasu procesora w przypadku napływających w sposób dynamiczny do systemu zadań. W pracy [53] pokazano, że algorytmy odzyskiwania zasobów systemu (ang. reclaiming algorithms) muszą być poprawne, to znaczy muszą zapewniać realizowalność zadań gwarantowanych. Następnie algorytmy takie powinny wносить do systemu jak najmniejsze narzuty, w szczególności powinny być one zanedbywane małe w porównaniu z czasem wykonania zadań. Algorytmy takie powinny odznaczać się także ograniczoną złożonością, która powinna być niezależna od liczby zadań, aby koszty z nimi związane mogły zostać wliczone do czasu wykonania zadania dla najgorszego przypadku. Wreszcie algorytmy takie muszą być efektywne, to znaczy muszą one istotnie zwiększać wydajność systemu.

W pracy [53] zaprezentowano dwa typy algorytmów odzyskiwania zasobów systemu. Są to odpowiednio algorytm podstawowy odzyskiwania (ang. Basic Reclaiming) oraz algorytm odzyskiwania z wcześniejszym startem (ang. Reclaiming with Early Start). W algorytmach tych zastosowano strategie polegające na lokalnej optymalizacji realizowalnych wieloprocesorowych planów uszeregowania zadań. Oba algorytmy posiadają ograniczoną złożoność czasową, ale algorytm z wcześniejszym startem jest droższy czasowo od algorytmu podstawowego. Przeprowadzone licznie symulacje wykazały, że zastosowanie prostych lokalnych optymalizacji może być bardzo efektywne, w przypadku podniesienia wydajności dynamicznych systemów czasu rzeczywistego. Odzyskiwanie zasobów jest zatem skuteczną metodą kompensacji utraty wydajności obliczeniowej systemu, spowodowanej przyjęciem czasów wykonywania zadań, odpowiadających najgorszemu z możliwych przypadków.

7.3. OBLICZENIA NIEPRECYZYJNE

W systemie czasu rzeczywistego z silnymi ograniczeniami czasowymi wiele zadań obliczeniowych jest krytycznych ze względu na czas obliczeń. Poprzez zadanie rozumie się podstawową jednostkę pracy, taką jak dokonanie pewnych obliczeń, jednostkę transmisji danych, przesłanie zawartości pliku itp. Zadania krytyczne ze względu na czas posiadają ograniczenia czasowe. Ograniczenia czasowe zadania są podawane typowo w postaci przedziału czasowego. Przedział ten rozpoczyna się czasem gotowości zadania, przy czym zadanie może rozpocząć swoje wykonanie w tym, bądź w dowolnym późniejszym momencie. Z kolei przedział ten kończy się punktem krytycznym zadania, przed upływem tego czasu zadanie musi zostać zakończone i musi ono dostarczyć odpowiednich rezultatów. Nie dostarczenie przez zadanie rezultatów na czas traktowane jest jako powstanie błędu czasowego (ang. timing fault). System czasu rzeczywistego pracuje poprawnie tylko w przypadku, gdy nie występują błędy czasowe. Niestety, jak pokazano w pracy [54] błędy czasowe powstają zawsze w przypadku wystąpienia przeciążenia systemu, a przeciążenia systemu są zwykle nie do uniknięcia.

Sposobem na uniknięcie powstania błędów czasowych podczas przejściowych przeciążeń systemu jest użycie techniki obliczeń nieprecyzyjnych (ang. imprecise computation technique). System bazujący na tej technice nazywany jest systemem nieprecyzyjnym (ang. imprecise system). W systemie nieprecyzyjnym każde zadanie krytyczne ze względu na czas lub zbiór takich zadań posiada część opcjonalną (ang. optional part). W normalnych warunkach operacyjnych, część opcjonalna wykonywana jest w całości a wynik dostarczony przez zadanie posiada wymaganą jakość i nazywa się go precyzyjnym. Podczas przeciążeń systemu, część opcjonalna lub pewna jej część może zostać pominięta (czyli obliczenia z nią związane pozostaną nie zakończone) w celu zachowania zasobów systemu. W przypadku pominięcia części opcjonalnej zadanie dostarcza wynik nieprecyzyjny, (to znaczy przybliżony). Jako przykłady zastosowań, w których korzystniej jest dysponować wynikami nieprecyzyjnymi niż dokładnymi, lecz spóźnionymi są przetwarzanie obrazów oraz śledzenie obiektów. Często jest znacznie lepiej dysponować jedynie fragmentami nieostrych obrazów, czy też estymatami lokalizacji śledzonego obiektu (celu), dostarczonymi w wymaganym przedziale czasu, niż obrazami o doskonałej jakości i dokładnymi danymi lokalizacyjnymi dostarczonymi zbyt późno.

Technika obliczeń nieprecyzyjnych może także zostać w naturalny sposób użyta do zwiększenia odporności systemu na powstałe błędy (ang. fault tolerance) i zapewnienia łagodnej degradacji (ang. graceful degradation) jakości pracy systemu w przypadku wystąpienia błędów. W tym miejscu brane są pod uwagę jedynie błędy i awarie sprzętowe (ang. hardware faults and failures), jak również przejściowe błędy oprogramowania (ang. transient software faults), które zanikają po ponownym doprowadzeniu systemu do stanu wolnego od błędów (ang. fault-free state). Stałe błędy oprogramowania nie są tutaj brane pod rozwagę. W celu pokazania w jaki sposób odporność na powstanie błędów może zostać zwiększona w systemie nieprecyzyjnym rozważone zostaną dla przykładu systemy śledzące i sterujące. Wystąpienie przemijającego błędu może spowo-

dować, że obliczenia związane ze śledzeniem obiektu zostaną zakończone zbyt wcześnie, a dostarczony wynik będzie nieprecyzyjny. Żadna dodatkowa akcja mająca na celu ponowne uruchomienie systemu po wystąpieniu błędu nie jest konieczna, jeśli otrzymany wynik pozwala systemowi utrzymać ślad śledzonego obiektu. Podobnie tak długo jak przybliżone rezultaty, dostarczone przez procedury sterujące są dostatecznie dokładne, aby zapewnić stabilność sterowanego systemu, wystąpienie błędu, w wyniku którego obliczenia kończone są zbyt wcześnie może być tolerowane. W systemach wbudowanych krytycznych czasowo nie ma często czasu na podjęcie akcji ponownego uruchomienia systemu (ang. recovery action), a ograniczone zasoby systemu nie pozwalają na pełną replikację składowanych danych i wykonywanych zadań. Dostarczenie użytecznych wyników przybliżonych, w obecności wystąpienia błędów jest sposobem na zwiększenie dostępności danych i usług systemu, zredukowanie ilości wystąpień sytuacji awaryjnych (ang. error-recovery operations), uproszczenie operacji ponownego uruchomienia systemu po wystąpieniu błędu i zmniejszenie kosztów związanych z replikacją zadań.

Metoda obliczeń nieprecyzyjnych opiera się na zasadzie, w myśl której w warunkach silnego przeciążenia systemu, gdy ograniczenia czasowe dla poszczególnych zadań nie mogą być dotrzymane, lepiej jest dysponować wynikami niedokładnymi, ale dostarczonymi na czas, niż dokładnymi, lecz spóźnionymi. W przypadku zastosowania metody obliczeń nieprecyzyjnych, każde zadanie składa się z części obowiązkowej dostarczającej wyniki przybliżone oraz części opcjonalnej dostarczającej wyniki dokładne. Jeśli przedział czasu przewidziany na wykonanie danego zadania jest odpowiednio duży, wówczas wykonywana jest część opcjonalna zadania i uzyskane wyniki są precyzyjne. W przypadku przeciwnym wykonywana jest tylko część obowiązkowa i trzeba zadowolić się wynikami przybliżonymi. W obliczeniach nieprecyzyjnych implementuje się często metodę zadań monotonicznych. Dane zadanie może być uważane za monotoniczne, gdy dokładność produkowanych przez nie wyników nie ulega pogorszeniu w trakcie jego liczenia, w miarę upływu czasu. Zadanie monotoniczne dzielone jest na część obowiązkową i opcjonalną. Po policzeniu części obowiązkowej w ramach zadanego limitu czasu mogą być kontynuowane obliczenia dla części opcjonalnej, zwiększającej dokładność obliczeń. Wyniki produkowane przez zadanie monotoniczne są wartościowe tylko pod warunkiem wcześniejszego zakończenia części obowiązkowej zadania. Po wykonaniu części obowiązkowej zadania monotonicznego program zarządzający może podjąć decyzję o jego zakończeniu w dowolnej chwili. Decyzja jaka część zadania opcjonalnego będzie wykonana bywa podejmowana zwykle przez system w sposób dynamiczny, w zależności od jego aktualnego obciążenia. Alternatywna metoda implementowana w obliczeniach nieprecyzyjnych polega na tworzeniu wielokrotnych wersji tego samego zadania. W metodzie tej konstruowana jest jedna wersja zadania dostarczająca wyniki dokładne, jednak czas potrzebny na wykonanie koniecznych obliczeń jest odpowiednio długi. Pozostałe wersje tego zadania dostarczają wyniki o coraz mniejszej dokładności, ale za to czas potrzebny na ich uzyskanie jest coraz krótszy. W warunkach krytycznego przeciążenia systemu podejmowana jest w sposób dynamiczny decyzja, która z wersji tego samego zadania powinna być wybrana, aby ograniczenia czasowe były dochowane.

8. DOWODZENIE SZEREGOWALNOŚCI ZBIORU ZADAŃ

Dowolne zadanie periodyczne τ_i może zostać scharakteryzowane przez czas wykonania dla najgorszego przypadku C_i (ang. worst case computation time) oraz przez swój okres T_i . Przyjmuje się, że każde z zadań okresowych musi zostać zakończone przed upływem jego okresu. Przyjmuje się, że zadania są niezależne, jeżeli nie wymagają one wzajemnej synchronizacji. W typowym systemie czasu rzeczywistego występują zarówno zadania periodyczne, jak i nieokresowe. Poprzez użycie prostej procedury odpytywania (ang. polling procedure) lub innej bardziej zaawansowanej techniki, jak np. opisanego w pracy [55] sporadycznego serwera (ang. sporadic server), szeregowanie zadań aperiodycznych może być realizowane zgodnie z algorytmem RM.

W pracy [2] wykazano prawdziwość twierdzenia, mówiącego, że zbiór niezależnych zadań okresowych, które zostały uszeregowane według algorytmu GRMS (Generalised Rate Monotonic Scheduling), zawsze spełniał będzie ograniczenia czasowe, dla wszystkich możliwych czasów rozpoczęcia zadań, jeżeli tylko spełniona będzie następująca nierówność:

$$\frac{C_1}{T_1} + \frac{C_2}{T_2} + \dots + \frac{C_n}{T_n} \leq n \left(2^{\frac{1}{n}} - 1 \right) \quad (1)$$

Gdzie C_i jest czasem wykonania zadania dla najgorszego przypadku, natomiast T_i jest okresem wywołania zadania τ_i . Graniczna wartość dopuszczalnego wykorzystania systemu $n(2^{\frac{1}{n}} - 1)$, dla dużych wartości n szybko zmierza do $\ln 2 = 0,69$. Wartość graniczna twierdzenia (1) jest bardzo pesymistyczna, ponieważ jest rzeczą niezwykle rzadko spotykaną w praktyce, aby dla wszystkich zadań ich rzeczywiste czasy wykonania były równe czasom odpowiadającym najgorszemu z możliwych przypadków. W pracy [56] pokazano, że w realizowanych w praktyce systemach średnie wykorzystanie zasobów wynosi około 88%. Tak więc wolne zasoby systemu mogą być wykorzystywane przez operujące w tle zadania o niskich priorytetach.

9. BIBLIOGRAFIA

1. J.Y. Chung, J.W. Liu, K.J. Lin: *Scheduling periodic jobs that allow imprecise results*. IEEE Transactions on Computers. Parallel Distributed Systems, vol. 1, no. 2, Apr. 1990, pp. 152–159
2. C.L. Liu, J.W. Layland: *Scheduling algorithms for multiprogramming in a hard real-time environment*. J. Assoc. Comput. Mach., vol. 20, no. 1, Jan. 1973, pp. 46–61
3. D. Peng, K.G. Shin: *Static allocation of periodic tasks with precedence constraints in distributed real-time systems*. Proc. 9th Int. Conf. On Distributed Computer Systems, June 1989, pp. 190–198
4. S.H. Bokhari: *A shortest tree algorithm for optimal assignments across space and time in a distributed processor system*. IEEE Trans. Software Eng., vol. 7, no. 6, 1981, pp. 583–589
5. K. Ramamritham: *Allocation and scheduling of complex periodic tasks*. Proc. Int. Conf. On Distributed Computing Systems, May 1990, pp. 108–115
6. J.P.C. Verhoosel, E.J. Luit, D.K. Hammer: *A static scheduling algorithm for distributed hard real-time systems*. Real-Time Syst., vol. 3, no.3, Sept. 1991, pp. 227–246

7. H. M. and d
8. K.W. made
9. A.K. On Co
10. W. Z. constr
11. W. Z. hard n
12. D.L. system
13. C.J. H. distrib
14. C.J. H. Int. Co
15. K. R. a requir
16. K.G. S. IEEE T
17. J. B. l. a determ
18. E.G. C.
19. R.L. C. determ
20. E. L. a the Arr
21. K. L. e Discret
22. G.D. C. 27, no.
23. C.D. L. executi
24. S. B. a F. Wa 1991
25. M.R. C. constr
26. M.R. C. leteness
27. E.L. L. mation
28. D.W. L. Eng., vo
29. J. Xu, relation
30. K. R. a n Comput
31. K. R. a n periodi

7. H. Mitra, P. Ramanathan: *A genetic approach for scheduling non-preemptive tasks with precedence and deadline constraints*. Proc. Int. Conf. On System Science, Jan. 1993
8. K.W. Tindell, A. Burns, A.J. Wellings: *Allocating hard real-time tasks: An NP-hard problem made easy*. Real-Time Syst., vol. 4, no. 2, June 1992, pp. 145–165
9. A.K. Mok, M.L. Dertouzos: *Multiprocessor scheduling in a hard real-time environment*. Proc. Conv. On Computing Systems, Nov. 1978
10. W. Zhao, K. Ramamritham, J.A. Stankovic: *Pre-emptive scheduling under time and resource constraints*. IEEE Trans. Comput., vol. C-36, no. 8, Aug. 1987, pp. 949–960
11. W. Zhao, K. Ramamritham, J.A. Stankovic: *Scheduling tasks with resource requirements in hard real-time system*. IEEE Trans. Software Eng., vol. SE-13, no. 5, May 1987, pp. 564–577
12. D.L. Eager, E.D. Lazowska, J. Zahorjan: *Adaptive load sharing in homogeneous distributed systems*. IEEE Tran. Software Ent., vol. SE-12, no. 5, May 1986, pp. 662–675
13. C.J. Hou, K.G. Shin: *Load sharing with consideration of future task arrivals in heterogeneous distributed real-time systems*. Proc. Real-Time Systems Symp., Dec. 1991, pp. 94–103
14. C.J. Hou, K.G. Shin: *Incorporation of optimal time-outs into distributed real-time load sharing*. Proc. Int. Conf. On System Science, Jan. 1993, pp. 603–612
15. K. Ramamritham, J.A. Stankovic: *Distributed scheduling of tasks with deadlines and resource requirements*. IEEE Trans. Comput., vol. 38, no. 8, Aug. 1989, pp. 1110–1123
16. K.G. Shin, Y.C. Chang: *Load sharing in distributed real-time systems with state-change broadcasts*. IEEE Trans. Comput., vol. 38, no. 8, Aug. 1989, pp. 1124–1142
17. J. Blazewicz, W. Cellary, R. Slowinski, J. Weglarz: *Scheduling under resource constraints-deterministic models*. Annals of Operations Research, Baltzer AG Scientific Pub. Company, 1986
18. E.G. Coffman: *Computer and Job/Shop Scheduling Theory*. New York, Wiley, 1976
19. R.L. Graham, E.L. Lawl, J.K. Lenstra, A.H.G.R. Kan: *Optimisation and approximation in deterministic sequencing and scheduling: A survey*. Annals Discrete Math., vol. 5, 1979
20. E. Lawler: *Recent results in theory of machine scheduling*. Mathematical Programming: The State of the Art. New York, Springer-Verlag, 1983
21. K. Lenstra, A.H.G.R. Kan, P. Bruchker: *Complexity of machine scheduling problems*. Annals of Discrete Mathematics, New York, North-Holland, 1977
22. G.D. Carlow: *Architecture of the Space Shuttle primary avionics software system*. Commun. ACM, vol. 27, no. 9, Sept. 1984
23. C.D. Locke: *Software architecture for hard real-time applications: Cyclic executive vs. Fixed priority executives*. Real-Time Systems, vol. 4, no. 1, March 1992
24. S. Baruah, G. Koren, D. Mao, B. Mishra, A. Raghunathan, L. Rosier, D. Shasha, F. Wang: *On the competitiveness of on-line real-time scheduling*. Proc. Real-Time System Symp. Dec. 1991
25. M.R. Garey, D.S. Johnson: *Complexity results for multiprocessor scheduling under resource constraints*. SIAM J. Comput., vol. 4, 1975
26. M.R. Garey, D.S. Johnson: *Computers and Intractability: A Guide to the Theory of NP-Completeness*. San Francisco, CA, Freeman, 1979
27. E.L. Lawler, C.U. Martel: *Scheduling periodically occurring tasks on multiple processors*. Information Processing Lett., vol. 12, no. 1, Feb. 1981
28. D.W. Leinbaugh: *Guaranteed response time in a hard real-time environment*. IEEE Trans. Software Eng., vol. SE-6, Jan. 1980
29. J. Xu, L. Parnas: *Scheduling processes with release times, deadlines, precedence, and exclusion relations*. IEEE Trans. Software Eng., March. 1990, pp. 360–369
30. K. Ramamritham: *Allocation and scheduling of complex periodic tasks*. 10th Int. Conf. On Distributed Computing Systems, Paris, France, June 1990
31. K. Ramamritham, G. Fohler, J.M. Adan: *Issue in the static allocation and scheduling of complex periodic tasks*. 10th IEEE Workshop on Real-Time Operating Systems and Software, May 1993

32. J.P. Lehoczky, L. Sha, Y. Ding: *The rate monotone scheduling algorithm: exact characterisation and average case behaviour*. Proc. IEEE Real-Time Systems Symp., Dec. 1989, pp. 166–171
33. L. Sha: *Generalized rate-monotonic scheduling theory: A framework for developing real-time systems*. Proc. of the IEEE, vol. 82, no. 1, Jan. 1994, pp. 68–82
34. K. Ramamritham, J.A. Stankovic, P. Shiah: *Efficient Scheduling algorithms for real-time multiprocessor systems*. IEEE Trans. Parallel Distributed Syst., vol. 1, no. 2, Apr. 1990, pp. 184–194
35. M.L. Dertouzos, A.K. Mok: *Multiprocessor on-line scheduling of hard-real-time tasks*. IEEE Trans. Software Eng., vol. 15, no. 12, Dec. 1989
36. W. Zhao, K. Ramamritham: *Simple and integrated heuristic algorithms for scheduling tasks with time and resource constraints*. J. Syst. Software, vol. 7, 1987, pp. 195–205
37. J.A. Stankovic: *The Spring kernel: A new paradigm for hard real-time operating systems*. IEEE Software, vol. 8, no. 3, May 1991, pp. 62–72
38. F. Wang, K. Ramamritham, J.A. Stankovic: *Bounds on the schedule length for some heuristic scheduling algorithm for hard real-time systems*. Proc. Real-Time Systems Symp., 1992
39. B.A. Blake, K. Schwan: *Experimental evaluation of a real-time scheduler for a multiprocessor system*. IEEE Trans. Software Eng., vol. 17, no. 1, Jan. 1991
40. C.J. Hou, K.G. Shin: *Load sharing with considerations of future arrivals in heterogeneous distributed real-time systems*. Proc. Real-Time Systems Symp., Dec. 1991, pp. 94–103
41. K. Ramamritham, J.A. Stankovic, W. Zhao: *Distributed scheduling of tasks with dead-lines and resource requirements*. IEEE Trans. Comput., vol. 38, no. 8, Aug. 1989, pp. 1110–1123
42. J.A. Stankovic: *Stability and distributed scheduling algorithms*. IEEE Trans. Software Eng., vol. SE-11, no. 10, 1985, pp. 1141–1152
43. C.D. Lock: *Best-effort decision making for real-time scheduling*. Carnegie Mellon Univ., Pittsburgh, May 1985
44. S. Biyabani, J.A. Stankovic, K. Ramamritham: *The integration of deadline and critical-ness in hard real-time scheduling*. Proc. Real-Time Systems Symp., Dec. 1988
45. H. Chetto, M. Chetto: *Some results of the earliest deadline scheduling algorithm*. IEEE Trans. Software Eng., vol. 15, no. 10, Oct. 1989
46. M.L. Dertouzos, A.K.L. Mok: *Multiprocessor on-line scheduling of hard-real-time tasks*. IEEE Trans. Software Eng., vol. 15, no. 12, Dec. 1989
47. K. S. Hong, J.Y.T. Leung: *On-line scheduling of real-time tasks*. Proc. Real-Time Systems Symp., Dec. 1988
48. A.K. Mok: *Fundamental design problems of distributed systems for the hard real-time environment*. Cambridge, May 1983
49. A.L. Liestman, R.H. Campbell: *A fault tolerant scheduling problem*. IEEE Trans. Software Eng., vol. SE-12, no. 11, Nov. 1986, pp. 1089–1095
50. K. Ramamritham, J.A. Stankovic, W. Zhao: *Distributed scheduling of tasks with dead-lines and resource requirements*. IEEE Trans. Comput., vol. 38, no. 8, Aug. 1989, pp. 1110–1123
51. R.L. Graham: *Bounds on multiprocessing timing anomalies*. SIAM J. Appl. Math., vol. 17, no. 2, Mar. 1969
52. G.K. Mancher: *Production and stabilisation of real-time tasks schedules*. J. Assoc. Comput. Mach., vol. 14, no. 3, 1967
53. C. Shen, K. Ramamritham, J.A. Stankovic: *Resource reclaiming in multiprocessor real-time systems*. IEEE Trans. Parallel Distributed Syst., vol. 4, no. 4, Apr. 1993, pp. 382–397
54. J.W.S. Liu: *Imprecise computations*. Proc IEEE, vol. 82, no. 1, Jan. 1994
55. H.M.B. Sprunt, L. Sha, J.P. Lehoczky: *Aperiodic task scheduling on hard real-time systems*. Real-Time Syst. J., June 1989
56. J.P. Lehoczky, L. Sha, Y. Ding: *The rate monotonic scheduling algorithm – exact characterisation and average case behaviour*. IEEE Real-Time Systems Symp., Dec. 1998

The

In the pap
scheduling
scheduling
scheduling
algorithm
scheduling

Keywork.

M. GAJER

PROBLEMS OF TASK SCHEDULING AND ALLOCATION IN REAL-TIME SYSTEMS

Summary

The paper summarises the state of real-time field in the areas of scheduling and operating system kernels. In the paper four paradigms underlying the scheduling approaches are presented. These are: static table driven scheduling, static priority preemptive scheduling, dynamic planning-based scheduling, and dynamic best effort scheduling. In the paper some other issues related with the field of task scheduling, like fault tolerant scheduling and imprecise computation technique, are also discussed. There are discussed task scheduling algorithms for both uniprocessor and multiprocessor systems. The condition for independent periodic task schedulability is also given.

Keywords: Real-time systems, task scheduling and allocation, fault tolerance, imprecise computations

opubl
strukt
ność
jściow
dokła
scalon
wyjść
rezys
zwięk
dane
TDA
dobrze
jściow
typov

Słowa

Pr
wyposa
układó
czej sie
oraz de
symula

Zwiększenie dokładności makromodelu scalonych wzmacniaczy mocy małej częstotliwości

JÓZEF STANCLIK

*Politechnika Wrocławska, Instytut Telekomunikacji i Akustyki,
50-370 Wrocław, Wybrzeże Wyspiańskiego 27
e-mail: jstan@zue.ita.pwr.wroc.pl*

*Otrzymano 2001.01.02
Autoryzowano 2001.07.18*

Przedstawiono modyfikację makromodelu scalonych wzmacniaczy mocy małej częstotliwości, opublikowanego w pracy [6], przeznaczonego do stosowania w programie PSpice. Podano ulepszoną strukturę i procedurę identyfikacji elementów makromodelu. Dzięki modyfikacji powiększono dokładność makromodelu i obszar warunków pracy układu scalonego (napięcia zasilające, napięcie wejściowe, rezystancja obciążenia), w którym makromodel zachowuje pełną dokładność. Poprawę dokładności makromodelu uzyskano dzięki uwzględnieniu zależności napięcia nasycenia układu scalonego od prądu wyjściowego. W zmodyfikowanym makromodelu uwzględniono zależność mocy wyjściowej, mocy zasilania i mocy strat, oraz sprawności energetycznej od napięć zasilających, rezystancji obciążenia w pełnym zakresie zmienności napięcia sterującego. Takie wyniki uzyskano bez zwiększenia stopnia złożoności makromodelu. Podstawę identyfikacji makromodelu stanowią typowe dane katalogowe układu scalonego. Podano przykład modelowania scalonego wzmacniacza mocy TDA2030A. Zidentyfikowany model poddano weryfikacji. Dokładność modelu okazała się bardzo dobra w całym obszarze zmienności napięć zasilających, rezystancji obciążenia i napięcia wejściowego. Uzyskane wyniki wskazują, że zmodyfikowany makromodel jest w pełni przydatny do typowych zastosowań praktycznych.

Słowa kluczowe: makromodel, wzmacniacz mocy m.cz., program PSpice.

1. WPROWADZENIE

Programy komputerowe służące do symulacji układów elektronicznych są zwykle wyposażone w biblioteki modeli elementów, w tym również w biblioteki makromodeli układów scalonych. Zaletą stosowania makromodeli jest zmniejszenie rozmiaru zastępczej sieci elektrycznej, co prowadzi do skrócenia czasu i obniżenia kosztów symulacji oraz do zmniejszenia ewentualnych problemów ze zbieżnością procedur numerycznych symulatorów układów elektronicznych.

Szeroko rozpowszechniony program PSpice jest wyposażony w biblioteki modeli analogowych układów scalonych, jednak biblioteki te nie zawierają makromodeli scalonych wzmacniaczy mocy małej częstotliwości. Makromodele takie zostały opracowane i przedstawione w pracy [6]. Zaproponowano tam struktury i algorytmy identyfikacji parametrów i elementów makromodelu o większej dokładności (i złożoności), przeznaczonego do modelowania wzmacniaczy o zasilaniu symetrycznym, oraz makromodelu o mniejszej złożoności (i dokładności), przeznaczonego do modelowania wzmacniaczy o zasilaniu niesymetrycznym.

Podczas użytkowania makromodeli z pracy [6] okazało się, że modelowanie właściwości układu scalonego w szerokim zakresie zmian warunków pracy wzmacniacza jest prawidłowe, za wyjątkiem modelowania poboru mocy ze źródeł zasilających (porównaj także [7]), oraz mocy wyjściowej, mocy strat i sprawności energetycznej. Wyniki symulacyjnych obliczeń mocy są dokładne jeśli wzmacniacz pracuje w warunkach, w których zidentyfikowano makromodel (rezystancja obciążenia, napięcie zasilające). Po zmianie rezystancji obciążenia (np. z 4Ω na 8Ω) błędy obliczeń mocy: wyjściowej, dostarczonej i traconej, oraz sprawności energetycznej, mogą dochodzić do 50% (patrz p. 3). Błędy te są jeszcze większe gdy wzmacniacz pracuje w warunkach przesterowania. Ponadto przebiegi czasowe prądów pobieranych ze źródeł zasilających nie odwzorowują prawidłowo przebiegów prądu wyjściowego po przekroczeniu zakresu liniowej pracy wzmacniacza (patrz rys. 7).

Proponowana w niniejszej pracy modyfikacja makromodelu scalonych wzmacniaczy mocy m.cz. prowadzi do zwiększenia dokładności obliczeń symulacyjnych mocy wyjściowej, mocy pobieranej i mocy strat, oraz sprawności energetycznej badanych wzmacniaczy, a także do poszerzenia zakresu warunków pracy układu scalonego (rezystancja obciążenia, napięcie zasilające, wysterowanie), w którym jest zachowana pełna dokładność makromodelu. Przedstawiono strukturę zmodyfikowanego makromodelu i sposób doboru jego parametrów i elementów na podstawie typowych danych katalogowych. Poprawę dokładności makromodelu, bez zwiększania liczby jego elementów, uzyskano w wyniku uwzględnienia zależności napięcia nasycenia układu scalonego od prądu wyjściowego i przeniesienia źródła próbkującego prąd wyjściowy. Wyniki weryfikacji zmodyfikowanego makromodelu wzmacniacza TDA2030A wskazują, że modyfikacja doprowadziła do poszerzenia obszaru stosowalności makromodelu i zwiększenia dokładności modelowania, a zmodyfikowany makromodel jest w pełni przydatny do typowych zastosowań praktycznych.

Przedstawiona modyfikacja dotyczy głównie makromodelu o większej dokładności (i złożoności), przeznaczonego do modelowania wzmacniaczy o zasilaniu symetrycznym [6]. Makromodel uproszczony z założenia jest mniej dokładny. Mimo to proponowana modyfikacja może również tam znaleźć zastosowanie i zapewnić poprawę dokładności makromodelu bez zwiększania liczby jego elementów.

Napięcie wyjściowe jest ograniczane gdy dioda D_3 lub D_4 przewodzi (rys. 1). Istota modyfikacji makromodelu polega na takim doborze parametrów modeli diod, aby spadek napięcia na diodach przewodzących V_{fwd} zmieniał się podobnie do zmian napięcia nasycenia V_{sat} układu scalonego w funkcji prądu wyjściowego I_o w pełnym zakresie zmienności tego prądu. Podstawą do wyznaczenia parametrów modeli diod jest charakterystyka prądowo-napięciowa $I_{fwd} = f(V_{fwd})$ dana w n punktach, przy czym:

$$V_{fwdi} \approx V_{sati}(I_{oi}) = V_{CC} - \sqrt{2R_{Li}P_{o\max}(R_{Li})},$$

$$I_{fwdi} \approx I_{oi} = \sqrt{\frac{2P_{o\max}(R_{Li})}{R_{Li}}}, \quad 1 < i \leq n \quad (1)$$

W makromodelach proponowanych w pracy [6] napięcie nasycenia przyjęto jako stałe. W obwodach ograniczników napięcia stosowano jednakowe idealne diody półprzewodnikowe o prądzie nasycenia $I_s = 8 \cdot 10^{-16}$ A, a progi ograniczników były ustalane przez napięcia zasilające i siły elektromotoryczne źródeł pomocniczych V_C i V_E (dane zależnością (13) w pracy [6]). Napięcie nasycenia praktycznie nie zależało od prądu wyjściowego układu scalonego. Błąd modelowania tego napięcia mógł przekroczyć 3V w całym zakresie zmienności prądu wyjściowego (porównaj rys. 5).

Błąd modelowania napięcia nasycenia można zmniejszyć o jeden rząd wielkości (do ułamka wolta) w wyniku uzupełnienia modelu diod ogranicznika napięcia o rezystancję szeregową (porównaj rys. 1 w pracy [1]). Wyznaczenie dwóch parametrów modelu diody jest możliwe na podstawie charakterystyki $I_{fwd} = f(V_{fwd})$ danej w dwóch punktach ($n = 2$, $0 < V_{fwd1} < V_{fwd2}$). Z równania diody półprzewodnikowej z rezystancją szeregową, spolaryzowanej w kierunku przewodzenia $I_{fwd} = I_{s3} \exp\left(\frac{V_{fwd} - R_s I_{fwd}}{\phi_T}\right)$ można wyznaczyć rezystancję szeregową R_s i prąd nasycenia I_{s3} :

$$R_s = \frac{(V_{fwd2} - V_{fwd1}) - \phi_T \ln\left(\frac{I_{fwd2}}{I_{fwd1}}\right)}{I_{fwd2} - I_{fwd1}}, \quad (2)$$

$$I_{s3} = \frac{I_{fwd2}}{\exp\left(\frac{V_{fwd2} - R_s I_{fwd2}}{\phi_T}\right)}. \quad (3)$$

Dalsze zmniejszenie błędów modelowania zależności napięcia nasycenia układu scalonego od prądu wyjściowego można uzyskać w wyniku uzupełnienia modelu diod ogranicznika napięcia o współczynnik emisyjności złącza. Równanie diody półprzewodnikowej o współczynniku emisyjności złącza N , z rezystancją szeregową R_s , spolaryzo-

wanej w kierunku przewodzenia $I_{fwd} = I_{s3} \exp\left(\frac{V_{fwd} - R_s I_{fwd}}{N \varphi_T}\right)$ może być podstawą do wyznaczenia parametrów I_s , R_s i N modeli diod, gdy charakterystyka $I_{fwd} = f(V_{fwd})$ jest dana w trzech punktach ($n = 3$, $0 < V_{fwd1} < V_{fwd2} < V_{fwd3}$) [5]. Parametry modelu diody są określone następująco:

— rezystancja szeregową:

$$R_s = \frac{(V_{fwd3} - V_{fwd1}) - \varphi_T \ln\left(\frac{I_{fwd3}}{I_{fwd1}}\right)}{I_{fwd3} - I_{fwd1}}, \quad (4)$$

— współczynnik emisyjności złącza:

$$N = \frac{(V_{fwd3} - V_{fwd1}) \frac{I_{fwd2} - I_{fwd1}}{I_{fwd3} - I_{fwd1}} - (V_{fwd2} - V_{fwd1})}{\varphi_T \left[\frac{I_{fwd2} - I_{fwd1}}{I_{fwd3} - I_{fwd1}} \ln\left(\frac{I_{fwd3}}{I_{fwd1}}\right) - \ln\left(\frac{I_{fwd2}}{I_{fwd1}}\right) \right]}, \quad (5)$$

Prąd nasycenia jest dany zależnością (3).

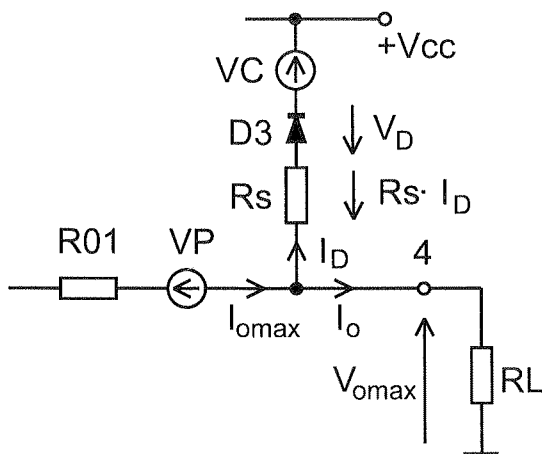
Na podstawie danych katalogowych układu scalonego można zwykle określić większą liczbę punktów charakterystyki $V_{sat} = f(I_o)$. Podstawą do wyznaczania parametrów modeli diod ogranicznika mogą wtedy być uśrednione wartości napięć i prądów. Dobrym kryterium aproksymacyjnym jest kryterium maksymalnego błędu bezwzględ-
nego.

Do uzyskania modeli diod ogranicznika napięcia wyjściowego makromodelu można zastosować program PARTS, służący do identyfikacji modeli elementów elektronicznych i stanowiący uzupełnienie programu PSpice (wchodzący np. w skład pakietu MicroSim Design Center [3]). Program aproksymuje charakterystykę prądowo-napięciową diody, minimalizując błędy aproksymacji. Model wyznaczony za pomocą programu PARTS ma kilka parametrów, z których najważniejsze to prąd nasycenia, rezystancja szeregową i współczynnik emisyjności złącza (IKF i XTI można przyjąć jako wielkości zerowe bez znaczącego wpływu na charakterystykę diody).

Źródła V_C i V_E służą do ustalenia progu ograniczania napięcia wyjściowego. Na rysunku 2 pokazano fragment schematu makromodelu z ogranicznikiem napięcia dodatniego, w którym uwzględniono rezystor R_s . Dioda D_3 przechodzi w stan przewodzenia gdy napięcie wyjściowe osiągnie wartość maksymalną $V_{o,max}$ wynikającą z maksymalnej mocy wyjściowej $P_{o,max}$ na rezystancji obciążenia R_L : $V_{o,max} = \sqrt{2P_{o,max}R_L}$. Płynie przez nią wówczas prąd $I_D = I_{o,max} - V_{o,max}/R_L$. Napięcie nasycenia układu scalonego jest równe: $V_{sat}(I_o) = V_{CC} - V_{o,max}(I_o) = V_C - V_D(I_D) - R_s I_D$. Na tej podstawie siła elektromotoryczna źródła V_C wyraża się zależnością:

$$V_C = V_{CC} - \sqrt{2R_L P_{o\max}(R_L)} + N\phi_T \ln\left(\frac{I_{o\max} - I_o}{I_{s3}}\right) + R_s(I_{o\max} - I_o) \quad (6)$$

gdzie $I_o = \sqrt{2 \frac{P_{o\max}(R_L)}{R_L}}$. W modelu diod ogranicznika o dwu parametrach współczynnik emisyjności złącza we wzorze (6) jest równy jedności. Gdy obciążenie jest symetryczne, siła elektromotoryczna źródła V_E jest równa V_C .



Rys. 2. Fragment schematu makromodelu z ogranicznikiem napięcia dodatniego

Napięcie nasycenia odpowiadające dodatnim prądom wyjściowym może się różnić od napięcia nasycenia dla prądów ujemnych (np. patrz dane układu scalonego wzmacniacza mocy m.c.z. LM3886 [2]). Wtedy w makromodelu trzeba niezależnie od siebie modelować oba napięcia nasycenia: $V_{sat}^+ = f(I_o)$ oraz $V_{sat}^- = f(I_o)$. Podstawą modelowania diod D_3 i D_4 są charakterystyki (odpowiednio):

$$\begin{aligned} V_{fwdi} &\approx V_{sat}^+(I_{oi}), & I_{fwdi} &= I_{oi} & 1 < i \leq n, \\ V_{fwdj} &\approx V_{sat}^-(I_{oj}), & I_{fwdj} &= I_{oj} & 1 < j \leq m, \end{aligned} \quad (7)$$

W przypadku modelowania diod ogranicznika modelem o trzech parametrach (I_s , R_s , N), siły elektromotoryczne źródeł pomocniczych V_C i V_E są określone następująco:

$$V_C = V_{sat}^+(I_o) + R_{s3}(I_{o\max} - I_o) + N_3 V_T \ln\left(\frac{I_{o\max} - I_o}{I_{s3}}\right) \quad (8a)$$

(6)

$$V_E = V_{sat}(I_o) + R_{s4}(I_{o\max} - I_o) + N_4 V_T \ln \left(\frac{I_{o\max} - I_o}{I_{s4}} \right) \quad (8b)$$

(w modelu diod o dwu parametrach $N = 1$).

Przyjęte modele diod ogranicznika zapewniają przybliżenie charakterystyki $V_{sat} = f(I_o)$ układu scalonego z błędami na poziomie ułamka wolta. W wyniku właściwego doboru wartości parametrów modeli diod błędy modelowania zależności napięcia nasycenia układu scalonego od prądu mogą być zmniejszone o jeden rząd.

3. PRZYKŁAD

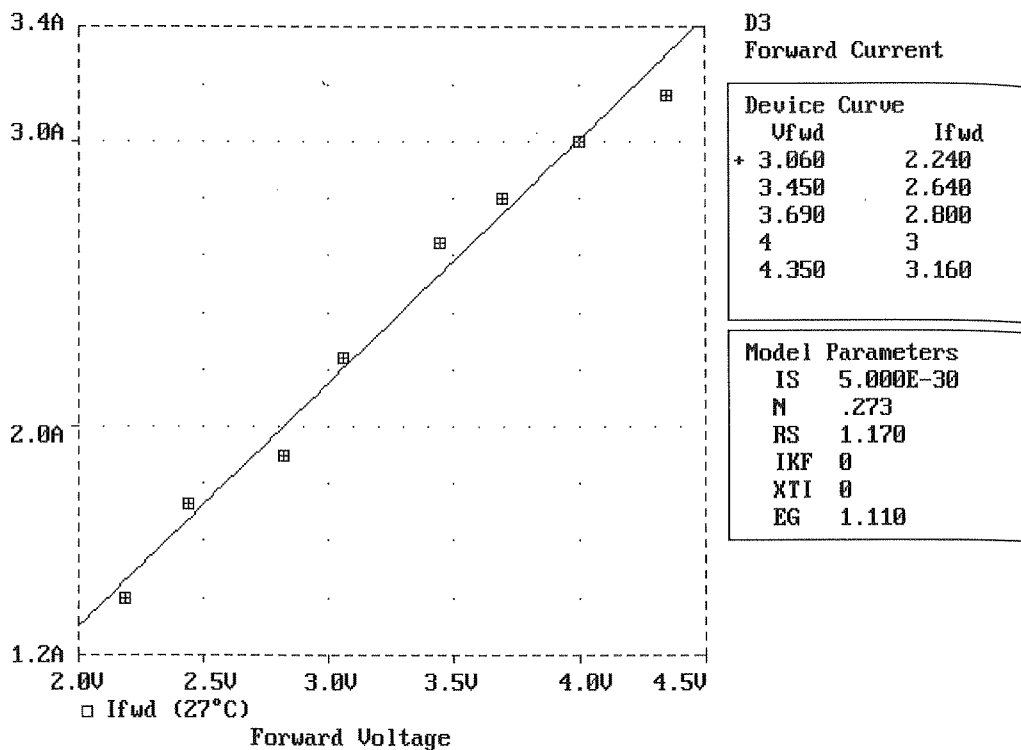
W celu zilustrowania przedstawionej modyfikacji makromodelu scalonych wzmacniaczy mocy m.cz. został zmodyfikowany makromodel układu scalonego TDA2030A. Wartości danych katalogowych tego układu przyjęto identyczne jak w pracy [6], za wyjątkiem napięcia nasycenia układu scalonego. Na ich podstawie obliczono wartości elementów makromodelu identyczne jak w [6], za wyjątkiem diod D_3 i D_4 oraz sił elektromotorycznych źródeł V_C i V_E . Zależność napięcia nasycenia układu scalonego od prądu wyjściowego wyznaczono na podstawie charakterystyk $P_{o\max} = f(V_{CC})$, przy $R_L = 4\Omega$ i $R_L = 8\Omega$ oraz współczynnika zawartości harmonicznych $h = 0,5\%$ (Fig. 3 [4]). Dane te oraz obliczone z zależności (1) wartości napięć i prądów charakterystyki prądowo-napięciowej diod ogranicznika zamieszczono w tabeli 1.

Tabela 1

Zależność napięcia nasycenia układu scalonego TDA2030A od prądu wyjściowego

V_{CC}	V	12	13,5	14	15	16	16,3	17	18
R_L	Ω	4	8	4	4	4	8	4	8
$P_{o\max}$	W	10	8	14	16	18	12	20	14,4
V_{fwd}	V	3,06	2,19	3,45	3,69	4,0	2,44	4,35	2,82
I_{fwd}	A	2,24	1,4	2,64	2,8	3,0	1,73	3,16	1,9

Modele diod ogranicznika napięcia wytworzono za pomocą programu PARTS. Po wprowadzeniu charakterystyki danej w tabeli 1 uzyskano: $I_{s3} = 5 \cdot 10^{-30} \text{ A}$, $N = 0,273$, $R_s = 1,17\Omega$. Na rysunku 3 pokazano charakterystykę prądowo-napięciową diody D_3 oraz parametry jej modelu.



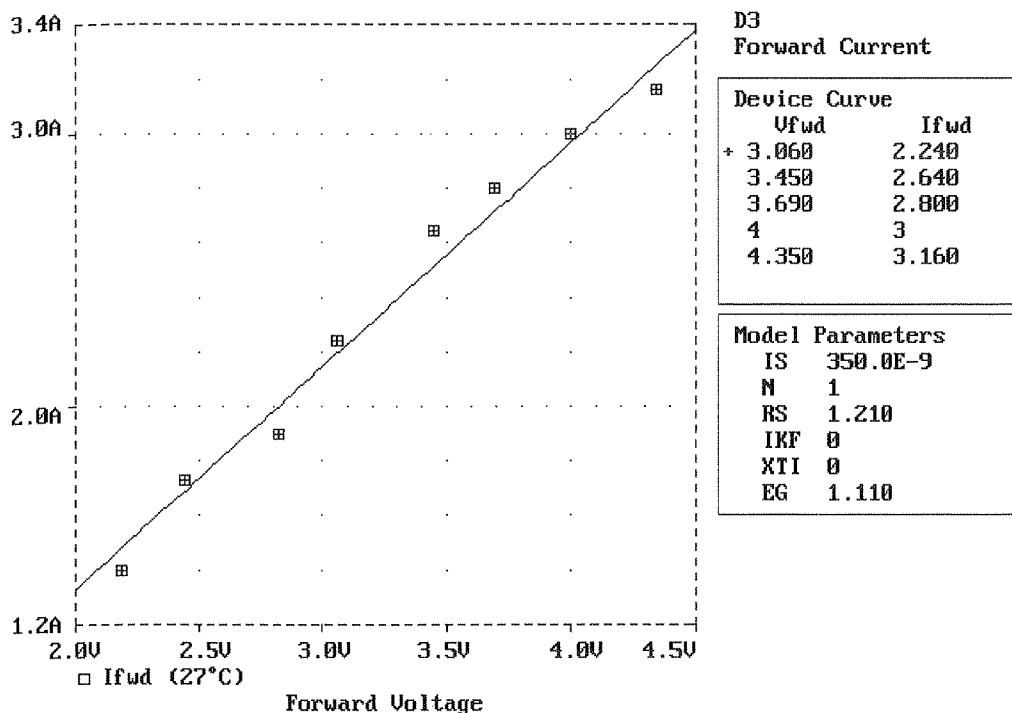
Rys. 3. Charakterystyka prądowo-napięciowa diod ogranicznika napięcia makromodelu układu scalonego TDA2030A (pełny model diody D3)

Do modelowania diod ogranicznika można także zastosować model uproszczony ($N = 1$). Parametry I_{s3} i R_s tego modelu można uzyskać posługując się programem PARTS lub obliczyć z zależności (2) i (3) na podstawie uśrednionych wartości napięć V_{fwd} i prądów I_{fwd} . Dla uśrednionych napięć $V_{fwd1} = 2,25$ V i $V_{fwd2} = 4,1$ V oraz prądów $I_{fwd1} = 1,5$ A i $I_{fwd2} = 3,0$ A uzyskano $I_{s3} = 3,5 \cdot 10^{-7}$ A i $R_s = 1,21$ Ω (rys. 4). Dokładność przybliżenia charakterystyki $V_{sat} = f(I_o)$ jest podobna jak dla pełnego modelu diody.

Oba modele diod dają przybliżenie zależności napięcia nasycenia układu scalonego od prądu wyjściowego z błędem mniejszym od 0,2 V, co przy wielkości napięcia nasycenia 2–5 V można uznać za wynik zadowalający. Wartości napięć V_C i V_E z zależności (6) wynoszą $V_C = V_E = 5,07$ V.

Makromodel wzmacniacza scalonego TDA2030A w formacie bibliotek podukładów liniowych programu PSpice zamieszczono w tabeli 2. Tłustym drukiem zaznaczono zmiany w stosunku do modelu podanego w pracy [6].

Weryfikacja zmodyfikowanego makromodelu polegała na obliczeniu za pomocą programu PSpice jego parametrów elektrycznych i porównaniu wyników z danymi katalogowymi układu scalonego. Wejściowe prądy polaryzacji i niezrównoważenia,



Rys. 4. Charakterystyka prądowo-napięciowa diod ogranicznika napięcia makromodelu układu scalonego TDA2030A (uproszczony model diody D3)

wejściowe napięcie niezrównoważenia, wzmocnienia napięciowe stało- i zmiennoprądowe, bieguny transmitancji, współczynnik SR oraz poziom ograniczania prądu wyjściowego są takie same, jak podano w pracy [6].

Napięcie wyjściowe jest ograniczane na poziomie zależnym od prądu wyjściowego. Na rysunku 5 pokazano zależność napięcia nasycenia zmodyfikowanego makromodelu od prądu wyjściowego. Widać, że napięcie to zmienia się ze zmianami prądu wyjściowego a różnice między odpowiednimi danymi katalogowymi i wartościami obliczonymi dla makromodelu nie przekraczają 0,2 V. Dla porównania na rysunku tym pokazano napięcie nasycenia obliczone w takich samych warunkach z użyciem makromodelu z pracy [6]. Napięcie nasycenia ma wartość 4,76 V i praktycznie nie zależy od prądu. Przy pracy z dużymi rezystancjami obciążenia oznacza to błąd modelowania dochodzący do 3 V co prowadzi do błędnych wyników obliczeń mocy wyjściowej, mocy dostarczonej i mocy strat.

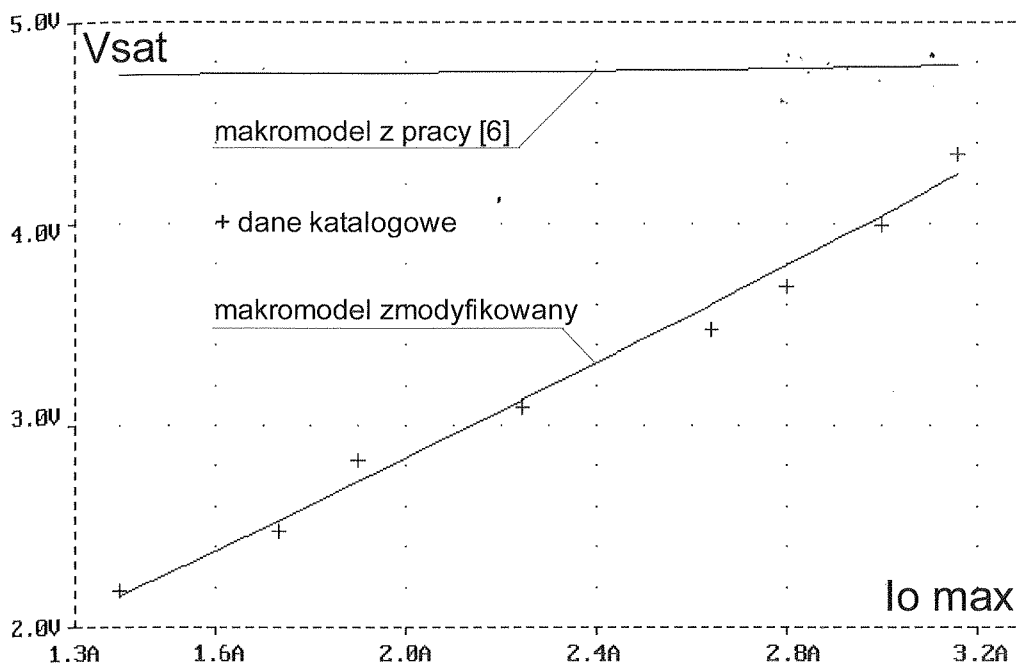
Wyniki obliczeń maksymalnej mocy wyjściowej w funkcji napięć zasilających ($h = 0,5\%$) pokazano na rys. 6. Błędy obliczeniowe mocy wyjściowej makromodelu z pracy [6] przy rezystancji $R_L = 4\Omega$ zwiększają się do 25% w miarę zmniejszania napięć zasilających i do niemal 50% przy $R_L = 8\Omega$. Tylko przy napięciu 18 V i rezystancji 4Ω

Makromodel scalonego wzmacniacza mocy m.cz. TDA2030A

```

*
* TDA2030A audio power amplifier "macromodel" subcircuit
*
* connections:      non-inverting input
*                   |      inverting input
*                   |      |      negative power supply
*                   |      |      |      output
*                   |      |      |      |      positive power supply
*                   |      |      |      |
*
.SUBCKT TDA2030A      1      2      3      4      5
Q1      11      1      13      TDA2030AQ1
Q2      12      2      14      TDA2030AQ2
RC1      5      11      5.5611E+02
RC2      5      12      5.5611E+02
C1      11      12      7.1550E-11
RE1      13      15      3.5794E+02
RE2      14      15      3.5794E+02
IEE      15      3      2.6200E-04
GA      16      0      12      11      1.7982E-03
R2      16      0      1.0000E+05
C2      16      17      3.0000E-11
GB      17      0      16      0      1.7684E+02
RO2      17      0      1.0000E+00
D1      17      18      TDA2030AD1
D2      18      17      TDA2030AD1
EC      18      0      4      0      1.0000E+00
RO1      17      21      3.1907E-01
D3      21      19      TDA2030AD2
VC      5      19      5.0700E+00
D4      20      21      TDA2030AD2
VE      20      3      5.0700E+00
VP      21      4
H      22      0      VP      1.0000E+03
D5      22      23      TDA2030AD1
RP1      23      0      1.0000E+06
D6      24      22      TDA2030AD1
RP2      24      0      1.0000E+06
GP1      5      0      POLY(1) 23      0      5.000E-02      1.000E-03
GP2      0      3      POLY(1) 24      0      5.000E-02      -1.000E-03
.MODEL TDA2030AD1 D (IS=8.00000E-16)
.MODEL TDA2030AD2 D (IS=3.50000E-07 RS=1.210E+00)
.MODEL TDA2030AQ1 NPN (IS=8.00000E-16 BF=622.85700)
.MODEL TDA2030AQ2 NPN (IS=8.64353E-16 BF=688.42100)
.ENDS

```

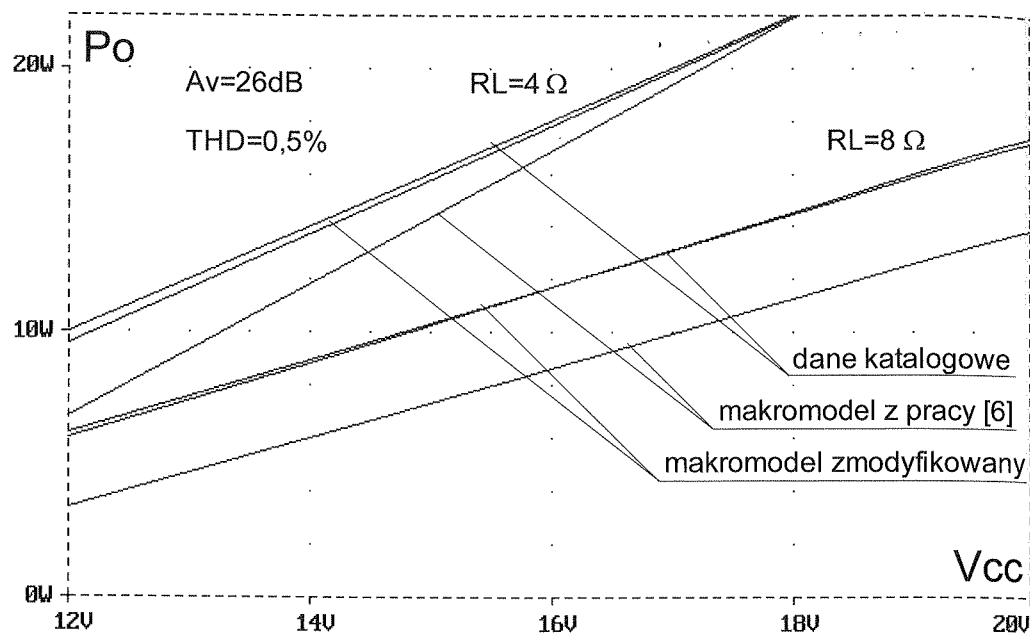


Rys. 5. Zależność napięcia nasycenia makromodelu układu scalonego TDA2030A od prądu wyjściowego

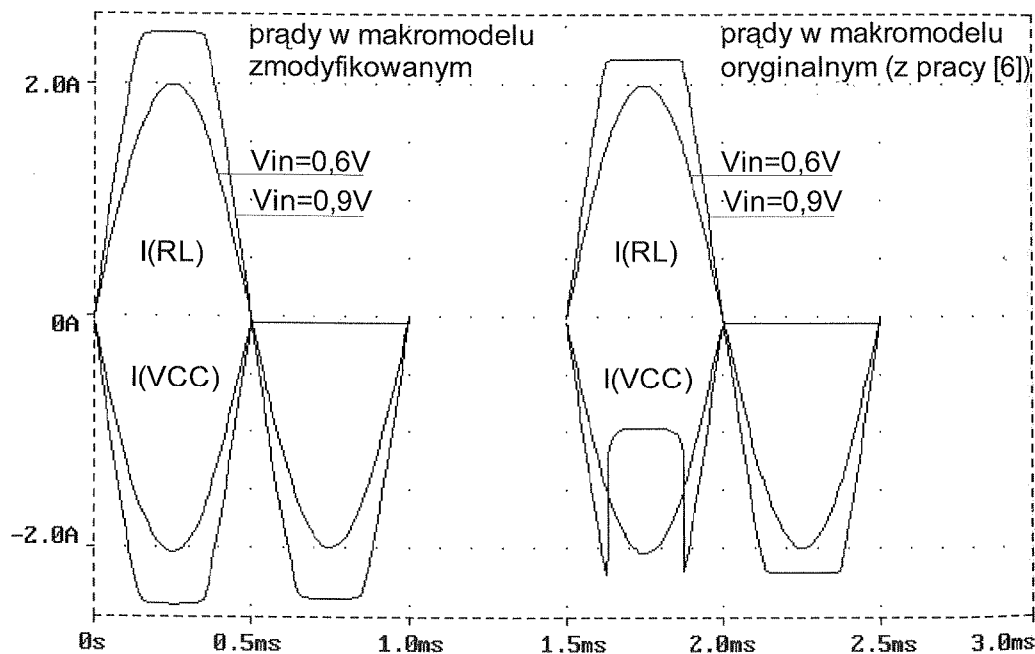
(przyjętych w pracy [6] jako podstawa modelowania) moc wyjściowa jest modelowana praktycznie bezbłędnie. Aby zatem uniknąć błędów modelowania mocy konieczne było stosowanie różnych makromodeli przy różnych rezystancjach obciążenia i różnych napięciach zasilających. W pracy [6], w przykładzie zastosowania makromodelu układu TDA2030A do analizy wzmacniacza elektroakustycznego, aby zapewnić zadowalającą dokładność obliczeń symulacyjnych toru niskotonowego, zmieniono wartości źródeł V_C i V_E makromodelu w stosunku do modeli stosowanych w pozostałych torach. Zgodność mocy wyjściowej, obliczonej z użyciem zmodyfikowanego makromodelu, z mocą katalogową układu scalonego jest w całym zakresie zmienności zasilania i obciążenia bardzo dobra (błędy mniejsze od kilku procent).

Przebiegi czasowe prądu wyjściowego i prądu pobieranego ze źródła napięcia dodatniego pokazano na rys. 7. Przebiegi czasowe prądów pobieranych ze źródeł zasilających obliczone przy użyciu makromodelu z pracy [6] nie odwzorowują prawidłowo przebiegów prądu wyjściowego po przekroczeniu zakresu pracy liniowej. W przypadku makromodelu zmodyfikowanego odwzorowanie jest dokładne (z przesunięciem o prąd spoczynkowy). Przebiegi prądu pobieranego ze źródła napięcia ujemnego mają analogiczne cechy.

Wyniki wykonanych obliczeń potwierdzają uzyskanie bardzo dobrej zgodności właściwości zmodyfikowanego makromodelu z danymi katalogowymi układu scalonego



Rys. 6. Zależność maksymalnej mocy wyjściowej od napięć zasilających we wzmacniaczu z układem scalonym TDA2030A



Rys. 7. Przebiegi czasowe prądów wyjściowych i zasilających w makromodelach układu scalonego TDA2030A

(najwię
zasilają
sterowa
dużą d

Pr
częstot
od prac
od rezy
modeli
Zmody
go w p
prawid
Za
makro
wartoś
jęciow
C
złożon
wzmac
(napię
Wynik
prakty

1. S. K...
- i Te...
2. LM...
- Co...
3. Mic...
4. SGS...
- 1/15...
5. N.C...
- Des...
6. J. S...
- i Te...
7. J. S...
- Ele...

(największe różnice nie przekraczają kilku procent) w całym zakresie zmienności napięć zasilających, rezystancji obciążenia i napięcia wejściowego (łącznie z obszarem przesterowania wzmacniacza). Można zatem uznać, że opracowany makromodel cechuje się dużą dokładnością i nadaje się do większości zastosowań praktycznych.

4. WNIOSKI I UWAGI KOŃCOWE

Przedstawiona modyfikacja makromodelu scalonych wzmacniaczy mocy małej częstotliwości zapewnia uwzględnienie zależności napięcia nasycenia układu scalonego od prądu wyjściowego, a więc zależności mocy wyjściowej, mocy zasilania i mocy strat od rezystancji obciążenia i napięć zasilających. Tej właściwości nie ma żaden z makromodeli analogowych układów scalonych stosowanych dotąd w programie PSpice. Zmodyfikowany makromodel eliminuje ponadto niedoskonałość modelu przedstawionego w pracy [6] polegającą na ograniczeniu zakresu napięć wejściowych, w którym jest prawidłowo modelowany pobór mocy ze źródeł zasilających.

Zaproponowana modyfikacja nie pociąga za sobą zwiększenia liczby elementów makromodelu – polega jedynie na przesunięciu źródła próbkującego V_p i zmianie wartości parametrów modeli diod D_3 i D_4 w obwodach ograniczników napięcia wyjściowego.

Chociaż makromodel przedstawiony w tej pracy cechuje się niewielkim stopniem złożoności, to jednak zapewnia modelowanie podstawowych właściwości scalonych wzmacniaczy mocy z większą dokładnością i w szerszym obszarze warunków pracy (napięcia zasilające, rezystancja obciążenia, wysterowanie), niż modele dotąd znane. Wyniki symulacji komputerowej świadczą o jego przydatności do typowych zastosowań praktycznych.

5. BIBLIOGRAFIA

1. S. Kuta, R. Wydmański: *Dokładny makromodel konweyora prądowego CCII[±]*. Kwartalnik Elektroniki i Telekomunikacji, 1997, T. 43, z. 1, ss. 149–165
2. LM3886 Overture™ High-Performance 68W Audio Power Amplifier w/Mute. National Semiconductor Co., April 1995 (<http://www.national.com>)
3. MicroSim PSpice A/D. Ver.6.3. Reference Manual. MicroSim Corporation, April 1996
4. SGS-THOMPSON Microelectronics, TDA2030A 18W Amplifier and 35W Driver. March 1995, pp. 1/15-15/15 (<http://www.st.com>)
5. N.O. Sokal, J.J. Sierakowski, J.J. Sirota: *Diode model is analysed by computer*. Electronic Design, 1967, Vol. 15, No 14, pp. 80–83
6. J. Stanlik: *Makromodele scalonych wzmacniaczy mocy małej częstotliwości*. Kwartalnik Elektroniki i Telekomunikacji, 1998, T. 44, z. 4, ss. 445–470
7. J. Stanlik: *Power consumption modelling of integrated circuit micropower operational amplifiers*. Electronics Letters, 2000, Vol. 36, No 8, pp. 701–702

J. STANCLIK

ACCURACY INCREASING OF MACROMODEL FOR INTEGRATED CIRCUIT
AUDIO POWER AMPLIFIERS

Summary

A modification to the macromodel [6] for integrated circuit (IC) audio power amplifiers intended for PSpice computer program is presented. Both improved structure of the macromodel and better identification procedure for its components are given. New macromodel is of better accuracy and wider area of operating conditions (supply and input voltages, load resistance) wherein full accuracy is maintained. This results from incorporating the dependence of IC saturation voltage on the output current into the macromodel. Thus, dependencies of the output power, supply power, total power dissipation, and energy efficiency on the supply voltages, load resistance, and the input signal are precisely modelled. These results have been achieved without increasing the macromodel complexity. The new macromodel parameters are found directly from typical IC data sheets. An example of the TDA2030A audio power amplifier modelling is given. It is shown that the modified macromodel accuracy is very good and the macromodel is useful for general purpose applications.

Keywords: macromodel, integrated circuit audio power amplifier, PSpice computer program

Adaptacyjna kompensacja wpływu tłumienia wprowadzanego przez opady deszczu dla systemów satelitarnych

MARIUSZ CZARNECKI

*Instytut Elektroniki, Politechnika Łódzka
ul. Stefanowskiego 18/22, 90-924 Łódź
e-mail: mbczarny@p.lodz.pl*

Otrzymano 2001.06.08

Autoryzowano 2001.07.11

W artykule opisano adaptacyjny system kompensacji tłumienia wprowadzanego przez opady deszczu dla satelitarnych systemów łączności pracujących w pasmie 20-30 GHz wykorzystujący modulację M-PSK do realizacji transmisji ze zmienną szybkością. Zaproponowano strukturę takiego systemu, opracowano metodę komputerowej symulacji działania i przeprowadzono symulacje z wykorzystaniem rzeczywistych danych propagacyjnych zarejestrowanych w czasie trwania eksperymentu z satelitą Olympus. Przeanalizowano wpływ zaników sygnału wynikających z propagacji wielodrogowej na pracę systemu kompensacji. Uzyskano wyniki pozwalające na dobór parametrów poszczególnych bloków systemu i jego optymalizację pod kątem maksymalnej dostępności lub przepustowości.

Słowa kluczowe: telekomunikacja satelitarna, kompensacja tłumienia, propagacja

1. WPROWADZENIE

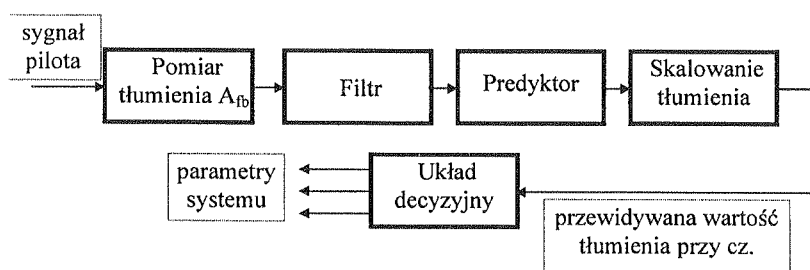
Satelitarne systemy łączności są postrzegane jako jedno z mediów mogących służyć do realizacji nowych usług telekomunikacyjnych wymagających szerokiego pasma. W związku z tym rośnie zainteresowanie wykorzystaniem pasma częstotliwości Ka (20/30 GHz) do szybkiej transmisji danych (np. w systemach multimedialnych). Jednym z problemów związanych z wykorzystaniem wielkich częstotliwości (powyżej 10 GHz) jest znaczne tłumienie wprowadzane w atmosferze przez opady deszczu, które utrudnia osiągnięcie wysokich dostępności systemów przy małym koszcie terminali. Wzrost tłumienia oznacza pogorszenie stosunku sygnału do szumu w punkcie odbioru. Zapewnienie wymaganej stopy błędu jest możliwe przez zwiększenie mocy nadawania (przy czym ograniczeniem są interferencje wspólnokanałowe) albo przez zmniejszenie szyb-

kości transmisji. Projektowanie systemów kompensacji tłumienia jest aktualnym i ważnym problemem badawczym i m.in. było jednym z wiodących tematów programu COST 255 – „Propagation for New SatCom Services in Ku-band and Above”.

W artykule zostanie omówiona budowa i działanie systemu kompensacji tłumienia działającego w oparciu o adaptacyjną modulację PSK. Koncepcja takiego systemu kompensacji (przeznaczonego do zastosowania w przyszłych satelitarnych systemach telekomunikacyjnych) pojawiła się w [6] i została rozwinięta w [8]. W niniejszej pracy zaproponowano strukturę systemu i opracowano algorytmy działania poszczególnych bloków funkcjonalnych (w szczególności układu decyzyjnego) oraz wykonano symulacje komputerowe działania systemu kompensacji mające na celu uzyskanie wskazówek dotyczących doboru parametrów systemu oraz ocenę rzeczywistych możliwości systemu.

2. STRUKTURA SYSTEMU KOMPENSACJI

Na rys. 1 przedstawiono schemat blokowy systemu kompensacji zaników. W modelowym systemie do oceny jakości łącza wykorzystano pomiar mocy sygnału odniesienia – niemodulowanego sygnału o częstotliwości równej częstotliwości nośnej dla łącza „satelita–ziemia” wysyłanego ze stałą mocą przez stację satelitarną, zwanego potocznie sygnałem „pilota”. Sygnał ten jest próbkowany co 1 sekundę. W celu wyeliminowania z sygnału „pilota” szybkich fluktuacji pochodzących m.in. od scyntylacji amplitudy konieczna jest filtracja dolnoprzepustowa sygnału (rys. 1). Użycie filtra powoduje opóźnienie sygnału filtrowanego o kilka sekund, konieczna jest zatem kompensacja tego opóźnienia tak, aby układ wykrywania zaników zdążył zareagować na



Rys. 1. Schemat blokowy systemu kompensacji tłumienia

zmiany tłumienia w atmosferze. Kompensację opóźnienia realizuje blok predyktora. Oprócz kompensacji opóźnienia wprowadzanego przez filtr dolnoprzepustowy predyktor wyznacza wartość tłumienia z wyprzedzeniem o np. 1 sekundę. Ma to na celu skompensowanie opóźnień w komunikacji między stacją naziemną, a stacją satelitarną, które wystąpią przy uzgadnianiu nowych parametrów transmisji.

Sygnał „pilota” ma najczęściej częstotliwość mniejszą niż nośna używana do transmisji sygnału użytecznego ze stacji naziemnej. Oznacza to, że wartość tłumienia mierzona przy częstotliwości „pilota” jest mniejsza niż wartość tłumienia, które system ma za zadanie skompensować. Konieczne jest więc użycie jednej z metod skalowania wartości chwilowej tłumienia dla częstotliwości pilota do wartości dla częstotliwości nośnej stacji naziemnej (ang. instantaneous frequency scaling of attenuation) [7].

Na wyjściu bloku skalowania tłumienia otrzymuje się przewidywaną wartość tłumienia fali nośnej dla transmisji „ziemia–satelita”. Wyznaczona wartość jest podawana na wejście układu decyzyjnego, którego zadaniem jest wyznaczenie parametrów transmisji zapewniających najlepsze, w danych warunkach, wykorzystanie łącza. Parametry te są związane z rodzajem systemu kompensacji. Dla systemu z adaptacyjnym sterowaniem mocą stacji naziemnej parametrem transmisji jest moc fali nośnej, a dla systemu ze zmienną szybkością transmisji jest to zestaw parametrów regulujących szybkość przesyłania informacji: rodzaj modulacji, redundancja kodu nadmiarowego czy liczba zajmowanych szczelin w TDMA.

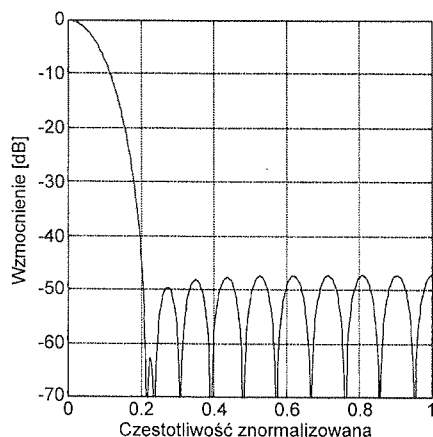
W dalszej części omówiono sposób działania poszczególnych bloków systemu. Opracowano kompletny model systemu, który posłużył do doboru wartości parametrów bloków systemu poprzez symulacje komputerowe. W symulacjach systemu wykorzystano dane propagacyjne zarejestrowane dla rzeczywistego łącza satelitarne (satelita Europejskiej Agencji Kosmicznej Olympus) dla częstotliwości 12.5, 20 i 30 GHz. Kryterium służącym za miarę jakości systemu była dostępność zdefiniowana jako procent czasu, w którym jest spełnione kryterium jakości transmisji wyrażone poprzez minimalną wartość stosunku energii sygnału nośnej przypadającej na jeden bit do widmowej gęstości mocy szumów – E_b/N_0 .

2.1. POMIAR TLUMIENIA

Pomiar tłumienia sygnału odniesienia jest wykonywany co 1 sekundę. Taka częstotliwość jest wystarczająca do zaobserwowania zmian tłumienia związanych z opadami deszczu. Na podstawie zmierzonego chwilowego poziomu mocy sygnału system oblicza wartość tłumienia przekraczającą wartość tłumienia wprowadzanego przez atmosferę w warunkach bezdeszczowych.

2.2. FILTR DOLNOPRZEPUSTOWY

Sygnał niosący informację o tłumieniu wprowadzanym przez opady deszczu zajmuje pasmo częstotliwości do około 0.02 Hz [11]. Do wyodrębnienia tej składowej sygnału zastosowano cyfrowy filtr dolnoprzepustowy o skończonej odpowiedzi impulsowej (SOI). Dobór rzędu filtru wynika z koniecznego kompromisu między skutecznością filtracji składowych szybkozmiennych sygnału pilota, a możliwością kompensacji opóźnienia grupowego filtru. Strukturę SOI wybrano ze względu na stałe, w funkcji częstotliwości, opóźnienie grupowe.



Rys. 2. Charakterystyka amplitudowa filtra służącego do separacji składowej związanej z opadami deszczu

W wyniku przeprowadzonych symulacji systemu stwierdzono, że optymalny, ze względu na dostępność systemu, rząd filtra wynosi $N = 20 \div 30$. Ze względu na stromość charakterystyki przejściowej filtra wybrano w realizacji filtr zaprojektowany przy użyciu okna Czebyszewa, z parametrem określającym minimalny poziom tłumienia w pasmie zaporowym równym 45 dB. Charakterystykę amplitudową filtra pokazano na rys. 2.

2.3. PREDYKCJA SYGNAŁU

Zasada działania zaproponowanego predyktora opiera się na wykorzystaniu szybkości narastania tłumienia. Dla każdej chwili t_i predyktor wyznacza szybkość narastania tłumienia ΔA jako różnicę aktualnej i poprzedniej wartości tłumienia $\Delta A = A(t_i) - A(t_{i-1})$ i zakładając, że różnica ta nie zmienia się przez czas $d_g + 1$ sekund, dokonuje ekstrapolacji wartości tłumienia do chwili $t_i + d_g + 1$. Sygnał A_p na wyjściu predyktora można opisać równaniem:

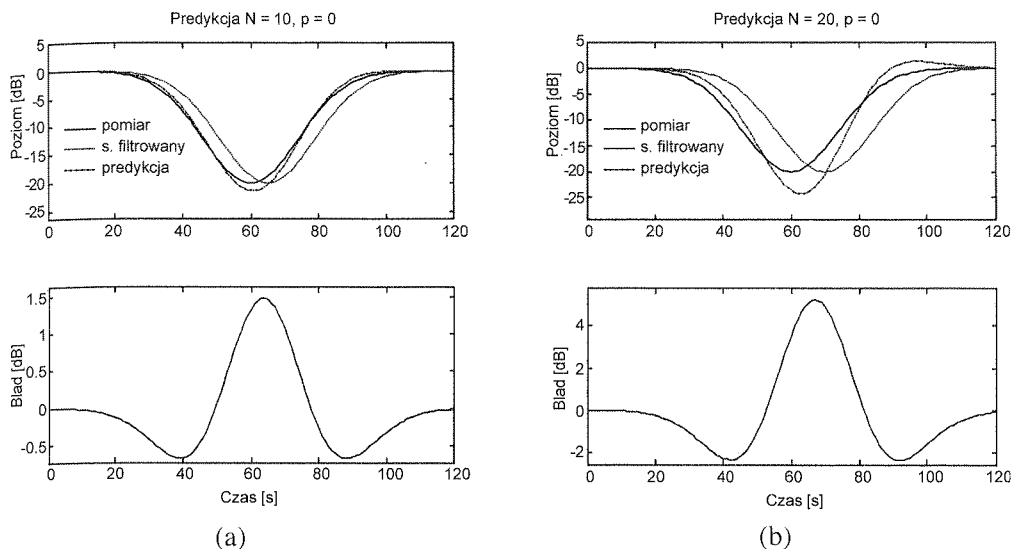
$$A_p(t_i + d_g + 1) = A(t_i) + (d_g + 1)[A(t_i) - A(t_{i-1})] \quad (1)$$

gdzie:

d_g – opóźnienie grupowe filtra;

A – wartość tłumienia wyrażona w dB.

Wadą opisanej metody jest nieprawidłowe działanie predyktora, w przypadku gdy zmienia się znak różnicy ΔA czyli w momencie gdy tłumienie osiąga wartość ekstremalną. Błąd predykcji jest tym większy im większe jest kompensowane opóźnienie. Na rys. 3 przedstawiono działanie predyktora dla różnych wartości opóźnienia grupowego. W celu poprawy czytelności rysunku z sygnału mierzonego usunięto szybkie fluktuacje, rysunki obrazują więc tylko efekt wpływu opóźnienia filtra na działanie predyktora.



Rys. 3. Działanie predyktora dla filtru rzędu 10 (a) i 20 (b)

W przypadku filtru rzędu 10 opóźnienie grupowe wynosi 5 s i może być dobrze skompensowane. W przypadku filtru dwukrotnie wyższego rzędu obserwuje się znaczny wzrost błędu predykcji. Powyższy problem może być rozwiązany przez zmianę sposobu działania układu kompensującego opóźnienie. Zamiast założenia o stałości szybkości narastania ΔA można przyjąć, że jest ona funkcją czasu, którą można przedstawić w postaci wielomianu stopnia p .

$$\Delta A(t) = a_p t^p + a_{p-1} t^{p-1} + \dots + a_1 t + a_0 \quad (2)$$

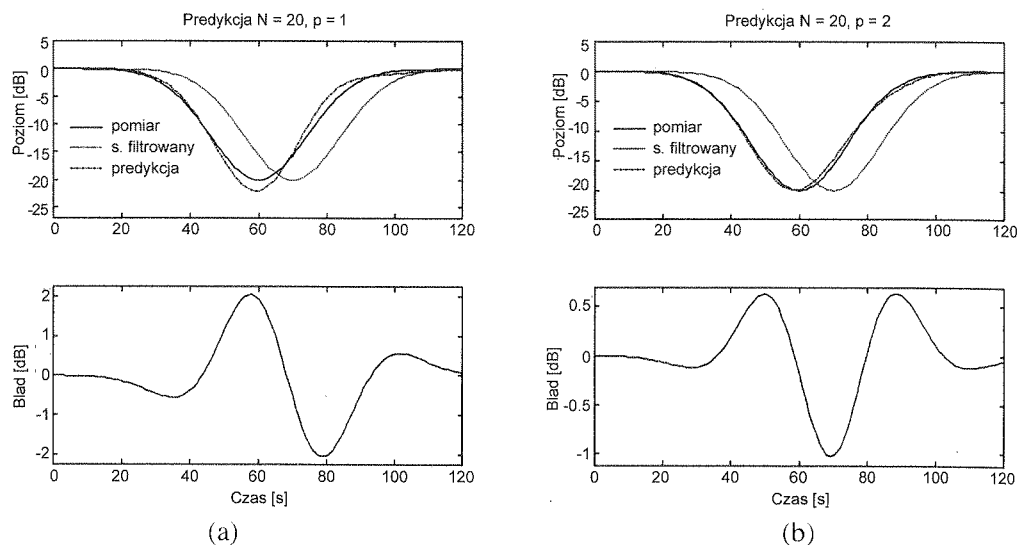
Znając wartość tłumienia w chwili t_i można obliczyć przewidywaną wartość tłumienia w chwili t_{i+n} jako:

$$A(t_{i+n}) = A(t_i) + \Delta A(t_{i+1}) + \Delta A(t_{i+2}) + \dots + \Delta A(t_{i+n}) \quad (3)$$

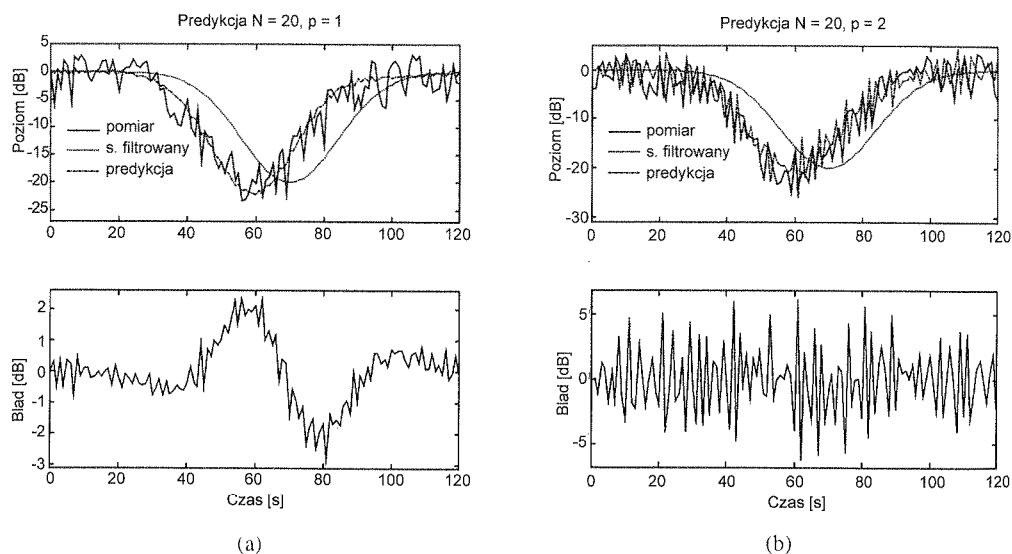
Współczynniki a_i wielomianu opisującego szybkość narastania tłumienia w czasie można znaleźć jako współczynniki wielomianu interpolującego przebieg ΔA w chwilach $t_i, t_{i-1}, \dots, t_{i-p+1}$. Działanie predyktora dla filtru rzędu 20 i stopnia wielomianu 1 i 2 pokazano na rys. 4.

W wyniku zastosowania opisanych modyfikacji metody predykcji uzyskano zauważalne zmniejszenie błędu predykcji. W przypadku obecności w sygnale filtrowanym pozostałości składowej szybkozmiennej interpolacja przebiegu szybkości narastania tłumienia wielomianem stopnia większego od 1 jest źródłem dużego błędu predykcji jak pokazano na rys. 5. W sygnale wyjściowym pojawiają się wówczas oscylacje o amplitudzie przekraczającej 5 dB.

Ponieważ w systemie z filtrem 20 rzędu w sygnale filtrowanym pozostałości składowej szybkozmiennej mogą być znaczne należy przyjąć, że stopień wielomianu



Rys. 4. Działanie predyktora z (a) liniową i (b) kwadratową interpolacją przebiegu szybkości narastania tłumienia



Rys. 5. Działanie predyktora z interpolacją (a) liniową i (b) kwadratową szybkości narastania tłumienia w przypadku obecności szybkich fluktuacji w sygnale

opisującego ΔA nie powinien być większy niż 1. Zastosowanie filtra większego rzędu, mimo potencjalnej możliwości lepszego wyeliminowania składowych szybkozmiennych, nie poprawia działania układu predykcyjnego ze względu na konieczność kompensacji większego opóźnienia.

2.4. CZĘSTOTLIWOŚCIOWE SKALOWANIE TŁUMIENIA

Skalowanie tłumienia w funkcji częstotliwości może być dokonane przy pomocy jednej z kilku metod. W symulacjach opisanych w dalszej części pracy korzystano z dwóch metod.

1. Metoda ITU-R

W metodzie tej, podanej w rekomendacji ITU-R P.618-5, skalowanie tłumienia z częstotliwości f_1 do f_2 odbywa się wg zależności:

$$A_{f_2} = A_{f_1} \left(\frac{\frac{f_2^2}{1 + f_2^2 10^{-4}}}{\frac{f_1^2}{1 + f_1^2 10^{-4}}} \right)^h \quad (4)$$

gdzie:

$$h = 1.12 \cdot 10^{-3} \sqrt{\frac{\frac{f_2^2}{1 + f_2^2 10^{-4}}}{\frac{f_1^2}{1 + f_1^2 10^{-4}}}} \left(\frac{f_1^2}{1 + f_1^2 10^{-4}} A_{f_1} \right)^{0.55} \quad (5)$$

2. Metoda Lastera i Stutzmana [7]

W metodzie tej tłumienia dla częstotliwości f_1 i f_2 są związane stałym współczynnikiem C :

$$A_{f_2} = C A_{f_1} \quad (6)$$

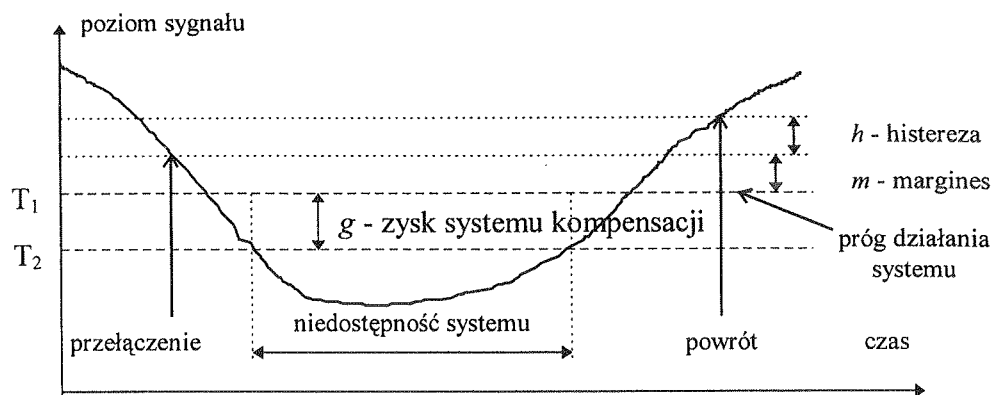
gdzie:

$$C = \left(\frac{f_2}{f_1} \right)^{1.99} \quad (7)$$

2.5. UKŁAD DECYZYJNY

Działanie układu decyzyjnego (dla systemu z dwoma stanami pracy) wyjaśniono na rys. 6.

Wartość progowa T_1 oznacza minimalny poziom sygnału, dla którego spełnione jest kryterium dostępności łącza (np. stopa błędów nie przekracza podanej wartości). Margines m , nazywany marginesem detekcji (ang. fade detection margin), ma za zadanie ochronę przed skutkami fluktuacji sygnału, a także przyspieszenie chwili zadziałania systemu kompensacji. Histerezę h wprowadza się w celu uniknięcia zbyt częstego przełączania parametrów systemu w przypadku gdy sygnał oscyluje wokół progu decyzyjnego.



Rys. 6. Działanie układu decyzyjnego systemu kompensacji tłumienia

Jeżeli poziom sygnału zmaleje poniżej wartości $T_1 + m$, aktywowany jest system kompensacji zaników. Nowa wartość progu wynosi $T_2 = T_1 - g$. Jeżeli poziom sygnału będzie mniejszy od tej wartości system będzie niedostępny (nie będzie spełnione kryterium jakości łącza). Przy powrocie do normalnego stanu pracy sprawdzane jest czy poziom sygnału osiągnął wartość $T_1 + m + h$. Jeśli tak, przywracane są nominalne wartości parametrów.

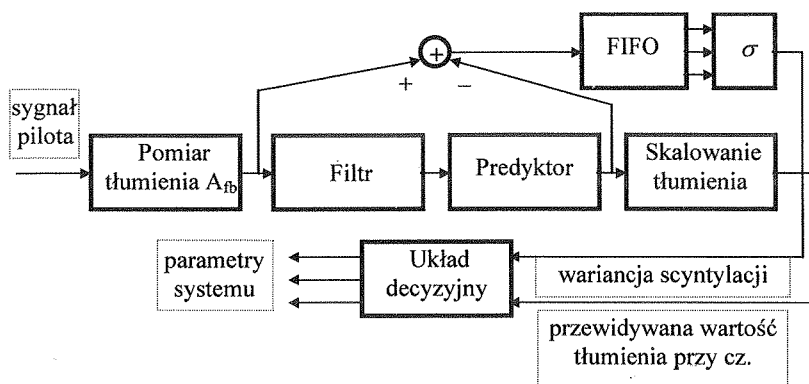
W ogólnym przypadku progów decyzyjnych może być kilka i przekroczenie każdego z nich spowoduje stopniowe uruchamianie kolejnych zasobów zarezerwowanych do kompensacji tłumienia. Z każdym z progów może być związana inna wartość marginesu detekcji m i histerezy h , jednak najważniejsze są wartości tych parametrów dla pierwszego progu T_1 , ponieważ w jego okolicy szybkość zmian poziomu sygnału jest najmniejsza i wahania poziomu sygnału najczęstsze [11].

Efektywność działania układu kompensacji jest uzależniona od doboru wartości marginesu detekcji m i histerezy h . Wartości tych parametrów mogą być stałe lub zmieniane automatycznie przez system.

W pracy zastosowano metodę automatycznego doboru wartości m i h [1, 13] polegającą na badaniu wariancji składowej szybkozmiennej usuwanej z sygnału odniesienia filtrem dolnoprzepustowym. Składową szybkozmienną sygnału można uzyskać odejmując od sygnału „pilota” sygnał z wyjścia predyktora (rys. 7). Próbkę składowej szybkozmiennej są podawane na wejście kolejki FIFO (ang. first-in first-out) o długości od 20 do 30 próbek. Zgromadzony w rejestrze przesuwającym zbiór próbek służy do wyznaczenia aktualnej wartości wariancji składowej szybkozmiennej.

Składowa szybkozmienna odpowiada scyntylacjom przy częstotliwości sygnału „pilota”. Miara intensywności scyntylacji jest wariancja lub odchylenie standardowe obliczane dla kilkudziesięciu próbek. Odchylenie standardowe scyntylacji dla częstotliwości nośnej może być obliczone z empirycznego wzoru [9]:

$$\sigma_{f_2} = \left(\frac{f_2}{f_1} \right)^{7/12} \sigma_{f_1} \quad (8)$$



Rys. 7. Schemat systemu kompensacji z automatycznym wyznaczaniem parametrów układu decyzyjnego

gdzie:

f_1 – częstotliwość sygnału odniesienia;

f_2 – częstotliwość nośna systemu;

σ_{f_1} – odchylenie standardowe scyntylacji dla częstotliwości f_1 ;

σ_{f_2} – odchylenie standardowe scyntylacji dla częstotliwości f_2 .

W powyższym wariancie systemu do wartości m i h dodawana jest dwukrotna wartość odchylenia standardowego σ_{f_2} . Zakłada się, że w wyniku zwiększenia wartości progu detekcji zaników i histerezy uzyska się lepszą ochronę systemu przed krótkotrwałymi zanikami poziomu sygnału.

3. SYMULACJE KOMPUTEROWE

W celu wyznaczenia wartości parametrów systemu kompensacji tłumienia zdefiniowano system działający w oparciu o adaptacyjną modulację PSK i dokonano jego symulacji dla różnych wartości rzędu filtru N , marginesu detekcji m i histerezy h .

Założono, że system kompensacji posiada cztery stany, tzn. w warunkach nominalnych wykorzystuje modulację 16-PSK i w miarę pogarszania się warunków transmisji krotność modulacji M jest zmniejszana kolejno do 8, 4 i 2. Jako kryterium jakości łącza przyjęto wartość $\text{BER} = 10^{-3}$. W Tabeli 1 podano wartości stosunku energii sygnału nośnego przypadającej na jeden bit do widmowej gęstości mocy szumów E_b/N_0 wymagane dla uzyskania żądanej jakości transmisji, względną szybkość przesyłania danych R_b oraz zysk systemu kompensacji G dla $M = 2, 4, 8$ i 16.

Zgodnie z tabelą 1, dla czterech stanów pracy systemu możliwe jest, przy założonym kryterium jakości, uzyskanie ponad 13 dB zysku kompensacji. Oznacza to, że w przypadku wystąpienia zaników o głębokości nie przekraczającej 13 dB system będzie w stanie zachować ciągłość transmisji. Przy zadziałaniu ostatniego stopnia kompensacji redukcja szybkości transmisji jest czterokrotna, jednak jak pokazują symu-

Tabela 1

Zysk systemu kompensacji w zależności od krotności modulacji PSK

M	2	4	8	16
$E_b/N_0(M)$ [dB]	6,79	6,79	10,02	14,35
$R_b(M)$	$0,25 R_{\max}$	$0,5 R_{\max}$	$0,75 R_{\max}$	$R_{\max}$
$G(M)$ [dB]	13,56	10,56	5,58	0

lacje, system przez większość czasu nadaje z maksymalną szybkością transmisji, a przez znaczny procent czasu wykorzystuje modulację 8-PSK przy szybkości transmisji zredukowanej do 0.75 szybkości maksymalnej.

Szybkość transmisji dobrano tak, aby uzyskać margines łącza równy 3 dB. W symulacjach pominięto blok skalowania tłumienia w funkcji częstotliwości, tzn. zarówno oceny jakości transmisji jak i badania dostępności systemu dokonywano na podstawie danych propagacyjnych zarejestrowanych dla częstotliwości 30 GHz.

Do symulacji wykorzystano dane propagacyjne (pomiaru poziomu sygnału odbieranego z satelity Olympus dla częstotliwości 20 i 30 GHz) zarejestrowane w lipcu i sierpniu 1990 r. w Belgii. W przypadku braku jakiegokolwiek kompensacji skutków tłumienia system charakteryzuje się parametrami podanymi w tabeli 2.

Tabela 2

Dostępność symulowanego systemu bez kompensacji tłumienia

	lipiec 1990	sierpień 1990
dostępność [%]	88,54	84,23

Z wyników przedstawionych w tabeli 2 widać, że przy niewielkim marginesie budżetu łącza (3 dB) nie jest praktycznie możliwe uzyskanie zadowalającej dostępności systemu bez zastosowania systemu kompensacji tłumienia. Wyniki symulacji obrazujące działanie systemu kompensacji przedstawiono w tabeli 3.

Tabela 3

Wpływ marginesu detekcji na dostępność systemu

rzęd filtru $N = 10$; $h = m$	$m = 0$ dB	$m = 0.1$ dB	$m = 0.2$ dB
dostępność [%] – lipiec 1990	98,30	99,43	99,78
dostępność [%] – sierpień 1990	97,27	99,00	99,61

Tabela 3 pokazuje wpływ doboru marginesu detekcji na dostępność systemu. Wraz ze wzrostem marginesu detekcji zwiększa się skuteczność wykrywania zaników, a tym

Tabela 1

a przez
zredu-y symu-
równow
dstawieału od-
w lipcu
kukców

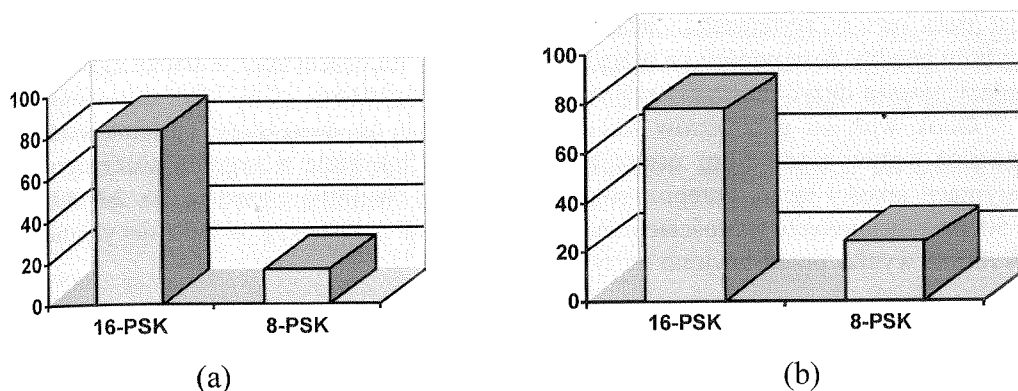
Tabela 2

ginesie
ępnosci
azujące

Tabela 3

. Wraz
, a tym

samym skuteczność działania systemu kompensacji. Dzięki jego zastosowaniu możliwe jest zwiększenie dostępności systemu z osiemdziesięciu kilku do ponad 99 %, co jest wystarczające dla wielu zastosowań. Należy jednak zauważyć, że zwiększanie marginesu detekcji i histerezy wpływa na zmianę proporcji czasu nadawania z maksymalną i z obniżoną szybkością przesyłania danych, co pokazano na rys. 8.



Rys. 8. Podział czasu transmisji między 16-PSK i 8 PSK dla $m = 0.1$ (a) i $m = 0.2$ (b)

Przy zwiększaniu marginesu detekcji rośnie procentowy czas przeznaczony na transmisję z obniżoną szybkością (w podanym przykładzie z 16,5 do 23,3%). Warto zaznaczyć, że dla symulacji wykonanych z użyciem danych z sierpnia 1990, łączny czas nadawania z modulacją QPSK nie przekroczył 0.3% czasu, a czas przeznaczony na nadawanie z modulacją BPSK 0.1% (tj. mniej niż 45 minut w skali miesiąca). Dla danych zarejestrowanych w lipcu czasy te wynosiły odpowiednio 0.1% dla QPSK i 0.01% dla BPSK. Oznacza to, że stosowanie wielu stopni w systemie kompensacji ma sens jedynie wówczas gdy dąży się do uzyskania dużej, przekraczającej 99.9 % dostępności systemu.

Skutkiem zmiany proporcji między czasem nadawania z maksymalną i obniżoną szybkością transmisji jest obniżenie średniej przepustowości systemu, co pokazano w tabeli 4.

Wpływ marginesu detekcji na przepustowość systemu

rzęd filtru $N = 10$; $h = m$	$m = 0$ dB	$m = 0.1$ dB	$m = 0.2$ dB
przepustowość [%] – lipiec 1990	95,30	95,26	93,90
przepustowość [%] – sierpień 1990	93,21	93,59	92,48

Tabela 4

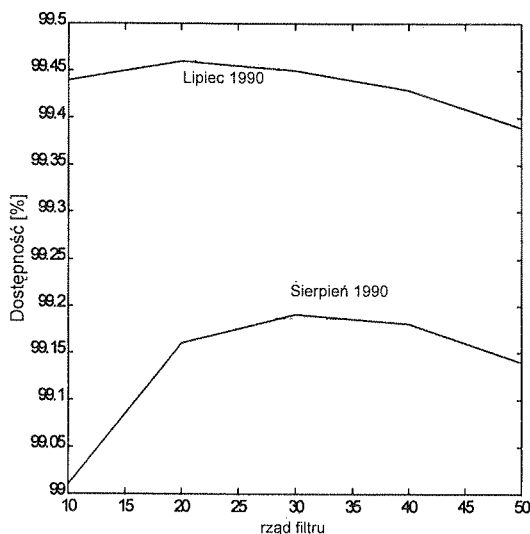
Przy dalszym zwiększaniu marginesu m powyżej 0.5 dB następuje znaczące obniżenie przepustowości systemu mimo, że dostępność systemu jest duża. Przy projektowaniu

systemu kompensacji niezbędne wydaje się więc przyjęcie kompromisu między żadaną dostępnością, a przepustowością systemu.

Decyzja o doborze wartości marginesu detekcji może być podjęta jedynie przy znajomości aplikacji wykorzystującej system transmisji danych. Jeżeli aplikacja toleruje zmienną szybkość transmisji (tak jest np. w większości systemów multimedialnych), to celowe jest maksymalizowanie dostępności systemu.

W przypadku systemów kompensacji, w których szybkość przesyłania informacji jest stała (np. w systemach z adaptacyjnym sterowaniem mocą nadawania) problem wyboru między dostępnością, a przepustowością nie ma znaczenia i w takich systemach należy zawsze dążyć do maksymalizacji dostępności systemu.

Obok wpływu parametrów układu decyzyjnego na dostępność systemu badano również wpływ rzędu filtru usuwającego szybkie fluktuacje z sygnału służącego do estymacji jakości łącza. Przeprowadzono symulacje dla filtru o rzędzie 10, 20, 30, 40 i 50 przy ustalonych wartościach marginesu detekcji i histerezy układu decyzyjnego. Wybrane wyniki przedstawiono na rys. 9.



Rys. 9. Zależność dostępności systemu od doboru filtru; symulacje dla danych z lipca i sierpnia 1990 r.; $m = 0.1$ dB i $h = 0.1$ dB

Na podstawie wyników symulacji, (przykładowe wyniki przedstawiono na rys. 8) można stwierdzić, że rzędz filtru powinien wynosić od 20 do 30. Z jednej strony zbyt mały rzędz filtru powoduje, że filtracja jest mało skuteczna, co prowadzi do błędów w działaniu układu decyzyjnego. Z drugiej strony ograniczeniem jest możliwość skompensowania opóźnienia grupowego wnoszonego przez filtr.

4. PRACA SYSTEMU W OBECNOŚCI ZANIKÓW SYGNAŁU WYNIKAJĄCYCH Z PROPAGACJI WIELODROGOWEJ

W przypadku odbioru sygnału w środowisku zabudowanym przez nisko zawieszoną antenę i przy małym kącie elewacji anteny (kąt między osią promieniowania anteny, a płaszczyzną gruntu) możliwe jest wystąpienie zaników sygnału spowodowanych propagacją wielodrogową. Zjawisko to jest szczególnie uciążliwe jeżeli moc sygnału odebranego bezpośrednio z nadajnika jest porównywalna lub mniejsza od mocy sygnału docierającego do odbiornika po dyfrakcji i odbiciu od przeszkód znajdujących się w otoczeniu odbiornika, zwłaszcza gdy antena stacji odbiorczej ma dużą szerokość głównego listka charakterystyki promieniowania.

Amplituda sygnału będącego wypadkową sygnałów docierających do odbiornika z różnymi wartościami amplitudy i fazy ma postać:

$$A = A_0 x \quad (9)$$

gdzie:

A_0 – amplituda sygnału przy braku zaników;

x – zmienna losowa.

Zmienna losowa x występująca we wzorze (9) ma funkcję gęstości prawdopodobieństwa nazywaną rozkładem Rice'a [10]:

$$p(x) = \frac{x}{\sigma^2} e^{-\frac{x^2 + r_\sigma^2}{2\sigma^2}} I_0\left(\frac{x r_\sigma}{\sigma^2}\right) \quad (10)$$

gdzie:

σ – reprezentuje średnią moc sygnału

r_σ – reprezentuje średnią moc sygnału docierającego do odbiornika drogą bezpośrednią;

I_0 – zmodyfikowana funkcja Bessela pierwszego rodzaju.

Stosunek mocy sygnału docierającego bezpośrednio do odbiornika do mocy sygnału docierającego drogami pośrednimi oznaczony symbolem K ma wartość [10]:

$$K = \frac{r_\sigma^2}{2\sigma^2} \quad (11)$$

Wartość średnia kwadratu zmiennej losowej x ma wartość:

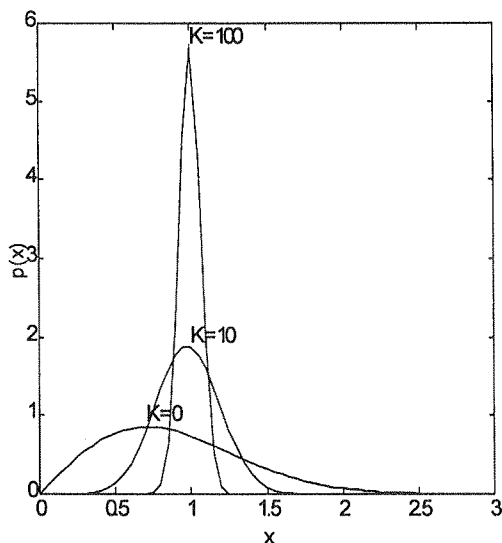
$$E_{x^2} = \int_{-\infty}^{\infty} x^2 p(x) dx = \sqrt{2\sigma^2 + r_\sigma^2} \quad (12)$$

Przy założeniu, że $E_{x^2} = 1$ (tzn. średnia moc sygnału w obecności zaników jest taka sama jak przy braku zaników) można uzależnić parametry rozkładu tylko od wielkości K .

$$r_g = \sqrt{\frac{K}{K+1}} \quad (13)$$

$$\sigma = \sqrt{\frac{1}{2(K+1)}} \quad (14)$$

Można pokazać, że dla $K = 0$ (tzn. przy braku sygnału bezpośredniego) rozkład Rice'a jest identyczny z rozkładem Rayleigha używanym do modelowania zjawiska szybkich zaników w radiokomunikacji ruchomej (zwłaszcza w środowisku miejskim). Przykładowy przebieg rozkładu Rice'a obliczony dla kilku wartości K podano na rys. 10.



Rys. 10. Funkcja gęstości prawdopodobieństwa w rozkładzie Rice'a dla różnych wartości K

Z wykresów przedstawionych na rys. 10 wynika, że wraz ze wzrostem K maleje prawdopodobieństwo powstawania głębokich zaników poziomu sygnału. Dla $K \rightarrow \infty$ rozkład Rice'a dąży do rozkładu Gaussa o wartości średniej równej $r_g = 1$. Biorąc pod uwagę wzór (9) długoterminowe prawdopodobieństwo błędnego odbioru symbolu w obecności zaników, przy modulacji M-PSK można wyznaczyć ze wzoru:

$$P_{sf}\left(\frac{E_s}{N_0}\right) = \int_0^\infty P_s\left(\frac{E_s}{N_0} x^2\right) p(x) dx \quad (15)$$

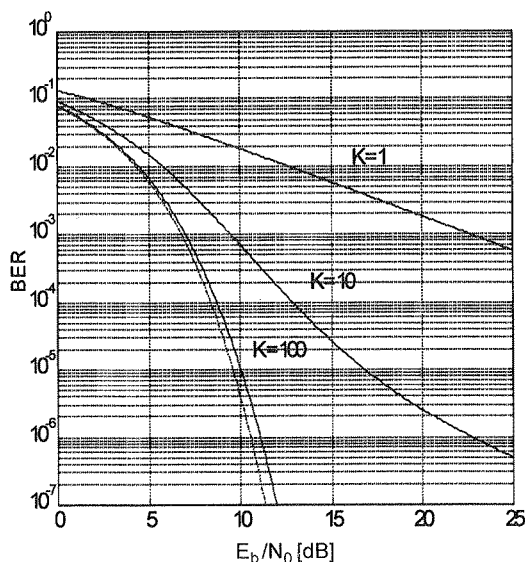
gdzie:

P_{sf} – prawdopodobieństwo błędnego odbioru symbolu w obecności zaników;

P_s – prawdopodobieństwo błędnego odbioru symbolu przy braku zaników [2, 3];

E_s/N_0 – stosunek energii sygnału nośnego przypadający na jeden symbol (E_s) do widmowej gęstości mocy szumu (N_0).

Dla kilku wartości K obliczono, przy pomocy wzoru (15), zależność stopy błędów od średniej wartości E_b/N_0 . Wynik porównania zamieszczono na rys. 11. Linia przerywaną oznaczono zależność stopy błędów (BER) od E_b/N_0 przy braku zaników. Z przedstawionych zależności wynika, że wraz ze zmniejszaniem się wartości współczynnika K określającego stosunek mocy sygnału docierającego bezpośrednio do odbiornika do mocy sygnału docierającego drogą pośrednią, rośnie znacząco prawdopodobieństwo błędnego odbioru przy zachowanym budżecie energetycznym łącza. Dla utrzymania pożądanej jakości konieczne jest zapewnienie odpowiednio większej wartości E_b/N_0 .



Rys. 11. Wpływ parametru K na przebieg zależności stopy błędów BER od E_b/N_0 dla modułacji BPSK

Na podstawie wykonanych obliczeń zaobserwowano, że przy malejącej wartości K zmniejszają się różnice wymaganej wartości E_b/N_0 dla różnych krotności modułacji PSK i ustalonej wartości BER. Przykładowe wartości E_b/N_0 dla różnych K i dla BER 10^{-3} podano w tabeli 5.

Z przedstawionych w tabeli danych wynika, że różnica wymaganej wartości E_b/N_0 dla ustalonej wartości BER i różnych wartości krotności modułacji M zmniejsza się wraz z malejącym K . Przykładowo, przy braku zaników różnica między wymaganym E_b/N_0 dla $M = 16$ i $M = 2$ wynosi $14.4 - 6.8 = 7.6$ dB, dla $K = 10$ różnica ta wynosi $16.7 - 9.5 = 7.2$ dB, a dla $K = 1$ zmniejsza się do 5.5 dB. Oznacza to, że skuteczność działania systemu kompensacji (zysk systemu kompensacji) w obecności szybkich zaników wynikających z propagacji wielodrogowej może być zauważalnie mniejsza niż w przypadku systemu, w którym problem ten nie występuje. Należy zatem dążyć, o ile

Tabela 5

Wymagana wartość E_b/N_0 w zależności od krotności modulacji M i wartości parametru K dla $BER 10^{-3}$

M	$K = \infty$	$K = 100$	$K = 10$	$K = 1$	$K = 0$
2	6.8	7.1	9.5	22.3	23.6
4	6.8	7.1	9.5	22.3	23.6
8	10.1	10.3	12.5	24.2	25.5
16	14.4	14.6	16.7	27.8	29.1

to możliwe, do maksymalizacji składowej sygnału docierającej do odbiornika drogą bezpośrednią i minimalizacji składowej związanej z propagacją wielodrogową. Można to osiągnąć poprzez zapewnienie możliwie dużego kąta elewacji anteny stacji naziemnej oraz zastosowanie anteny o małej szerokości głównego listka charakterystyki promieniowania. Spełnienie tych warunków może być w konkretnym systemie trudne. W przypadku terminali współpracujących z satelitą geostacjonarnym wielkość kąta elewacji anteny wynika z szerokości geograficznej, na której jest zlokalizowany terminal i w przypadku Europy Środkowej jest stosunkowo mała (mniejsza od 30°). Większą odporność na problemy związane z propagacją wielodrogową zapewniają systemy z satelitami pracującymi na orbitach niskich (LEO). Spełnienie drugiego warunku wymaga zastosowania większych i droższych systemów antenowych.

5. UWAGI I WNIOSKI

Z przeprowadzonych symulacji wynikają następujące wnioski dotyczące projektowania adaptacyjnych systemów kompensacji tłumienia dla łączy satelitarnych:

1. Usuwanie szybkich fluktuacji z sygnału pilota powinno być dokonywane za pomocą filtru o możliwie małej szerokości pasma przejściowego i dużym tłumieniu w pasmie zaporowym. W badaniach zastosowano filtr SOI rzędu 20 – 30 zaprojektowany z wykorzystaniem okna Czebyszewa.

2. Dobór parametrów układu decyzyjnego zależy od klasy systemu kompensacji. Dla systemów ze zmienną szybkością transmisji wartość marginesu detekcji i histerezy wynika z kompromisu między żądaną dostępnością i przepustowością systemu.

3. Możliwe jest automatyczne dobieranie wartości parametrów układu decyzyjnego na podstawie wariacji szybkozmiennej składowej sygnału „pilota”. Takie rozwiązanie zapewnia najlepszą ochronę systemu przed fluktuacjami sygnału wynikającymi ze scyntylacji.

4. Systemu kompensacji przeznaczony do pracy w obecności zaników wywołanych propagacją wielodrogową powinien być projektowany z uwzględnieniem skutków propagacji wielodrogowej, w szczególności należy wziąć pod uwagę mniejszy zysk kompensacji.

Powyższe wnioski zostały praktycznie wykorzystane m.in. w innych pracach autora dotyczących systemów kompensacji tłumienia wprowadzanego przez opady deszczu [4, 5, 12].

1. B.
V a
Ch
2. N.
pp.
3. C.
IEE
4. M.
Ma
ss.
5. M.
nat
22
6. M.
Jou
7. J.
Lin
8. A.
Co
Sa
9. T.
lan
pp
10. D.
11. Re
Pr
12. H.
VS
fo
13. M.
Co

I
modul
compu
propag
the fac
with c
either
Keywo

6. BIBLIOGRAFIA

1. B. Audoire, M. Bousquet, L. Castanet, M. Czarnecki, D. Mertens, A. Page, A. Vander Vorst, H. Vasseur: *Ka Band Videoconference VSAT System*, COST 255 Final Report, Chapter Va.2.
2. N. Celandroni et al: *Quality Estimation of PSK Modulated Signals*, IEEE Comm. Mag. Jul. 1997, pp. 50–55.
3. C. M. Chie: *Bounds and Approximations for Rapid Evaluation of Coherent MPSK Error Probabilities*, IEEE Trans. Comm., Vol. COM-33, No. 3, March 1985, pp. 271–273.
4. M. Czarnecki: *Wybrane problemy przesyłania strumieni multimedialnych w kanale satelitarnym*, Materiały konferencji Systemy i Technologie Telekomunikacji Multimedialnej STM2000, Łódź, 2000, ss. 225–230.
5. M. Czarnecki: *Compensation of Rain Attenuation for Ka-Band Satellite Systems*, Proc. XIII International Conference on Microwaves, Radar and Wireless Communications MIKON-2000, Wrocław, May 22–24 2000, pp. 439–442.
6. M. Filip, E. Vilar: *Adaptive Modulation As a Fade Countermeasure. An Olympus Experiment*, Int. Journal Sat. Comm., Vol. 8, 1990, pp. 31–41.
7. J. D. Laster, W. L. Stutzman: *Frequency Scaling of Rain Attenuation for Satellite Communication Links*, IEEE Trans. Antennas Propagat. Vol 43, No. 11, Nov. 1995, pp 1207–1216.
8. A. Malygin, M. Filip: *Deployment Strategy for a Fade Countermeasure System: A Generic Fade Countermeasure Controller*, Proc. 1st International Workshop on Radiowave Propagation Modelling for SatCom Services at Ku-band and Above, Oct. 1998, pp. 219–226.
9. T.J. Mouldsley, E. Vilar: *Experimental and Theoretical Statistics of Microwave Amplitude Scintillations on Satellite Down-links*, IEEE Trans. Antennas Propagat., Vol. AP-30, No. 6, Nov. 1982, pp. 1099–1106.
10. D. Parsons: *The Mobile Radio Propagation Channel*, Pentech Press, London, 1994.
11. Reference Book on Attenuation Measurement and Prediction, OPEX Second Workshop of the OLYMPUS Propagation Experimenters, Volume 1, Nov. 1994, pp. 87–123.
12. H. Vasseur, M. Czarnecki, L. Castanet, M. Bousquet: *Performance Simulation of a Ka-band VSAT Videoconferencing System*, Proc. 1st International Workshop on Radiowave Propagation Modelling for SatCom Services at Ku-band and Above, Oct. 1998, pp. 227–234.
13. M. J. Willis, B. G. Evans: *Fade Countermeasures at Ka Band for Olympus*, Intern. Journal of Sat. Comm., Vol. 6, 1988, pp. 301–311.

M. CZARNECKI

ADAPTIVE COMPENSATION OF THE INFLUENCE OF RAIN FADES
FOR SATELLITE COMMUNICATIONS SYSTEMS

Summary

In the paper an adaptive fade countermeasure system for Ka-band satellite systems utilizing M-PSK modulation with variable data rate has been described. A structure of such a system has been proposed and computer simulations of the performance of the system have been carried out. In the experiment real propagation data recorded during Olympus propagation experiment have been used. Also the performance of the fade countermeasure system in the presence of multipath effects has been analyzed. The results obtained with computer simulations may be used in the design of fade mitigation systems and allow for optimization of either the availability or the throughput.

Keywords: satellite communications, fade countermeasure, propagation

w u
mno
mno
pośr
DKC
prze
pośv
szyb
wyb
czas
mno
mno

Słow

M
twarza
Pierws
nik V
parall
VCM

Implementacja szybkich układów mnożących w strukturach FPGA

KAZIMIERZ WIATR, ERNEST JAMRO

*Katedra Elektroniki, Akademia Górniczo-Hutnicza w Krakowie
Al. Mickiewicza 30, 30-059 Kraków
e-mail: wiatr@uci.agh.edu.pl*

*Otrzymano 2001.03.23
Autoryzowano 2001.07.16*

Artykuł ten prezentuje różne rozwiązania szybkiego układu mnożącego implementowanego w układach reprogramowalnych FPGA. Przedstawione rozwiązania to: pełno-funkcjonalny układ mnożący o zmiennym współczynniku mnożenia VCM (ang. *Variable Coefficient Multiplier*), układ mnożący przez stały współczynnik KCM (ang. *Constant Coefficient Multiplier*) oraz rozwiązanie pośrednie – układ mnożący przez stały współczynnik z możliwością dynamicznej rekonfiguracji DKCM (ang. *Dynamic Constant Coefficient Multiplier*). Dla układów FPGA, które mogą być szybko przeprogramowane, wybór pomiędzy VCM i KCM jest trudnym zagadnieniem, któremu ten artykuł poświęca dużo uwagi. Co więcej istnieje rozwiązanie pośrednie – układ DKCM, który może być szybciej przeprogramowany niż KCM, ale zajmuje więcej zasobów układu FPGA. W układach FPGA wybór architektury układu mnożącego jest uzależniony od trzech czynników: zajmowanych zasobów, czasu propagacji oraz czasu przeprogramowania. W celu zwiększenia szybkości projektowania układu mnożącego zostało opracowane narzędzie do automatycznej generacji optymalnej architektury układu mnożącego w postaci kodu języka VHDL, na podstawie parametrów wejściowych.

Słowa kluczowe: układy dedykowane, procesory specjalizowane, układy programowalne, arytmetyka rozproszona

1. WSTĘP

Mnożenie równoległe jest powszechnie stosowaną operacją w cyfrowym przetwarzaniu sygnałów i może być przeprowadzone wykorzystując trzy różne architektury. Pierwszą architekturą jest pełno-funkcjonalny układ mnożący przez zmienny współczynnik VCM (ang. *Variable Coefficient Multiplier*), np. tablicowy układ mnożący (ang. *parallel array multiplier*) [1] lub układ mnożący z drzewem Wallace'a [2]. Dla układu VCM współczynnik mnożenia może być dowolnie zmieniany, ale układ ten zajmuje

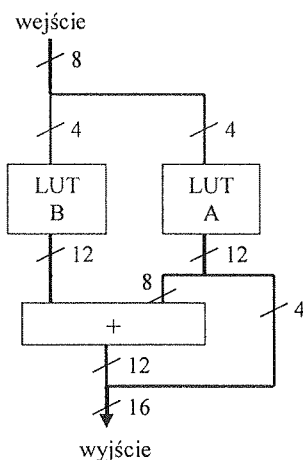
relatywnie dużą powierzchnię układu FPGA. Alternatywnym rozwiązaniem jest układ mnożący o stałym współczynniku mnożenia KCM (ang. *Constant Coefficient Multiplier*), który w porównaniu z układem VCM zajmuje dużo mniejszą powierzchnię (około 26÷33% układu VCM [3, 4]). W konsekwencji układ KCM jest zalecany pod warunkiem, że współczynnik mnożenia jest stały podczas procesu obliczeniowego. Dla układów ASIC i układu KCM, wartość współczynnika mnożenia raz zdefiniowana nie może być zmieniona. Prawdopodobnie ta nie jest jednak prawdziwa dla układów FPGA, dla których wartość współczynnika mnożenia może być zmieniona poprzez reprogramowanie układu FPGA. Proces reprogramowania przeważnie zajmuje od kilku do kilkudziesięciu ms [5], dlatego jeżeli proces obliczeniowy może być zaprzestany przez ten czas i współczynnik mnożenia jest relatywnie stały [6] powinno się użyć układu mnożącego KCM. Czas reprogramowania może być znacząco zmniejszony poprzez użycie układu FPGA z częściowym reprogramowaniem, np. układu Virtex firmy Xilinx [5], jednakże i w tym przypadku czas ten może być zbyt długi do zaakceptowania. Ponadto układ mnożący KCM musi być ponownie zaprojektowany dla innego współczynnika mnożenia. Istnieją narzędzia umożliwiające szybkie zaprojektowanie układu mnożącego KCM [7, 17], które automatycznie w przeciągu paru sekund wygenerują kod układu mnożącego. Jednakże nowy układ musi być powtórnie poddany procesowi implementacji (osadzania) w strukturze układu FPGA (ang. *place and rout*) co jest niezwykle czasochłonne (zazwyczaj od 1min. do 1godz.). W konsekwencji zmiana współczynnika mnożenia dla układu KCM wymaga nie tylko rekonfiguracji układu FPGA, ale przede wszystkim przeprojektowania układu. Dlatego często wybiera się mniej optymalne rozwiązanie czyli układ VCM.

Rozwiązaniem pośrednim jest układ mnożący przez stały współczynnik z możliwością dynamicznej rekonfiguracji DKCM (ang. *Dynamic Constant Coefficient Multiplier*). Układ DKCM jest zbudowany na bazie układu mnożącego KCM z wykorzystaniem pamięci LUT (ang. *Look-Up Table*), czyli układu LM (ang. *Look up table based Multiplier*) [4, 8, 9]. Dlatego zmiana współczynnika mnożenia jest dokonywana poprzez odpowiednią zmianę zawartości pamięci LUT. Zmiana zawartości pamięci LUT jest możliwa dzięki dodatkowemu układowi zaimplementowanemu we wnętrzu układu FPGA, dlatego zmiana współczynnika mnożenia jest szybka i nie wymaga powtórnego przeprogramowania układu. Wadą układu DKCM jest to, że układ zajmuje więcej miejsca niż układ KCM.

W pierwszej części tej pracy omówiony jest układ mnożący z wykorzystaniem pamięci LUT i jego modyfikacja dla układu DKCM. Układ DKCM może wykorzystywać trzy różne sposoby reprogramowania pamięci LUT: multipleksację w logice CLB, multipleksację w postaci buforów trójstanowych oraz w postaci pamięci dwuportowych. Dalsza część opracowania jest poświęcona porównaniu układów: KCM, DKCM oraz VCM. Poprzez skrótowe określenie „powierzchnia układu FPGA” autorzy rozumieją wielkość zużytych zasobów logicznych w danej aplikacji. Czasami zasoby te określa się mianem „kosztu” danego układu.

2. UKŁADY MNOŻĄCE LM Z WYKORZYSTANIEM PAMIĘCI LUT

Obliczenie dowolnej ograniczonej wartościowo funkcji możliwe jest z wykorzystaniem pamięci typu LUT, na której wejście adresowe podane są wartości argumentów, a na wyjściu otrzymuje się wynik funkcji. Zastosowanie pamięci LUT w teorii daje najszybszą implementację funkcji, ponieważ nie jest wymagane zastosowanie żadnych układów arytmetycznych. Niestety zastosowanie w praktyce pojedynczej pamięci LUT dla mnożenia jest możliwe tylko dla małych wartości argumentu, ponieważ wielkość pamięci wzrasta wykładniczo wraz z szerokością argumentu. Na przykład dla L -bitowego argumentu i K -bitowego współczynnika mnożenia, wymagana jest pamięć o rozmiarze $(L+K) \cdot 2^L$. Dlatego praktycznym rozwiązaniem jest zastosowanie kombinacji małych pamięci LUT i układów dodających poprzez rozbitcie argumentu oraz użycie małych pamięci LUT i następnie odpowiednie zsumowanie wyjść pamięci LUT [3, 8, 9]. Przykład takiego układu mnożącego jest pokazany na rysunku 1.



Rys. 1. Układ mnożący LM dla szerokości argumentu $L = 8$
i szerokości współczynnika mnożącego $K = 8$

Zawartość pamięci LUT dla mnożenia $Y = A \cdot B$ może być obliczona bezpośrednio z mnożenia tak jak to pokazano na przykładzie w tabeli 1.

Warto zwrócić uwagę, że zawartość komórek pamięci LUT zależy tylko od linii adresowych o wadze mniejszej lub równej wadze komórki wyjściowej. Dlatego w przedstawionym przykładzie w tabeli 1, zawartość komórki pamięci y_0 zależy tylko od linii adresowej a_0 ; y_1 zależy od a_0 i a_1 ; itd. Maksymalna szerokość magistrali jest jednak ograniczona od góry przez szerokość argumentu, dlatego szerokość magistrali adresowej komórki y_i w ogólnym przypadku wynosi $\text{MAX}(i+1, n)$, gdzie n -szerokość magistrali adresowej pamięci LUT. W konsekwencji $n-1$ najmniej znaczących bitów pamięci

Tabela 1

Zawartość komórek pamięci LUT ($y_5 - y_0$) dla różnych wartości argumentu podanego na magistralę adresową; współczynnik mnożenia wynosi 19; szerokość adresu – szerokość magistrali adresowej dla poszczególnych komórek pamięci

Adres	wartość	y_5	y_4	y_3	y_2	y_1	y_0
0	0	0	0	0	0	0	0
1	19	0	1	0	0	1	1
2	38	1	0	0	1	1	0
3	57	1	1	1	0	0	1
szerokość adresu		1	1	2	2	2	1

wymaga mniejszej szerokości magistrali adresowej, a przez to i mniejszych rozmiarów pamięci, co powoduje znaczącą redukcję kosztów pamięci i będzie oznaczane dalej jako LAWR (ang. *LSBs Address Width Reduction*).

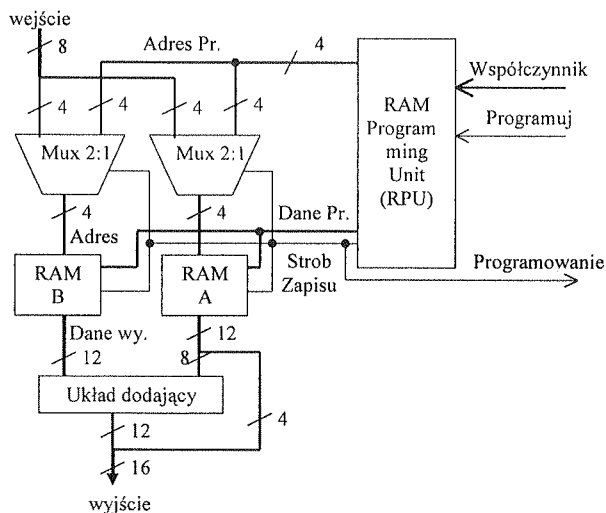
Dodatkową redukcję szerokość magistrali adresowej można zaobserwować dla komórek pamięci, dla których zawartość pamięci nie zależy od pewnej linii adresowej. Osiągnięte dzięki tej metodzie oszczędności będą dalej w skrócie nazywane jako DAWR (ang. *Don't care Address Width Reduction*). DAWR jest z reguły obserwowany dla najbardziej znaczących bitów pamięci LUT i najmniej znaczących wejść adresowych, co potwierdza przykład w tabeli 3, dla którego DAWR występuje dla komórek pamięci y_5 i y_4 . Dodatkowe oszczędności można osiągnąć dzięki dzieleniu wspólnych komórek pamięci MS (ang. *Memory Sharing*), jeżeli te komórki mają taką samą zawartość. Na przykładzie tabeli 1 można zauważyć, że komórki pamięci y_0 i y_4 są takie same, dlatego tylko jedna z nich może być zastosowana. Warto w tym miejscu podkreślić, że redukcję szerokości magistrali adresowej DAWR oraz dzielenie wspólnej pamięci MS nie można uogólnić i zależą one silnie od wartości współczynnika mnożenia i szerokości magistrali adresowej. Ponadto, wymagany jest zaawansowany algorytm przeszukujący i implementujący wspomniane techniki. W przedstawionych rozważaniach zastosowano algorytm porównujący wszystkie możliwe kombinacje komórek i linii adresowych.

3. UKŁAD MNOŻĄCY DKCM O STAŁYM WSPÓŁCZYNNIKU MNOŻENIA Z MOŻLIWOŚCIĄ DYNAMICZNEJ REKONFIGURACJI

Układ DKCM [7] nazywany również układem mnożącym z wbudowanym układem rekonfiguracji [10,11], jest układem LM, w którym pamięci LUT typu ROM zastąpione zostały pamięciami RAM. Zmiana współczynnika mnożenia polega więc na odpowiedniej zmianie zawartości pamięci LUT. W tym celu wymagany jest dodatkowy, wewnętrzny moduł generujący sekwencję danych programujących pamięć LUT nazywany

modułem RPU (ang. *RAM Programming Unit*). Ponadto sam układ mnożący ma pewne dodatkowe ograniczenia.

Dodatkowy układ służący do przeprogramowania pamięci LUT może być podzielony na dwie części: układ umożliwiający programowanie pamięci RAM, składający się przeważnie z multiplexerów adresu oraz właściwy moduł RPU. Przykład układu DKCM jest podany na rysunku 2. Jest to układ odpowiadający układowi KCM z rysunku 1.



Rys. 2. Przykład układu mnożącego DKCM dla danej wejściowej o szerokości 8

W układzie DKCM wskutek możliwości zmian wartości współczynnika mnożenia zmienia się zakres wyjść pamięci LUT, dlatego zarówno pamięci LUT jak i układ dodający muszą uwzględniać te zmiany. Najprostszym rozwiązaniem tego problemu w przypadku układu dodającego jest rozważanie tylko sumy wartości maksymalnych i minimalnych każdego z wejść. Jednakże w tym przypadku nie rozważa się korelacji pomiędzy wejściami co powoduje dodatkowy wzrost powierzchni zajmowanej przez układ dodający. Dlatego rozwiązanie to jest raczej nie do przyjęcia. Przykład układu dodającego rozważającego korelację pomiędzy wejściami i nie biorący pod uwagę tej korelacji jest podany w tabeli 2. Należy podkreślić, że w przypadku układu KCM, uwzględnienie korelacji pomiędzy wejściami tylko w małym stopniu wpływa na szerokość układu dodającego. Tabela 2 rozważa tylko dwuwejściowy układ dodający, przy czym w ogólnym przypadku należy również rozważać korelacje pomiędzy sumami cząstkowymi.

Tabela 2

Minimalne i maksymalne wartości oraz szerokość wyjść pamięci LUT z rysunku 2.

A, B – wyjścia z odpowiednich pamięci LUT,

Y – wyjście układu dodającego z uwzględnieniem i bez uwzględnienia korelacji.

Zakres danej wejściowej $-99+99$

	KCM, współ. = -99			KCM, współ. = 99			DKCM, współ. = $-99+99$		
	min.	max.	szerok.	min.	max.	szerok.	min.	Max.	szerok.
A	-1 485	0	12	0	1 485	11	-1 485	1 485	12
B	-594	693	11	-693	594	11	-693	693	11
Y bez korelacji	-10 989	11 088	15	-11 088	10 989	15	-24 453	24 453	16
Y z korelacją	-9 801	9 801	15	-9801	9801	15	-9 801	9 801	15

3.1. MULTIPLEKSOWANIE PAMIĘCI

Można zauważyć na rysunku 2, że pamięci RAM w układach FPGA z reguły mają oddzielne dane do zapisu i do odczytu [5], dlatego multipleksowanie danych lub też wykorzystanie buforów trójstanowych nie jest wymagane. Jednakże linie adresowe są te same dla odczytu i dla zapisu pamięci, dlatego wymagane jest dodatkowe multipleksowanie adresów w celu przełączania pomiędzy tymi adresami. Multipleksowanie może być zaimplementowane w układach FPGA z wykorzystaniem elementów logicznych wewnątrz bloków CLB (ang. *Configurable Logic Blocks*) [5] lub też z wykorzystaniem buforów trójstanowych TSB (ang. *Triple State Buffer*). Bufory trójstanowe nie zajmują dodatkowych CLB ponieważ są implementowane wykorzystując zasoby programowanych połączeń układu FPGA [5]. Jednakże multipleksowanie w TSB jest z reguły wolniejsze w porównaniu z multipleksowaniem w CLB oraz liczba buforów TSB w układzie FPGA jest ograniczona. Multipleksacja magistrali adresowej może być wyeliminowana poprzez użycie pamięci dwuportowej DP RAM (ang. *Dual Port RAM*). Rozwiązanie DP RAM jest zazwyczaj szybsze, nie ma opóźnień wnoszonego przez multipleksery, ale zazwyczaj zajmuje więcej powierzchni FPGA. Dla przykładu dla układów Virtex, pamięć 16×1 DP RAM zajmuje dwa razy więcej powierzchni niż pamięć jednoportowa SP RAM (ang. *Single Port RAM*) 16×1 . Z drugiej strony duże bloki 4kb pamięci BSR (ang. *BlockSelect RAM*) [5] są pamięciami dwuportowymi, dlatego w tym przypadku zastosowanie pamięci dwuportowej jest najlepszym rozwiązaniem. Podsumowując, optymalizacja układu DKCM wymaga rozważenia trzech różnych rozwiązań:

- multipleksacja w CLB (układ oznaczany jako DKCM-L),
- multipleksacja w buforach trójstanowych (DKCM-T),
- z użyciem pamięci dwuportowej (DKCM-D).

3.2. MODUŁ RPU PROGRAMUJĄCY PAMIĘĆ LUT

Głównym zadaniem bloku RPU jest dostarczenie sekwencji adresu i danych programujących pamięć LUT. Rozważmy najpierw przypadek, kiedy dana wejściowa jest nieujemna i wszystkie pamięci mają taką samą szerokość magistrali adresowej. W tym przypadku wszystkie pamięci mają wspólne linie adresowe i wspólne linie danych (tak jak na rys. 2), a moduł RPU składa się z licznika inkrementującego adres o jeden w każdym cyklu zegarowym oraz układu akumulującego, którego wartość zaczyna się od 0 i jest zwiększana cyklicznie o wartość nowego współczynnika mnożenia [11]. W konsekwencji sekwencja danych jest następująca:

$$\begin{aligned} d_0 &= 0 \\ d_1 &= d_0 + \text{coeff} = \text{coeff} \\ d_2 &= d_1 + \text{coeff} = 2 \cdot \text{coeff} \\ &\dots \end{aligned} \quad (1)$$

gdzie: d_i – dana dla adresu o wartości i , coeff – nowy współczynnik mnożenia.

W tym miejscu należy podkreślić, że liczba zapisów do pamięci (czas martwy układu mnożącego) zależy od pojemności pamięci, np. dla pamięci 16×1 potrzebne jest szesnaście cykli zapisu. Dlatego w niektórych rozwiązaniach korzystne jest wykorzystanie tylko części dostępnej pamięci w celu zmniejszenia czasu martwego układu mnożącego – układ mnożący podczas programowania nie funkcjonuje poprawnie. Jednakże częściowe wykorzystanie pamięci powoduje zwiększenie powierzchni zajmowanej przez układ mnożący [10].

Moduł RPU staje się bardziej skomplikowany w przypadku użycia pamięci o różnej szerokości magistrali adresowej wskutek użycia różnych bloków pamięci, albo też jeżeli szerokość danej wejściowej układu mnożącego nie może być równomiernie rozłożona pomiędzy wszystkie użyte pamięci. W tym przypadku sygnał *write enable* każdej pamięci powinien być nieaktywny po tym jak adres zapisu przepełni zakres adresowy pamięci. Jednakże wymaga to dodatkowej logiki sterującej poszczególnymi liniami *write enable*. Problem ten można rozwiązać poprzez programowanie adresu począwszy od maksymalnej dostępnej wartości (same jedynki) do zera. W ten sposób wszystkie pamięci mogą być zawsze zapisywane niezależnie od szerokości magistrali adresowej ponieważ ostatni zapis do pamięci jest poprawny. Sekwencję danych zapisujących pamięć można przedstawić następująco:

$$\begin{aligned} d_{s-1} &= (\text{coeff} \ll w) - \text{coeff} = (s-1) \cdot \text{coeff}; \\ d_{s-2} &= d_{s-1} - \text{coeff} = (s-2) \cdot \text{coeff} \\ &\dots \\ d_0 &= d_1 - \text{coeff} = 0 \end{aligned} \quad (2)$$

gdzie: $\text{coeff} \ll w$ – współczynnik mnożenia przesunięty 'w' bitów w lewo, w – maksymalna szerokość magistrali adresowej, s – maksymalny rozmiar pamięci: $s = 2^w$.

Wadą tego układu jest dodatkowy multiplexer 2:1 potrzebny do przełączania pomiędzy ($coeff \ll w$) a d_{i+1} . W przypadku sytuacji opisanej zależnościami (1) potrzebny był jednak układ zerujący, aby ustawić wartość początkową zero.

Blok RPU musi być dodatkowo zmodyfikowany w przypadku ujemnej danej wejściowej (zapisanej w kodzie uzupełnień do dwóch). W tym przypadku wszystkie pamięci oprócz najbardziej znaczącej (MSB) operują na dodatnich danych wejściowych, dlatego mogą być programowane zgodnie z wcześniejszym schematem. Podobnie jest dla MSB-RAM i dla najbardziej znaczącego bitu (bitu znaku) równego 0. Jednakże dla bitu znaku równego 1, pamięć MSB musi być programowana inną sekwencją danych, która może być wygenerowana w sposób nawiązujący do zależności (2):

$$\begin{aligned} \dots \\ d_0 &= d_1 - coeff = 0 \\ d_{-1} &= d_0 - coeff = -coeff \\ d_{-2} &= d_{-1} - coeff = -2 \cdot coeff \\ \dots \\ d_{-sn} &= d_{-sn+1} - coeff = -sn \cdot coeff \end{aligned} \quad (3)$$

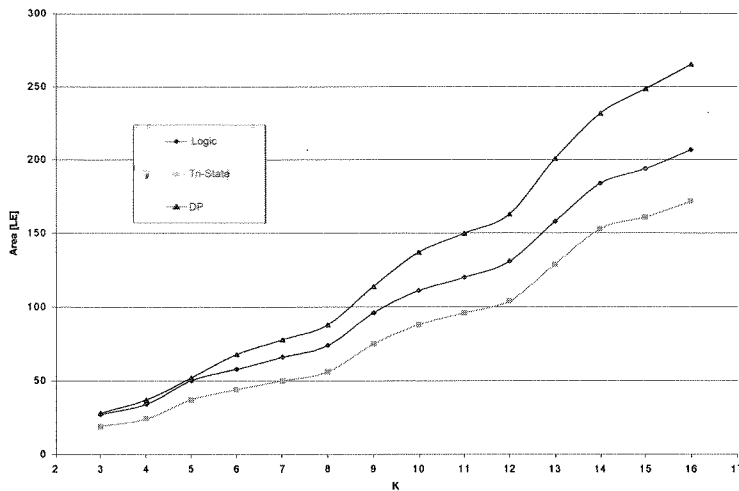
gdzie: sn – potęga rozmiaru pamięci MSB.

Warto podkreślić, że sposób opisany zależnościami (3) nie wymaga dodatkowych nakładów w porównaniu ze sposobem opisanym zależnościami (2), bowiem wykorzystuje ten sam układ odejmujący i licznik adresu, jednakże konieczny jest dodatkowy układ kontroli *write enable* wyłączający zapis wszystkich pamięci oprócz pamięci MSB w momencie przejścia licznika adresu przez zero. Warto podkreślić, że dla takiego rozwiązania czas martwy układu mnożącego ulega wydłużeniu o sn .

4. WYNIKI DOŚWIADCZALNE

4.1. WYNIKI DLA UKŁADU DKCM

Optymalna architektura układu DKCM zależy w dużym stopniu od wybranego układu FPGA, dlatego na początek zostaną przeprowadzone rozważania dla układu Xilinx XC4000 [5]. Mnożenie wymaga w pierwszym przybliżeniu: układu multiplexującego 2:1, dodającego oraz pamięci RAM, dlatego na razie będą rozważane tylko te elementy. Układy XC4000 posiadają pamięci 16×1 , 32×1 oraz pamięci dwuportowe 16×1 , których koszt jest następujący: 1, 2 i 2 LE (ang. *Logic Elements*), przy czym $1 \text{ LE} \approx \frac{1}{2} \text{ CLB}$. Układ dodający z przeniesieniem skrośnym RCA (ang. *Ripple Carry Adder*) zajmuje w przybliżeniu 1 LE/bit. Multiplexer implementowany wewnątrz CLB zajmuje 1 LE lub też wykorzystuje tylko zasoby programowanych połączeń, jeżeli jest implementowany w postaci buforów trójstanowych.

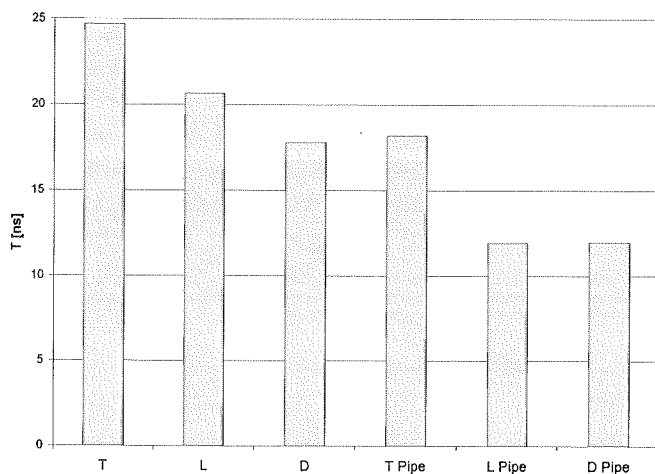


Rys. 3. Obszar zajmowany przez DKCM dla XC4000 dla różnej szerokości danej wejściowej i współczynnika mnożenia K

Obszar zajmowany przez układ DKCM przedstawia rysunek 3. Relatywnie wysoki koszt układu DKCM dla małych K jest spowodowany przez blok RPU, który przykładowo zajmuje 63% całego układu DKCM-L dla $K = 3$ i tylko 23% dla $K = 16$. Ponadto z rysunku 3 można odczytać, że najlepszym rozwiązaniem jest układ DKCM-T (z buforami trójszanowymi).

Wadą układu DKCM-T jest duży czas propagacji spowodowany przez opóźnienie wnoszone przez bufor trójszanowy co jest pokazane na rysunku 4. W wyniku tego popularny współczynnik $A \cdot T$ [12] (iloczyn obszaru i czasu propagacji), będący miarą jakości układu, może być gorszy niż dla innych układów mnożących. Rysunek 4 pokazuje, że duży czas propagacji buforów trójszanowych jest jeszcze bardziej widoczny dla potokowej architektury układu mnożącego, dla której wpływ czasu propagacji buforów trójszanowych na częstotliwość taktowania jest dominujący. Z drugiej strony w celu przyspieszenia pracy układu DKCM powinno się użyć pamięci dwuportowych. Rozwiązanie DKCM-D zajmuje jednak dużo większą powierzchnię układu FPGA i przynosi przyspieszenie tylko w przypadku zwykłego (bez użycia architektury potokowej) układu mnożącego. Jeżeli jednak koszt pamięci dwuportowej i jednoportowej jest ten sam powinno się użyć pamięć DP RAM.

Istnieje również pamięć pozorna 2×1 DP RAM, która jest implementowana jako 2-wejściowa bramka AND, dla której odpowiedni bit współczynnika mnożenia jest podawany na jedno wejście. Koszt tego modułu wynosi 1 LE, co stanowi połowę kosztu pamięci 16×1 DP RAM. Pamięć ta powinna być więc zaimplementowana dla bitu LSB każdej pamięci LUT (dla bitu LSB szerokość magistrali adresowej wynosi 1). Co więcej, przy użyciu tylko pamięci pozornej 2×1 DP RAM (zamiast innych pamięci) można uzyskać tablicowy układ mnożący [9] VCM.



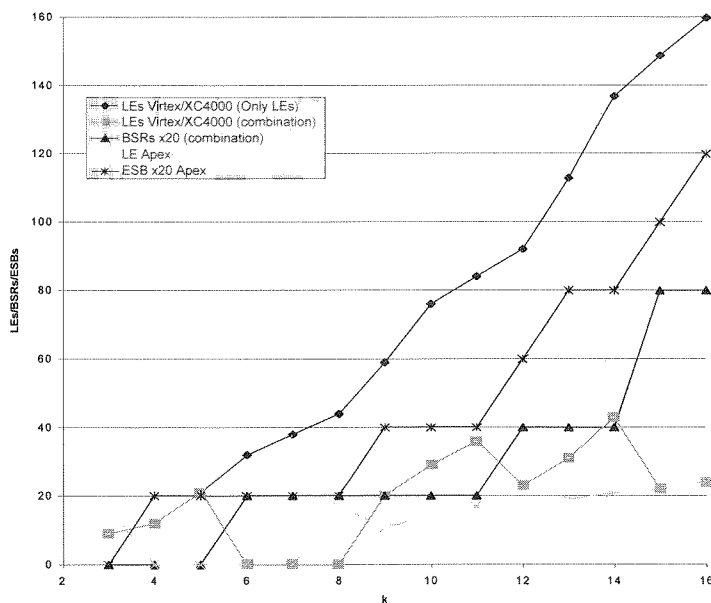
Rys. 4. Czas propagacji dla różnych rozwiązań układu DKCM oraz z architekturą potokową i bez jej zastosowania (układ XC4003 i $K = 8$)

Do tej pory rozważano tylko układy serii XC4000, które zawierają tylko małe pamięci RAM. Układy nowszej serii Virtex posiadają duże bloki pamięci dwuportowej BSR (ang. *BlockSelect RAM*) o pojemności 4kb i programowalnej szerokości: $4k \times 1$, $2k \times 2$, $1k \times 4$, 512×8 oraz 256×16 .

Konstruowanie optymalnego układu mnożącego składającego się z pamięci BSR, pamięci 16×1 , 32×1 , $16 \times 1DP$ RAM, pamięci pozornej $2 \times 1DP$ RAM oraz układów dodających i multiplexerów jest zadaniem skomplikowanym i wymaga rozważenia następujących zagadnień:

- Relacji kosztów pomiędzy wymienionymi elementami. Pamięć BSR zajmuje fizycznie w przybliżeniu 16 Virtex CLB ≈ 64 LE na powierzchni układu scalonego. Jednakże blok BSR nie może być przeprogramowany na CLB, co powoduje, że praktyczny koszt pamięci BSR jest zależny od aplikacji: w przypadku wolnych bloków pamięci BSR i zajętych bloków CLB można przenieść układ mnożący do pamięci BSR i na odwrót.
- Czas programowania układu mnożącego jest proporcjonalny do pojemności pamięci, dlatego w przypadku gdy czas martwy układu mnożącego odgrywa ważną rolę, mniejsze pamięci są preferowane.
- Czas propagacji układu mnożącego jest raczej mniejszy dla dużych rozmiarów pamięci, ponieważ liczba operacji arytmetycznych jest mniejsza. Z drugiej strony czas dostępu do pamięci z reguły zwiększa się wraz z rosnącym rozmiarem pamięci oraz jest utrudnione łączenie dużych bloków pamięci BSR z modułami arytmetycznymi, ponieważ położenie bloków BSR jest ściśle określone w układzie scalonym FPGA. Rozważania są jeszcze bardziej skomplikowane w przypadku architektury potokowej, gdzie należy uwzględnić koszt dodatkowych przerzutników oraz częstotliwości zegara.

Rysunek 5 pokazuje wynik doświadczalny z użyciem kombinacji pamięci BSR i małych pamięci 32×1 , 16×1 DP RAM i 2×1 DP RAM. Warto zwrócić uwagę, że liczba użytych bloków BSR zależy od przyjętej proporcji kosztów pomiędzy tymi blokami i dla proporcji $1 \text{ BSR} > 44 \text{ LE}$, pamięci BSR nie są w ogóle wykorzystywane.

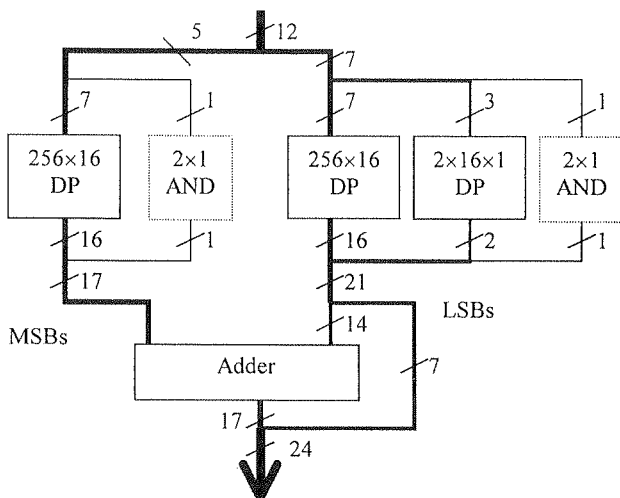


Rys. 5. Wzrost zużytych zasobów LE i BSR (w skali 20:1) dla układu Virtex, LE bez użycia BSR oraz LE i ESB (w skali 20:1) dla układów Apex. Moduł RPU nie jest brany pod uwagę, a koszt równoważny $1 \text{ BSR} = 1 \text{ ESB} = 20 \text{ LE}$

Układy Apex 20K firmy Altera [13] również posiadają wbudowane układy dodające z przeniesieniem skośnym, których koszt jest w przybliżeniu 1 LE/bit , ale w porównaniu z układami Xilinx XC4000 oraz Virtex, przy czym układy Apex posiadają tylko duże dwuportowe pamięci RAM $2k \times 1$, $1k \times 2$, 512×4 , 256×8 oraz 128×16 w każdym bloku ESB (ang. *Embedded System Block*). W konsekwencji, jak to można zobaczyć na rys. 5, liczba bloków ESB w porównaniu z liczbą bloków BSR jest większa, niemniej zmniejszyła się liczba użytych bloków LE. Konsekwencją braku małych pamięci w układach Apex jest również dłuższy czas reprogramowania układu mnożącego w porównaniu z układami Xilinx.

W celu szybkiej generacji układu mnożącego opracowano pakiet AuToCon (ang. *Automated Tool for Convolution processor design*), który automatycznie generuje układ mnożący sprawdzając wszystkie możliwe kombinacje dla podanych parametrów wejściowych: zakresu danej wejściowej i współczynnika mnożenia, relacji kosztów pomiędzy pamięciami, układami dodającymi, multiplekserami oraz przerzutnikami, itp. Te

parametry mogą być dowolne, dlatego program AuToCon może generować układ mnożący dla dowolnych układów FPGA, a nawet układów ASIC w postaci wynikowego kodu VHDL. W celu zobrazowania różnorodności rozważanych przez AuToCon układów na rysunku 6 przedstawiono przykład wygenerowanego układu. Warto zwrócić uwagę, że przedstawiony układ używa tylko pamięci dwuportowych oraz bramki AND, dzięki czemu multipleksowanie nie jest potrzebne (blok RPU nie jest pokazany).

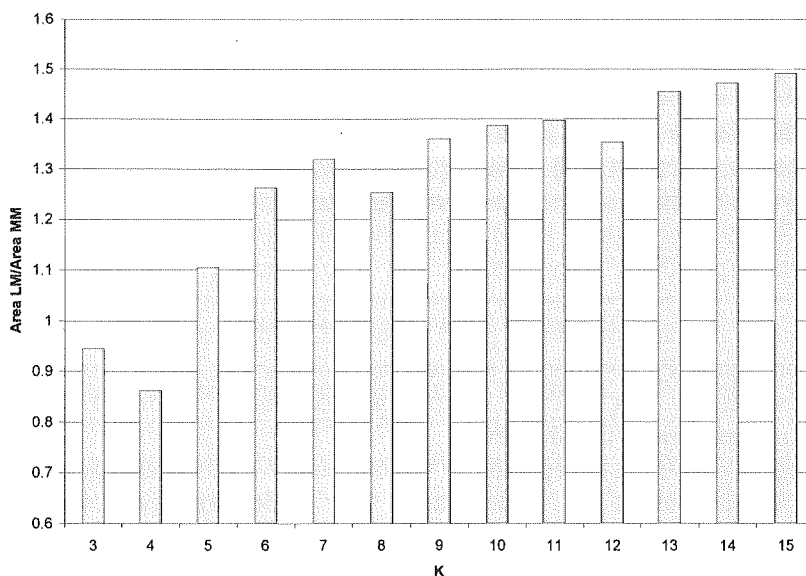


Rys. 6. Przykład układu mnożącego dla zakresu danej wejściowej i współczynnika mnożenia równego $0 + 2^{12} - 1$

4.2. PORÓWNANIE UKŁADÓW DKCM I KCM

Architektura układu DKCM wydaje się być bardzo podobna do architektury układu KCM, w której wymagany jest tylko dodatkowy blok RPU oraz multipleksowanie linii adresowych (rys. 1 i rys. 2). Jednakże, układ KCM może być zaprojektowany z użyciem pamięci LUT lub też przy pomocy alternatywnej techniki z użyciem tylko układów dodających i przesunięć bitowych MM (ang. *Multiplierless Multiplication*) [14]. Metoda MM jest jednak bardziej pożądana wraz ze wzrostem szerokości współczynnika mnożenia [17], ponieważ wraz ze wzrostem współczynnika mnożenia wzrasta skuteczność technik optymalizacji takich jak CSD (ang. *Canonical Sign Digit*) [14] czy też dzielenie wspólnej podstruktury SS (ang. *Substructure Sharing*) [15]. W konsekwencji dla układów Xilinx XC4000 oraz szerokości współczynnika mnożenia i danej wejściowej większych od 5 ($K \geq 5$), układ mnożący LM zajmuje przeciętnie 25-50% więcej powierzchni w porównaniu z układem MM [17], co jest pokazane na rysunku 7.

układ
owego
ukła-
wrócić
AND,



Rys. 7. Stosunek średnich powierzchni układu XC4000 zajmowanych przez układy mnożące KCM dla różnych opcji LM i MM (dane dla różnej szerokości danej wejściowej i współczynnika mnożenia K)

Układ LM dla stałego współczynnika mnożenia może być dodatkowo zoptymalizowany przy użyciu poniższych technik:

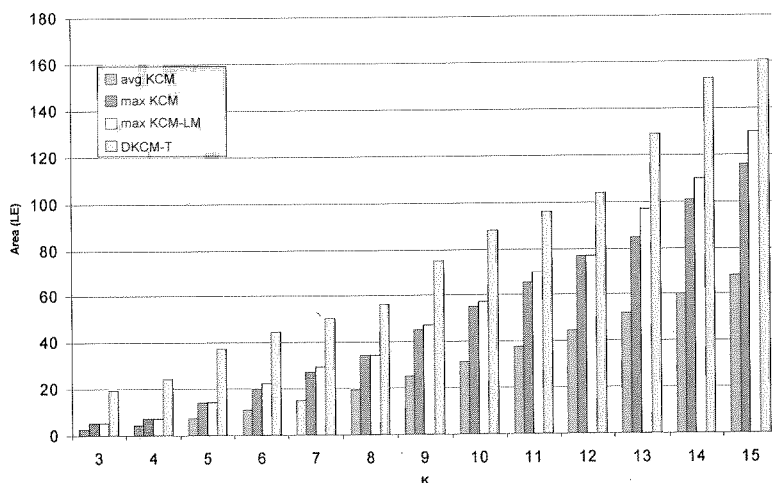
- Uproszczenie dla parzystych współczynników mnożenia – w tym przypadku bity najmniej znaczące (LSB) wyniku są zawsze równe zero. Co więcej dla współczynnika mnożenia 1, 2, 4, 8, itd. układ mnożący może być całkowicie zastąpiony przesunięciem bitowym.
- Redukcja rozmiaru pamięci LUT. W niektórych przypadkach szerokość magistrali adresowej maleje [17], ponieważ zawartość pamięci LUT nie zależy od wybranej linii adresowej. Przykład tej optymalizacji jest podany w tabeli 1, gdzie szerokość magistrali adresowej komórki y_3 jest zmniejszona z 2 do 1.
- Dzielenie wspólnej komórki pamięci. Niektóre komórki pamięci mogą posiadać taką samą zawartość pamięci i dlatego tylko jedna z nich musi być użyta. Dla przykładu, w tabeli 1 komórki y_0 i y_4 są takie same.
- Brak kodowania bitu znaku – dla stałego współczynnika mnożenia zawartość pamięci LUT jest z reguły zawsze nieujemna lub też zawsze niedodatnia [17], co pozwala na nie kodowanie bitu znaku.

Układ DKCM ze względu na swoją dynamikę może wykonywać mnożenie przez wiele współczynników w określonym zakresie, co wyraźnie odróżnia go od układu KCM, dla którego dla różnych mnożących wymagane jest projektowanie wielu układów. Co więcej architektura układu KCM różni się bardzo dla różnych współczynników

mnożenia, co powoduje duże różnice w zajmowanej powierzchni układu. Dlatego w celu porównania układów DKCM i KCM, powinny być uwzględnione trzy różne statystyczne koszty układu KCM:

1. Średni obszar zajmowany przez KCM dla danego zakresu współczynników mnożenia (zazwyczaj $1+2^K-1$). Ten koszt jest najbardziej odpowiedni dla statycznie konfigurowanych systemów [16] (układ FPGA jest konfigurowany tylko jednym kodem), dla których następuje statystyczne porównywanie odpowiednich układów DKCM i KCM. Średni koszt układu KCM (wraz z dodatkowym marginesem odchyleń) może być również przydatny przy dynamicznej rekonfiguracji [16] w przypadku kiedy równocześnie jest przeprogramowywana duża liczba układów mnożących KCM.
2. Maksymalny obszar zajmowany przez układ KCM dla danego zakresu współczynników mnożenia. Porównanie to jest rekomendowane dla dynamicznie rekonfigurowanych systemów, dla których wartość współczynnika mnożenia jest zmieniana poprzez rekonfigurację układu FPGA.
3. Maksymalny obszar dla danego zbioru współczynników mnożenia, stosowany podobnie jak w punkcie 2, ale w przypadku gdy zbiór możliwych współczynników mnożenia jest relatywnie mały. Ten koszt wydaje się być najodpowiedniejszy dla dobrze określonego projektu, jednakże może ograniczyć dalsze zmiany w projekcie. To rozwiązanie nie może być rozpatrywane w ogólnym przypadku i dlatego nie jest dalej rozważane.

Porównanie układów KCM i DKCM jest zilustrowane na rysunku 8. Dla małych wartości K obszar zajmowany przez układ DKCM jest znacznie większy niż dla KCM, w związku z silnym wpływem bloku RPU. Na rysunku 8 koszt bloku RPU jest

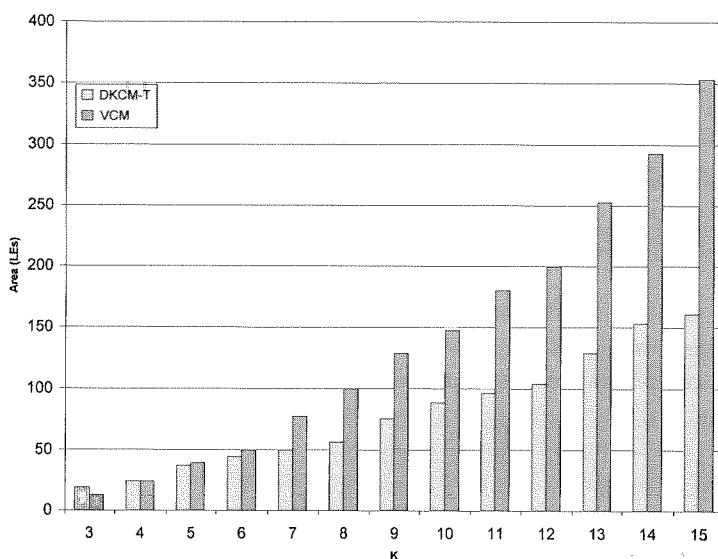


Rys. 8. Zajmowana powierzchnia układu XC4000 mierzona w LE (1LE = 1/2 CLB) dla: DKCM-T, maksymalnej KCM-LM (z użyciem pamięci LUT), maksymalnej KCM (LM albo MM) oraz średniej KCM (zakres danej wejściowej $0+2^K-1$, współczynnika mnożenia $1+2^K-1$)

z zilustrowany jako różnica pomiędzy kosztem układu DKCM-T i maksymalnym kosztem KCM-LM. Jednakże wraz ze wzrostem K relatywny koszt bloku RPU maleje i dodatkowy koszt układu DKCM w porównaniu z KCM zależy od strategii porównania: wartość średnia czy maksymalna układu KCM. Dla przykładu dla $K = 15$, dodatkowy koszt wynosi 39% i 136% dla maksymalnego i średniego kosztu układu KCM. Można również zauważyć, że dodatkowy koszt metody (LM kontra MM) wzrasta ze wzrostem K jak to można zobaczyć na rysunku 7 oraz na rysunku 8 (max KCM-LM kontra max KCM).

4.3. PORÓWNANIE UKŁADÓW DKCM I VCM

Układ mnożący o zmiennym współczynniku mnożenia VCM jest w pełni funkcjonalnym układem, z użyciem bramek AND i układów dodających [1,2], dla którego nie ma żadnych ograniczeń związanych ze zmianą wartości współczynnika mnożenia. Wadą układu VCM jest jego duży koszt w porównaniu z układami DKCM, co zostało zilustrowane na rysunku 9. Jednakże dla małych wartości K koszt układu DKCM jest zdominowany przez blok RPU i dlatego jest zalecany układ VCM.



Rys. 9. Zajmowana powierzchnia układu XC4000 mierzona w LE dla DKCM-T i VCM (dla różnych wartości K)

Według rysunku 9 układ DKCM powinien być użyty dla $K \geq 7$. Jednakże pokazana powyżej relacja kosztów jest najlepszą z możliwych do osiągnięcia w rzeczywistym przetwarzaniu danych, ponieważ układ DKCM wymaga przy zmianie wartości współ-

czynnika mnożenia przeprogramowania pamięci LUT, podczas którego nie może wykonywać operacji mnożenia. W konsekwencji można rozróżnić dwie, poniżej opisane, grupy aplikacji.

Grupa A

Projekt, w którym nie występuje negatywny efekt przeprogramowania pamięci LUT – zmiana współczynnika mnożenia jest bardzo rzadka i/lub nie zakłóca pracy układu. Na przykład dla przetwarzania obrazu w czasie rzeczywistym zmiana współczynnika mnożenia może przebiegać podczas powrotu plamki, kiedy nie jest ona widoczna lub też zmiana jest dokonywana tak rzadko, że jest prawie niezauważalna. W tym przypadku należy rozważyć układ KCM, który pozwoliłby na dodatkowe oszczędności.

W tym miejscu należy podkreślić, że dla DKCM-D (z użyciem pamięci dwuportowych) i dla systemu adaptacyjnego, dla którego różnica pomiędzy poprzednią i nową wartością współczynnika mnożenia jest niewielka, wynik mnożenia otrzymany podczas programowania pamięci DP RAM jest tylko nieznacznie przekłamany. Co więcej wynik mnożenia jest w zakresie wyników otrzymanych dla starej i nowej wartości współczynnika mnożenia, pod warunkiem, że wejście jest nieujemne oraz pod warunkiem, że równoczesny zapis i odczyt tej samej komórki pamięci dwuportowej nie powoduje dodatkowych przekłamań. W przypadku pamięci w układzie XC4000 równoczesny zapis i odczyt do tej samej komórki jest możliwy, a wartość wyjścia jest zdefiniowana [5].

Grupa B

Projekt, w którym czas reprogramowania układu DKCM wpływa na działanie układu, czyli przypadek kiedy współczynnik mnożenia zmienia się często i jego zmiana zakłóca działanie układu. W tym przypadku są możliwe cztery różne rozwiązania:

- DKCM-P – istnieją dwa równoległe zestawy pamięci RAM wraz z dodatkowymi zestawami multiplexerów [10]. Dla tego układu w chwili kiedy programowany jest jeden zestaw pamięci RAM, drugi zestaw jest używany do obliczeń.
- DKCM-D – metoda zaprezentowana w grupie A. Niestety dla tego układu wartość wyjściowa może być czasami przekłamana. Układ DKCM-D z reguły zajmuje więcej powierzchni niż porównywalny inny układ (np. DKCM-L), co jest swoistego rodzaju karą za dynamiczną zmianę wartości współczynnika mnożenia.
- Układ VCM, dla którego zmiana współczynnika mnożenia może być dowolna.
- DKCM – układ mnożący, dla którego proces obliczeniowy jest zatrzymany podczas rekonfiguracji pamięci LUT. W tym układzie należy również rozważyć wpływ zatrzymania obliczeń przez układ mnożący na całe otoczenie układu mnożącego.

W celu ocenienia korzyści płynących z zastosowania układu DKCM wprowadzone zostanie pojęcie gęstości funkcjonalnej D [6, 11]:

$$D = \frac{1}{A \cdot T} \quad (4)$$

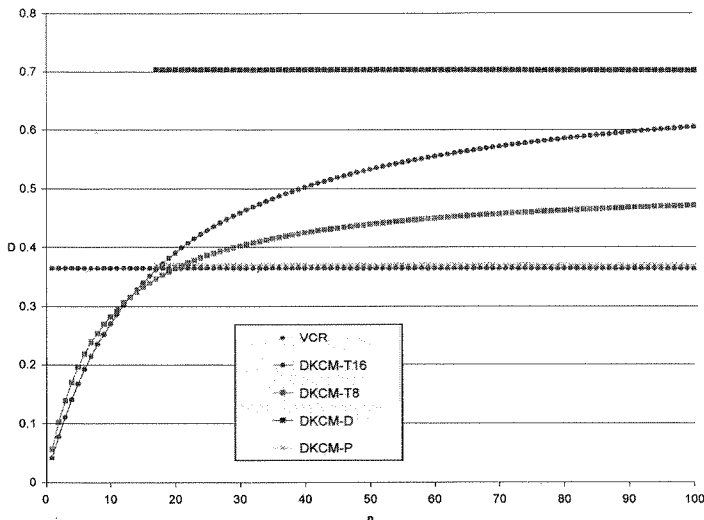
gdzie: A – powierzchnia, T – maksymalne opóźnienie.

Gęstość funkcjonalna dla układu DKCM musi jednak uwzględniać czas martwy podczas programowania pamięci LUT:

$$D_D = \frac{1}{A_D \cdot T_D \cdot \left(1 + \frac{r}{n}\right)} \quad (5)$$

gdzie: r – ilość cykli rekonfiguracji, n – ilość cykli pracy układu pomiędzy reprogramowaniem.

W równaniu (5) został wprowadzony dodatkowy czynnik karny r/n , który uwzględnia czas martwy podczas rekonfiguracji układu DKCM. Czynnik ten może ulec zmniejszeniu poprzez zwiększenie liczby cykli pracy n układu pomiędzy reprogramowaniem lub też poprzez zmniejszenie liczby cykli r potrzebnych do przeprogramowania pamięci LUT. Jednakże można zauważyć na rysunku 10 dla układu DKCM-T8, dla którego liczba cykli r została zmniejszona z $r = 16$ do $r = 8$ kosztem mniejszego wykorzystania pamięci 16×1 , że zmniejszenie liczby cykli konfiguracji r niewiele zwiększa gęstość funkcjonalną D dla małych n , a dla dużych n gęstość funkcjonalna układu DKCM-T8 jest dużo mniejsza w porównaniu z układem DKCM-T16 (z pełnym użyciem pamięci 16×1). W konsekwencji dla małych n powinno się używać układu VCM. Rozwiązaniem alternatywnym jest DKCM-D dla $n \geq 16$, dla którego zmiana współczynnika mnożenia nie powoduje zaprzestania procesu obliczeniowego, jednakże



Rys. 10. Gęstość funkcjonalna D [1/LE((s)] w zależności od liczby cykli n pomiędzy programowaniem dla układów: DKCM-D, DKCM-T16, DKCM-T8, DKCM-P i VCR ($K = 8$ i układ XC4003E-1, dla DKCM-P układ XC4005E-1 z powodu niewystarczającej liczby buforów trójstanowych dla tego rozwiązania)

podczas tej zmiany dane wyjściowe mogą być w małym stopniu przekłamane. Dla DKCM-P dodatkowy zestaw pamięci RAM oraz związane z tym układy multiplekserów powodują dodatkowy wzrost powierzchni układu i czasu propagacji co powoduje, że układ ten nie jest zalecany.

5. PODSUMOWANIE

Mnożenie jest podstawową operacją i jak przedstawiono powyżej, prawidłowy wybór jego architektury dla tej operacji układ FPGA nie jest rzeczą prostą. Dla układów ASIC można wybrać dwie architektury: VCM i KCM, dla których współczynnik mnożenia jest albo zmienny, albo też ustalony na stałe. Dla układów FPGA ta granica nie jest jednak tak łatwa do określenia, ponieważ układy FPGA mogą być szybko przeprogramowywane. Dlatego implementacja układów KCM jest zalecana, ponieważ układ ten zajmuje średnio $17 \pm 23\%$ i maksymalnie $29 \pm 41\%$ powierzchni układu VCM w zależności od szerokości układu mnożącego. Co więcej mniejszy obszar zajmowany przez układ mnożący często pociąga za sobą krótszy czas propagacji i w konsekwencji jeszcze większy wzrost gęstości funkcjonalnej. Z drugiej strony zmiana współczynnika mnożenia jest obciążona karą czasu przeprogramowania co zmniejsza gęstość funkcjonalną układu (równanie 5). W konsekwencji aby zmniejszyć liczbę cykli rekonfiguracji coraz powszechniej stosuje się rekonfigurację cząstkową.

Czas rekonfiguracji można dodatkowo zmniejszyć poprzez zastosowanie układu mnożącego o stałym współczynniku z możliwością dynamicznej rekonfiguracji (układ DKCM). Układ DKCM potrzebuje mniejszej liczby cykli rekonfiguracji, ale za to zajmuje większą powierzchnię w porównaniu z układem KCM, dlatego układ DKCM jest rozwiązaniem pośrednim pomiędzy KCM i VCM. Układ DKCM wymaga dodatkowego bloku RPU programującego pamięć LUT układu mnożącego, co znacząco wpływa na powierzchnię układu DKCM dla małych szerokości układu. Alternatywnym rozwiązaniem jest programowanie pamięci LUT za pomocą częściowego reprogramowania układu FPGA zamiast używania bloku RPU. W tym przypadku blok RPU oraz dodatkowa multipleksacja nie są potrzebne. Niestety częściowa rekonfiguracja może nie być dostępna w wybranym układzie FPGA lub jego otoczeniu oraz jest wolniejsza. Ponadto wymaga odpowiedniego przeliczenia zawartości pamięci LUT i umieszczenia przeliczonych wartości w odpowiednim miejscu kodu programującego układ FPGA. Dla adaptacyjnego procesu obliczeniowego zalecany jest układ DKCM-D, dla którego rekonfiguracja przebiega bez czasu martwego pod warunkiem, że dane wyjściowe mogą być zakłamane w niewielkim stopniu.

Proces zmiany współczynnika mnożenia dla układu KCM wymaga nie tylko reprogramowania układu FPGA, ale przede wszystkim powtórzonego projektowania układu mnożącego. Istnieją jednak programy automatycznie generujące kod układu mnożącego, dlatego najbardziej czasochłonne wydaje się być osadzenie projektu w strukturze fizycznej układu FPGA. To rozwiązanie nie jest jednak do zaakceptowania dla szybkich systemów adaptacyjnych.

W tym artykule umieszczono gruntowne rozważania nad implementacją układu mnożącego w układach FPGA, jednakże nie podano ogólnych wniosków jako, że optymalna architektura zależy silnie od wybranego układu FPGA oraz od założeń projektowych. Pokazano jednak możliwe rozwiązania architekture i wyniki doświadczeń, co może ułatwić wybór najlepszego rozwiązania. Ponadto warto podkreślić, że opracowano narzędzie AuToCon pozwalające na szybką automatyczną generację układu mnożącego, po zadaniu odpowiednich parametrów. Parametry takie mogą być zadawane ręcznie lub mogą pochodzić z innych bloków toru przetwarzania sygnału.

6. BIBLIOGRAFIA

1. F.P. Brooks, D.A. Plaisted: *Computer Arithmetic Systems*. Prentice Hall, 1994
2. C.S. Wallace: *A suggestion for a fast multiplier*. IEEE Trans. on Electron. Comput., 1964, Vol. EC-13, pp. 14–17
3. K. Chapman: *Constant Coefficient Multipliers for the XC4000E*. Xilinx Application Note, XAPP 054, December 1996
4. R. Petersen, B.L. Hutchings: *An Assessment of the Suitability of FPGA-Based Systems for Use in Digital Signal Processing*, In 5th International Workshop on Field Programmable Logic and Applications, Oxford England, August 1995, pp. 293–302
5. Xilinx Co.: *The Programmable Logic*, Data Book, 1999
6. M.J. Wirthlin, B.L. Hutchings: *Improving Functional Density Through Run-Time Constant Propagation*, ACM/SIGDA International Symposium on Field Programmable Gate Arrays, 1997, pp. 86–92
7. Xilinx Co.: *Core Generator*. Foundation 2.1, 1999
8. K. Chapman: *Fast Integer Multiplier fit in FPGA's*, EDN 1993 Design Idea Winner, END May 12th, 1994
9. A.R. Omondi: *Computer Arithmetic Systems. Algorithms, Architectures and Implementations*, Prentice Hall, 1994
10. M. Wojko, H. ElGindy: *Configuration Sequencing with Self Configurable Multipliers*. 13th International Parallel Processing Symposium and 10th Symposium on Parallel and Distributed Processing, San Juan, Puerto Rico, USA, April 1999, pp. 643–651
11. M. Wojko, H. ElGindy: *Self Configuring Binary Multipliers for LUT addressable FPGAs*. 5th Australasian Conference on Parallel and Real-Time Systems. University of Adelaide, Australia, 28-29th September 1998, pp. 201–212
12. C.L. Seitz: *Concurrent VLSI architectures*, IEEE Trans. on Computers, 1984, Vol. C-33, No. 12, pp. 1247–1265
13. Altera Co.: *Apex 20K Programmable Logic Device Family, Data Sheet*, ver. 2.05, Nov., 1999
14. H. Garner: *Number Systems and Arithmetic*. Advances in Computing, 1965, vol. 6, pp. 131–194
15. R.I. Hartley: *Subexpression Sharing in Filters Using Canonic Signed Digit Multipliers*. IEEE Transactions on Circuits and Systems II – Analog and Digital Signal Processing, vol. 43, no. 10, Oct. 1996
16. E. Sanchez, M. Sipper, J. Haenni, J. Beuchat, A. Perez-Urbe: *Static and Dynamic Configurable Systems*. IEEE Transactions on Computers, June 1999, col. 48, no. 6, pp. 556–563
17. K. Wiatr, E. Jamro: *Constant Coefficient Multiplication in FPGA Structures*. Proc. of the 26th Euromicro Conference on Digital Systems Design: Architecture, Methods, and Tools, Maastricht, Netherlands 2000, IEEE Computer Society – Washington – Brussels – Tokyo 2000, pp. 252–259
18. K. Wiatr, E. Jamro: *Implementation of Multipliers in FPGA Structures*, Proceedings of the IEEE International Conference on Quality Electronic Design, San Jose, USA, March 26-28, 2001

K. WIATR, E. JAMRO

HIGH SPEED MULTIPLICATION CIRCUITS
IMPLEMENTED IN FPGAs CHIPS

Summary

This paper studies different solutions for carrying out multiplication: a fully functional multiplier denoted as Variable Coefficient Multiplier (VCM), Constant Coefficient Multiplier (KCM) and self-configurable multiplier denoted as Dynamic Constant Coefficient Multiplier (DKCM). For FPGAs which can be easily reconfigured, the choice between the VCM and KCM cannot be easily defined. Furthermore, the DKCM is an additional, middle-way between the KCM and VCM solution, as it offers shorter reprogramming time but occupies more area in comparison with the KCM. In FPGAs, the choice of the optimum multiplier involves three factors: area, propagation and reconfiguration time, which have been thoroughly studied and respective implementation results given. Furthermore, to speed-up implementation of multipliers a design-automated tool has been developed, which generates optimum (for given input parameters), VHDL description of multipliers.

Keywords: dedicated circuits, specialised processors, programmable logic devices, distributed arithmetic

mu
gau
trak
zan

gdz
neg

wa
otr
cha

Jakość transmisji binarnej w kanale Hoyta

KRYSTYNA NOGA

Wyższa Szkoła Morska, Katedra Automatyki Okrętowej
ul. Morska 81/87, 81-225 Gdynia
e-mail: jagat@vega.wsm.gdynia.pl

Otrzymano 2001.05.11

Autoryzowano 2001.07.26

Do opisu sygnału użytecznego w kanale radiokomunikacyjnym stosuje się różne modele. W przypadku różnych mocy średnich jego składowych ortogonalnych i braku składowej dominującej sygnał ten posiada rozkład Hoyta. W pracy zostaną przedstawione zależności umożliwiające ocenę jakości transmisji binarnej w kanale z zanikami Hoyta.

Słowa kluczowe: kanał radiokomunikacyjny, kanał z zanikami, prawdopodobieństwo błędu elementowego, strumień błędów, wagowe widmo błędów, prawdopodobieństwo błędnego bloku

1. CHARAKTERYSTYKA SYGNAŁU UŻYTECZNEGO

W kanale radiokomunikacyjnym na sygnał nadany $u(t)$ działają losowe zakłócenia multiplikatywne $k(t)$ oraz addytywne. Do opisu zakłóceń addytywnych stosujemy szum gaussowski o zerowej wartości średniej. Natomiast działanie zakłóceń multiplikatywnych traktujemy jako przypadkowe tłumienie sygnału. Zjawisko to jest określane mianem zaników. Niech wąskopasmowy sygnał użyteczny $s(t)$ określony jest zależnością

$$s(t) = k(t)u(t) = x(t)\cos\omega_0 t - y(t)\sin\omega_0 t = r(t)\cos[\omega_0 t + \phi(t)] \quad (1)$$

gdzie $r(t) \geq 0$ jest obwiednią sygnału użytecznego, $\phi(t)$ jest fazą chwilową sygnału użytecznego.

Zakładamy, że składowe ortogonalne tego sygnału mają rozkład normalny o różnej wariancji (mocy średniej) $\sigma_x^2 \neq \sigma_y^2 > 0$ i o wartości średniej $m_x = m_y = 0$ (dla $\sigma_x^2 = \sigma_y^2$ otrzymujemy zaniki Rayleigha). Wówczas obwiednia sygnału (1) posiada rozkład Hoyta charakteryzujący się gęstością prawdopodobieństwa [1]

$$p(r) = \frac{r}{\sigma_x \sigma_y} \exp\left(-\frac{\sigma_x^2 + \sigma_y^2}{4\sigma_x^2 \sigma_y^2} r^2\right) I_0\left(\frac{\sigma_y^2 - \sigma_x^2}{4\sigma_x^2 \sigma_y^2} r^2\right) \quad (2)$$

gdzie:

$$I_0(\alpha) = \frac{1}{2\pi} \int_0^{2\pi} \exp(\alpha \cos \phi) d\phi = \sum_{k=0}^{\infty} \frac{(\alpha/2)^{2k}}{(k!)^2}$$

jest zmodyfikowaną funkcją Bessela pierwszego rodzaju rzędu zerowego [2].

Gęstość prawdopodobieństwa (2) można również przedstawić w następującej postaci [1]

$$p(r) = \frac{r}{\sigma_x \sigma_y} \exp\left(-\frac{r^2}{2\sigma_x^2}\right) {}_1F_1\left(\frac{1}{2}, 1, \frac{\sigma_y^2 - \sigma_x^2}{2\sigma_x^2 \sigma_y^2} r^2\right) \quad (3)$$

gdzie: ${}_1F_1(a, b, x)$ jest funkcją Kummera (zdegenerowaną funkcją hipergeometryczną) zdefiniowaną jako [2]

$${}_1F_1(a, b, x) = \frac{\Gamma(b)}{\Gamma(a)\Gamma(b-a)} \int_0^1 \exp(xt) t^{a-1} (1-t)^{b-a-1} dt; \quad \text{Re}(b) > \text{Re}(a)$$

lub

$${}_1F_1(a, b, x) = \frac{\Gamma(b)}{\Gamma(a)} \sum_{n=0}^{\infty} \frac{\Gamma(a+n) x^n}{\Gamma(b+n) n!}$$

$\Gamma(a) = \int_0^{\infty} t^{a-1} \exp(-t) dt$ jest funkcją gamma [2].

Rozkład Hoyta nazywany jest też rozkładem pod-Rayleigha [1] lub rozkładem Nakagami – q [5]. Gęstość prawdopodobieństwa obwiedni w kanale z zanikami Hoyta została przedstawiona na rysunku 1.

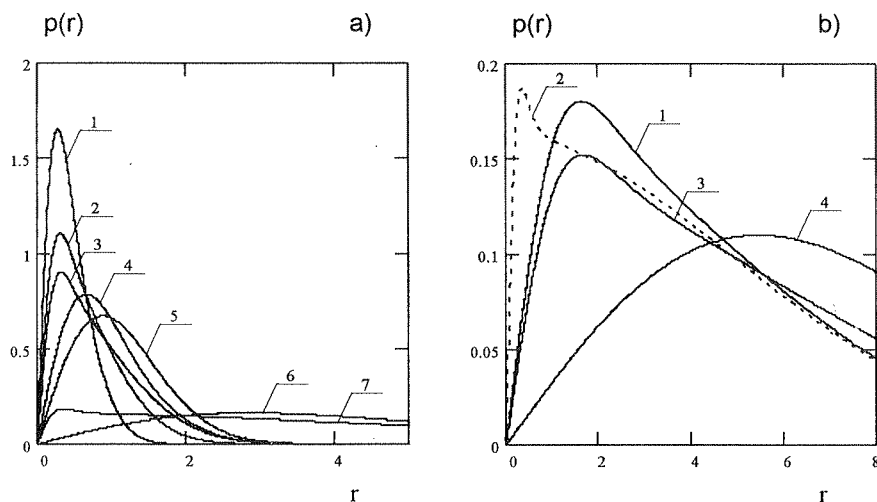
Rozkład Hoyta można zastosować do opisu obwiedni sygnału użytecznego w przypadku, gdy składowe ortogonalne tego sygnału posiadają różne moce średnie i gdy nie występuje składowa dominująca. W przypadku istnienia składowej dominującej i różnych mocy średnich składowych ortogonalnych, obwiednia sygnału posiada rozkład trójparametrowy lub czteroparametrowy [4].

W kanale Hoyta zmienna losowa $w = \frac{r^2}{2}$ reprezentująca moc sygnału użytecznego posiada następujący rozkład

$$p(w) = \frac{1}{\sigma_x \sigma_y} \exp\left(-\frac{\sigma_x^2 + \sigma_y^2}{4\sigma_x^2 \sigma_y^2} w\right) I_0\left(\frac{\sigma_y^2 - \sigma_x^2}{2\sigma_x^2 \sigma_y^2} w\right) \quad (4)$$

Jednocześnie zmienna losowa $q = \frac{r^2}{2N_0}$ reprezentująca chwilowy stosunek mocy sygnału użytecznego do średniej mocy N_0 zakłócenia addytywnego posiada rozkład

$$p(q) = \frac{N_0}{\sigma_x \sigma_y} \exp\left[-\left(\frac{\sigma_x^2 + \sigma_y^2}{\sigma_x \sigma_y}\right)^2 \frac{q}{4q_0}\right] I_0\left[\frac{q}{4q_0} \left(\frac{\sigma_y^4 - \sigma_x^4}{\sigma_x^2 \sigma_y^2}\right)\right] \quad (5)$$



Rys. 1. Gęstość prawdopodobieństwa obwiedni w kanale z zanikami Hoyta

- a) 1. $\sigma_x = 0.2, \sigma_y = 0.5$; 2. $\sigma_x = 0.2, \sigma_y = 0.8$; 3. $\sigma_x = 0.2, \sigma_y = 1$; 4. $\sigma_x = 0.5, \sigma_y = 1$; 5. $\sigma_x = 0.8, \sigma_y = 1$;
6. $\sigma_x = 2, \sigma_y = 5$; 7. $\sigma_x = 0.2, \sigma_y = 5$
b) 1. $\sigma_x = 1, \sigma_y = 5$; 2. $\sigma_x = 0.2, \sigma_y = 5$; 3. $\sigma_x = 1, \sigma_y = 6$; 4. $\sigma_x = 5, \sigma_y = 6$

gdzie $\varrho_0 = E(\varrho) = \frac{\sigma_x^2 + \sigma_y^2}{2N_0}$.

Wprowadźmy współczynnik $\beta = \frac{\sigma_x}{\sigma_y}$ charakteryzujący stopień niesymetrii kanału z zanikami. Wówczas funkcję gęstości prawdopodobieństwa określoną zależnością (5) można przedstawić w następującej równoważnej postaci

$$p(\varrho) = \frac{1}{2\varrho_0} \left(\frac{\beta^2 + 1}{\beta} \right) \exp \left[-\frac{\varrho}{4\varrho_0} \left(\frac{\beta^2 + 1}{\beta} \right)^2 \right] I_0 \left[\frac{\varrho(1 - \beta^4)}{4\varrho_0\beta^2} \right] \quad (6)$$

Jedną z miar jakości transmisji binarnej w kanale bez zaników jest statyczne prawdopodobieństwo błędu elementowego, które można obliczyć na podstawie zależności [9]

$$P_s(\varrho) = \frac{1}{2} \frac{\Gamma(b, a \cdot \varrho)}{\Gamma(b)} \quad (7)$$

gdzie: a i b to współczynniki zależne od rodzaju modulacji i detekcji: $a = b = 1/2$ dla CFSK, $a = 1/2, b = 1$ dla NCFSK, $a = 1, b = 1/2$ dla CPSK, $a = b = 1$ dla DPSK,

$\Gamma(a, x) = \int_x^\infty t^{a-1} \exp(-t) dt$ jest niepełną funkcją gamma [2],

przy czym dla $a \in \mathbb{N}$ $\Gamma(a, x) = (a-1)! \exp(-x) \sum_{k=0}^{a-1} \frac{1}{k!} x^k$.

Granice całkowania w (7) są zależne od zmiennej q , co utrudnia dalsze analityczne uśrednianie. Statyczne prawdopodobieństwo błędu elementowego można również przedstawić w następującej równoważnej postaci

$$P_s(q) = \frac{(a \cdot q)^{\frac{b-1}{2}}}{2\Gamma(b)} \exp\left(-\frac{a \cdot q}{2}\right) W_{\frac{b-1}{2}, \frac{b}{2}}(a \cdot q) \quad (8)$$

gdzie $W_{\lambda, \nu}$ jest funkcją Whittakera zdefiniowaną jako [2]

$$W_{\lambda, \nu}(x) = \frac{x^{\nu+\frac{1}{2}} \exp\left(-\frac{x}{2}\right)}{\Gamma\left(\nu-\lambda+\frac{1}{2}\right)} \int_0^{\infty} \exp(-xt) t^{\nu-\lambda-\frac{1}{2}} (1+t)^{\nu+\lambda-\frac{1}{2}} dt$$

Po uwzględnieniu postaci funkcji Whittakera zależność (8) można przedstawić jako

$$P_s(q) = \frac{(a \cdot q)^b}{2\Gamma(b)} \exp(-a \cdot q) \int_0^{\infty} \exp(-a \cdot q \cdot t) (1+t)^{b-1} dt \quad (9)$$

W zależności (9) górna granica całkowania wynosi nieskończoność. Jeszcze inną postać wzoru na statyczne prawdopodobieństwo błędu elementowego przedstawiono w pracy [8]

$$P_s(q) = \frac{(a \cdot q)^b}{\Gamma(b)} \int_0^{\pi/2} \frac{\cos \phi}{(\sin \phi)^{2b+1}} \exp\left(-\frac{a \cdot q}{\sin^2 \phi}\right) d\phi \quad (10)$$

2. DYNAMICZNE PRAWDOPODOBIENSTWO BŁĘDU ELEMENTOWEGO

W kanale z zanikami miarą jakości transmisji jest między innymi dynamiczne prawdopodobieństwo błędu elementowego określone zależnością

$$P_D(q_0) = \int_0^{\infty} P_s(q) p(q) dq \quad (11)$$

Zależność (11) jest jedną z miar jakości transmisji binarnej w kanale z zanikami stosowaną wówczas, gdy q zmienia się na tyle wolno, że może być uważane jako stałe w czasie transmisji sygnału elementarnego. Kanał, w którym spełniony jest ten warunek, nazywamy kanałem z wolnymi zanikami.

Wzór (11), po uwzględnieniu (10), przyjmuje postać

$$P_D(q_0) = \frac{a^b}{\Gamma(b)} \int_0^{\pi/2} \frac{\cos \phi}{(\sin \phi)^{2b+1}} \int_0^{\infty} q^b \exp\left(-\frac{a \cdot q}{\sin^2 \phi}\right) p(q) dq d\phi \quad (12)$$

Całka

$$K(\varrho_0; a, b, \phi) = \int_0^\infty \varrho^b \exp\left(-\frac{a \cdot \varrho}{\sin^2 \phi}\right) p(\varrho) d\varrho \quad (13)$$

jest transformatą Laplace'a funkcji $\varrho^b p(\varrho)$.

Dla odbioru NCFSK oraz DPSK statyczne prawdopodobieństwo błędu elementowego określone równoważnymi zależnościami (7)–(10) upraszcza się do postaci

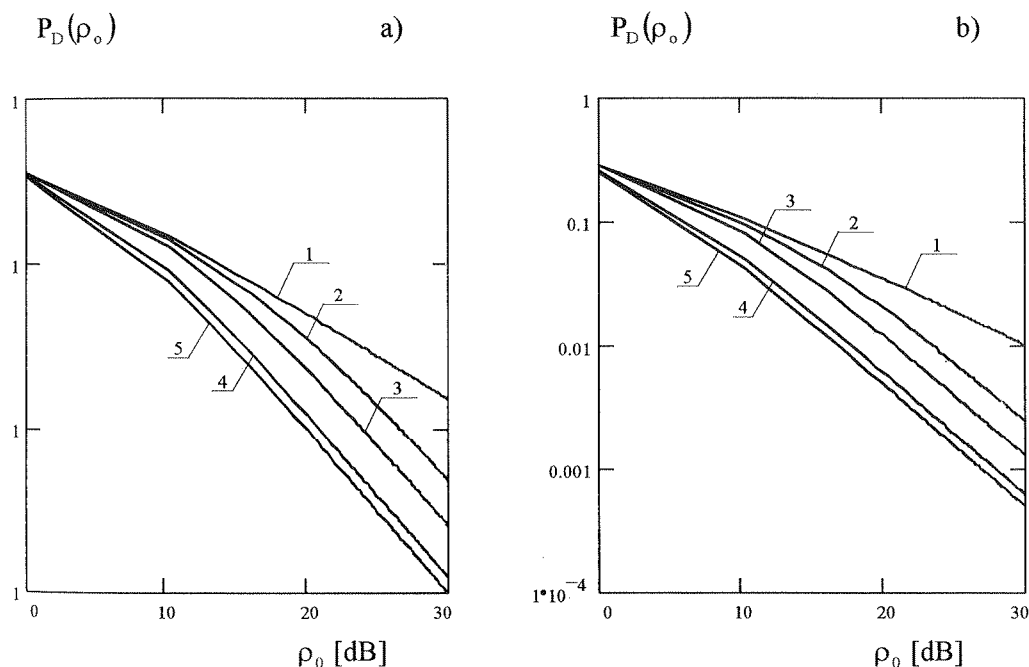
$$P_s(\varrho) = \frac{1}{2} \exp(-\alpha \cdot \varrho) \quad (14)$$

gdzie: $\alpha = 0.5$ dla NCFSK oraz $\alpha = 1$ dla DPSK.

Wówczas też dynamiczne prawdopodobieństwo błędu elementowego, określone zależnością (11), można obliczyć na podstawie wzoru

$$\begin{aligned} P_D(\varrho_0) &= \frac{1}{4\varrho_0} \left(\frac{\beta^2 + 1}{\beta} \right) \int_0^\infty \exp\left[-\frac{\varrho}{4\varrho_0\beta^2} [4\alpha\beta^2\varrho_0 + (1+\beta^2)^2]\right] I_0\left[\frac{(1-\beta^4)}{4\varrho_0\beta^2} \varrho\right] d\varrho = \\ &= \frac{(1+\beta^2)}{2} [(2\alpha\beta^2\varrho_0 + 1 + \beta^2)(2\alpha\varrho_0 + 1 + \beta^2)]^{-\frac{1}{2}} \end{aligned} \quad (15)$$

Dynamiczne prawdopodobieństwo błędu elementowego dla różnych wartości współczynnika β dla odbioru NCFSK zostało przedstawione na rysunku 2.



Rys. 2. Dynamiczne prawdopodobieństwo błędu elementowego dla odbioru

a) NCFSK b) DPSK i dla różnych wartości β

1. $\beta = 0.01$, 2. $\beta = 0.1$, 3. $\beta = 0.2$, 4. $\beta = 0.5$, 5. $\beta = 1$

3. CHARAKTERYSTYKA STRUMIENIA BŁĘDÓW

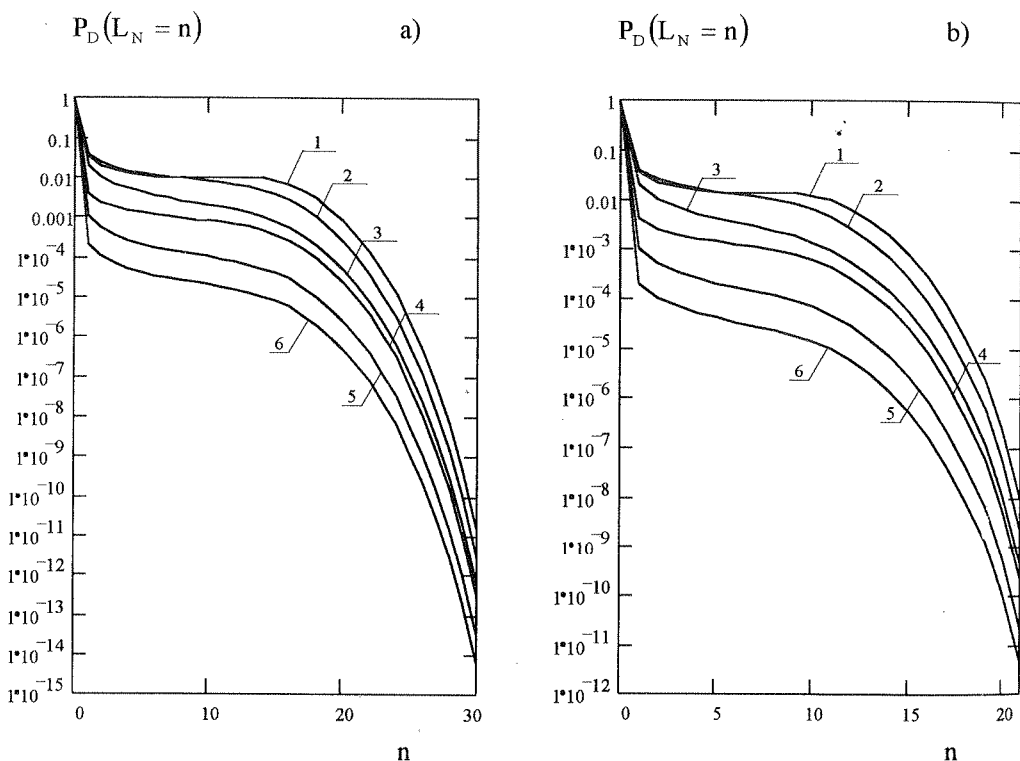
Miarą jakości transmisji w kanale z zanikami jest również średnie prawdopodobieństwo wystąpienia n błędów w ciągu kodowym o długości N bitów, które dla odbioru NCFSK można określić na podstawie zależności

$$P_D(L_N = n) = \int_0^{\infty} \binom{N}{n} [P_s(\varrho)]^n [1 - P_s(\varrho)]^{N-n} p(\varrho) d\varrho =$$

$$= \binom{N}{n} \sum_{i=0}^{N-n} \binom{N-n}{i} (-1)^i E[P_s^{i+n}(\varrho)] \quad (16)$$

gdzie: L_N jest zmienną losową reprezentującą liczbę błędów elementowych w ciągu kodowym o długości N sygnałów elementarnych.

Prawdopodobieństwo określone zależnością (16) charakteryzuje strumień błędów, często jest nazywane wagowym widmem błędów lub rozkładem krotności błędów. Zależność ta nie jest słuszna dla odbioru DPSK, gdyż różnicowa detekcja sąsiednich sygnałów elementarnych powoduje powstawanie par błędów.



Rys. 3. Wagowe widmo błędów dla odbioru NCFSK a) $N = 30$, b) $N = 21$

1. $\beta = 0.01$, $q_0 = 20$ dB, 2. $\beta = 0.1$, $q_0 = 20$ dB, 3. $\beta = 1$, $q_0 = 20$ dB,
4. $\beta = 0.01$, $q_0 = 40$ dB, 5. $\beta = 0.1$, $q_0 = 40$ dB, 6. $\beta = 1$, $q_0 = 40$ dB

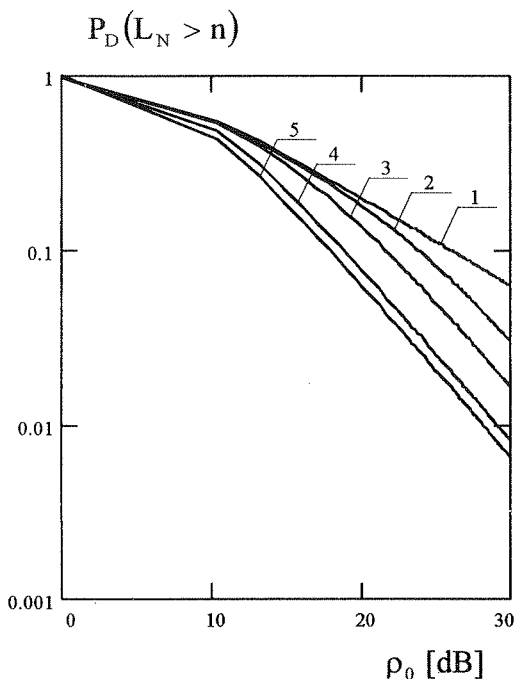
W kanale z zanikami Hoyta wagowe widmo błędów można określić na podstawie

$$P_D(L_N = n) = \binom{N}{n} \sum_{i=0}^{N-n} \binom{N-n}{i} (-1)^i \left(\frac{1}{2}\right)^{i+n} (1+\beta^2) \times \\ \times \{[\beta^2 \varrho_0(i+n) + 1 + \beta^2][\varrho_0(i+n) + 1 + \beta^2]\}^{-\frac{1}{2}} \quad (17)$$

Zależność (17) została przedstawiona graficznie na rysunku 3.

Strumień błędów można również opisać przy pomocy średniego prawdopodobieństwa wystąpienia w ciągu kodowym N elementowym więcej niż n błędów, które dla kanału z zanikami określa zależność

$$P_D(L_N > n) = \int_0^\infty \sum_{k=n+1}^N \binom{N}{n} [P_s(\varrho)]^k [1 - P_s(\varrho)]^{N-k} p(\varrho) d\varrho = \\ = \sum_{k=n+1}^N \binom{N}{k} \sum_{i=0}^{N-k} \binom{N-k}{i} (-1)^i E[P_s^{i+k}(\varrho)] \quad (18)$$



Rys. 4. Dynamiczne prawdopodobieństwo wystąpienia co najmniej jednego błędu ($n = 0$) dla odbioru NCFSK w bloku o długości $N = 30$

1. $\beta = 0.01$, 2. $\beta = 0.1$, 3. $\beta = 0.2$, 4. $\beta = 0.5$, 5. $\beta = 1$

Prawdopodobieństwo określone wzorem (18) jest nazywane dynamicznym prawdopodobieństwem błędnego bloku. W kanale z zanikami opisanymi rozkładem Hoyta dla odbioru NCFSK prawdopodobieństwo błędnego bloku przyjmuje postać

$$P_D(L_N > n) = \sum_{k=n+1}^N \binom{N}{k} \sum_{i=0}^{N-k} \binom{N-k}{i} (-1)^i \left(\frac{1}{2}\right)^{i+k} (1+\beta^2) \times \\ \times \{[\beta^2 \varrho_0(i+k) + 1 + \beta^2][\varrho_0(i+k) + 1 + \beta^2]\}^{-\frac{1}{2}} \quad (19)$$

Dynamiczne prawdopodobieństwo wystąpienia co najmniej jednego błędu ($n = 0$) zostało przedstawione na rysunku 4.

4. PODSUMOWANIE

Obecność w kanale sygnałów zakłócających powoduje błędy pomiędzy wyjściowymi i wejściowymi ciągami danych. Aby osiągnąć wysoki poziom niezawodności niezbędne jest zastosowanie kodowania. Celem, do jakiego się dąży przy kodowaniu jest zwiększenie odporności cyfrowego systemu telekomunikacyjnego na zakłócenia kanałowe. Przedstawione w artykule zależności można wykorzystać do oceny jakości transmisji w kanale Hoyta, a przede wszystkim do opisu błędów seryjnych spowodowanych wielodrogowymi zanikami. Błędy te w kanałach radiowych są powodowane również przez zakłócenia atmosferyczne oraz interferencje od innych użytkowników pasma. Opis zakłóceń atmosferycznych wymaga zastosowania innych rozkładów, np. hiperbolicznego, gamma. Natomiast do opisu zakłóceń interferencyjnych można zastosować powszechnie znane rozkłady, np. Nakagami, Rice'a, Rayleigha, również Hoyta [1]. Przedstawione w pracy zależności można zastosować w technice kodowania z kontrolą błędów, która stanowi istotny składnik niezawodnych systemów radiokomunikacyjnych. Wpływ błędów jest zredukowany przez dodanie po stronie nadawczej nadmiaru do przekazywanych danych. Nadmiar ten umożliwia dekodownikowi w odbiorniku wykrycie i korekcję błędów.

5. BIBLIOGRAFIA

1. D.D. Kłowski: *Teoria pieriadaczi sygnałów*, Moskwa, Swiaż 1973
2. I.S. Gradsztejn, I.M. Ryzik: *Tablice całek, sum, szeregów i iloczynów*, Warszawa, PWN, 1964
3. J.G. Proakis: *Digital communications*, New York, McGraw-Hill, third edition 1995
4. K. Noga: *Zaawansowane modele probabilistyczne sygnałów w kanałach z zanikami*, Kwartalnik Elektroniki i Telekomunikacji, 2000, Z. 1, ss. 47–62
5. M.K. Simon, M.S. Alouini: *A unified approach to the performance analysis of digital communication over generalized fading channels*, Proceedings of the IEEE, Vol. 86, No 9, September 1998, pp. 1860–1877
6. M.K. Simon, M.S. Alouini: *A unified approach to the probability of error for noncoherent and differentially coherent modulations over generalized fading channels*, IEEE Transactions on Communications, Vol. 46, No 12, December 1998, pp. 1625–1638

7. M.K. Simon, M.S. Alouini: *A unified performance analysis of digital communication with dual selective combining diversity over correlated Rayleigh and Nakagami – m fading channels*, IEEE Transactions on Communications, Vol. 47, No 1, January 1999, pp. 33–43
8. M.S. Alouini, M.K. Simon: *Generic form for average error probability of binary signals over fading channels*, Electronics Letters, 14 th May 1998, Vol. 34, No 10, pp. 949–950
9. S. Lindner: *Statystyki błędów transmisji binarnej w kanale z bardzo wolnymi zanikami Nakagamięgo i szumem gaussowskim*, Archiwum Elektrotechniki, Zeszyt 3/4, 1987, ss. 957–965

K. NOGA

PERFORMANCE OF BINARY TRANSMISSION IN HOYT CHANNEL

Summary

The transmission of binary information over radio channels with multiple changing propagation paths is typically subject to random fading. In this paper are presented the average probability of bit errors $P_D(Q_0)$ for slow Hoyt fading. The average is formed over the instantaneous receiver signal – to – noise – ratio (SNR) which has the probability density function $P_D(Q_0)$. The analysis has been limited to the special case of very slow signal fading and Gaussian noise. For digital transmission over a fading channel, the time variation causes a changing error probability with the effect of clustering errors at the receiver output. In this paper are derived the average probability of n errors occurring in N bits and the average block error probability, i.e. the probability of more than n errors in a block of N symbols. Expressions are derived for systems employing noncoherent frequency shift keying (NCFSK). Although DPSK has basically the same simple formula for the bit error probability $P_b(Q)$ in channel without fading with the instantaneous SNR Q and Gaussian noise, the block error probability is difficult to obtain due to bit error correlation. The numerical results presented in this paper are applicable to block transmission employing error detection and error correction. The results presented in paper provide useful design data and establish the need for specialised treatment of block error in a fading environment.

Keywords: radio communication channel, Hoyt channel, probability of bit errors, probability of block error, probability of n errors occurring in N bits

No
dyn

ny
zał
tra
wz

Nowy algorytm wyznaczania nieizotermicznych charakterystyk dynamicznych elementów półprzewodnikowych w programie SPICE

JANUSZ ZARĘBSKI, KRZYSZTOF GÓRECKI

Katedra Radioelektroniki Morskiej, Wyższa Szkoła Morska

81-225 Gdynia, ul. Morska 83

E-mail: zarebski@wsm.gdynia.pl

gorecki @wsm.gdynia.pl

Otrzymano 2001.07.20

Autoryzowano 2001.09.03

Praca dotyczy problemu wyznaczania charakterystyk dynamicznych elementów półprzewodnikowych w programie SPICE z uwzględnieniem zjawisk termicznych. Takie charakterystyki, nazywane nieizotermicznymi, różnią od charakterystyk izotermicznych, odpowiadających ustalonej temperaturze i wyznaczanych bezpośrednio za pomocą programu SPICE. W celu otrzymania nieizotermicznych charakterystyk dynamicznych elementów półprzewodnikowych, autorzy opracowali i uruchomili nowy algorytm ITER rozwiązywania równań, opisujących model elektryczny i termiczny elementu. W algorytmie tym wykorzystano też nową metodę wyznaczania mocy czynnej wydzielanej w elemencie półprzewodnikowym, niezbędnej do wyznaczenia temperatury wnętrza elementu w poszczególnych krokach obliczeń. Dobra zgodność przedstawionych w pracy wyników symulacji charakterystyk wybranych elementów półprzewodnikowych, pracujących w prostych układach elektronicznych, otrzymanych z wykorzystaniem nowego algorytmu oraz znanego z literatury algorytmu analizy elektrotermicznej, opartego na metodzie wspólnych iteracji, stanowi potwierdzenie skuteczności i wiarygodności algorytmu ITER.

Słowa kluczowe: SPICE, analiza elektrotermiczna, modelowanie elementów półprzewodnikowych

1. WPROWADZENIE

Program SPICE jest powszechnie uznanym, wygodnym symulatorem przeznaczonym do analizy układów elektronicznych, zwłaszcza układów scalonych. Jego istotną zaletą są wbudowane fizyczne modele elementów półprzewodnikowych, np. diody p-n, tranzystora bipolarnego, tranzystora MOS oraz wybranych układów scalonych, np. wzmacniaczy operacyjnych, z parametrami uzależnionymi od temperatury.

W znanych autorom wersjach programu SPICE temperatura, która może być różna dla każdego elementu analizowanego układu, jest wielkością ustaloną, niezmienną w czasie trwania obliczeń, co jest istotnym ograniczeniem programu, rzutującym bezpośrednio na wiarygodność otrzymanych wyników obliczeń. Dotyczy to zwłaszcza układów pracujących w warunkach wydzielania znacznych mocy elektrycznych, np. układów impulsowego przetwarzania energii.

Jak wiadomo właściwości materiałów półprzewodnikowych, a tym samym elementów oraz układów zależą od temperatury wnętrza T_j , która różni się od temperatury otoczenia T_a na skutek efektów termicznych – samonagrzewania oraz wzajemnych oddziaływań termicznych między elementami. Wartość temperatury T_j wpływa też silnie na niezawodność elementów i układów.

Wyznaczenie temperatury wnętrza elementów półprzewodnikowych w analizach za pomocą programu SPICE, zwłaszcza w przypadku dynamicznym, nie jest zadaniem trywialnym chociaż dzisiaj można już mówić o stosunkowo bogatej literaturze poświęconej problematyce modelowania i analizy, za pomocą programu SPICE, układów z uwzględnieniem efektów termicznych, np. [1, 2, 3, 4, 5, 6, 7, 8, 9]. Przedstawiony w cytowanych pracach sposób realizacji takiej analizy nazywanej analizą elektrotermiczną, czy też metody wyznaczania charakterystyk elementów półprzewodnikowych, nazywanych charakterystykami nieizotermicznymi, w odróżnieniu od charakterystyk izotermicznych określanych dla ustalonej temperatury (otoczenia), jest zróżnicowany. Metody te wymagają sformułowania i rozwiązania równań opisujących dwa rodzaje modeli: modelu elektrycznego w postaci zależności prądowo-napięciowo-temperaturowych z parametrami zależnymi od temperatury oraz modelu termicznego, stanowiącego zależność temperatury wnętrza elementu od wydzielanej mocy elektrycznej.

Z uwagi na sposób rozwiązywania równań opisujących obydwa rodzaje modeli, można wyróżnić dwie grupy metod analizy elektrotermicznej – metody wspólnych iteracji, w których modele elektryczny i termiczny rozwiązywane są jednocześnie oraz metody rozdzielonych iteracji, w których modele te rozwiązywane są naprzemiennie.

Spośród omawianych metod, wygodniejsze z punktu widzenia projektanta układów elektronicznych są metody rozdzielonych iteracji z uwagi na ich większą elastyczność, tzn. możliwość korzystania bezpośrednio z istniejących wbudowanych w programie SPICE modeli elementów oraz możliwość swobodnego formułowania postaci termicznego modelu elementu rozwiązywanego „na zewnątrz” programu SPICE.

Jednakże, zdaniem autorów znane literaturowe metody analizy elektrotermicznej oparte na koncepcji rozdzielonych iteracji nie odpowiadają w pełni potrzebom inżyniera-projektanta posługującego się programem SPICE. Przykładowo, przedstawione w pracach [1, 2, 3, 8] metody analizy wymagają znajomości wartości parametrów termicznego modelu elementu o stałych rozłożonych, takich jak np. przewodność cieplna, lokalizacja oraz rozmiary poszczególnych elementów konstrukcji mechanicznej przyrządu – na ogół niedostępnych dla rozważanej klasy użytkowników programu. Z kolei praca [2] dotyczy analizy d.c. z wykorzystaniem wcześniejszych wersji programu SPICE, w której istniała możliwość wprowadzenia tylko jednej wartości (takiej

samej dla wszystkich elementów) temperatury, natomiast w pracy [4] autorzy korzystają ze skupionego modelu termicznego elementu opisanego za pomocą całki splotu oraz przejściowej impedancji termicznej, a zastosowany sposób analizy (sekwencja analiz d.c.) nie pozwala na uwzględnienie inercji elektrycznej elementu i układu. Natomiast pominięcie ważnych fragmentów opisu procedury obliczeniowej w pracy [1] nie pozwala na implementację i realizację algorytmu przedstawionego w cytowanej pracy.

Wymienione powyżej mankamenty metod literaturowych skłoniły autorów do podjęcia próby opracowania algorytmu wyznaczania nieizotermicznych przebiegów czasowych prądów, napięć i temperatury wnętrza elementów półprzewodnikowych, opisanych skupionym modelem termicznym oraz wbudowanym w programie SPICE elektrycznym modelem izotermicznym takich elementów, opartego na metodzie rozdzielonych iteracji. Przedstawiony w rozdziale trzecim algorytm ITER stanowi modyfikację i rozszerzenie algorytmu przedstawionego w pracy [10], opartego na metodzie zaproponowanej przez Petegema [1] i pozwala na bezpośrednie wykorzystanie dostępnych w programie SPICE modeli elementów półprzewodnikowych. Zaproponowana modyfikacja obejmuje zastosowanie skupionego modelu termicznego elementu, opartego na koncepcji całki splotu wraz z tzw. algorytmami bezpamięciowymi [5, 11], implementację nowego algorytmu wyznaczania przebiegu mocy czynnej na podstawie zaciskowych przebiegów napięć i prądów [12] oraz autorskiej metody przewidywania i korekcy długości bieżącego przedziału czasowego analizy w programie SPICE.

2. POSTAĆ MODELI ELEMENTÓW PÓŁPRZEWODNIKOWYCH A PROBLEM MOCY CZYNNEJ

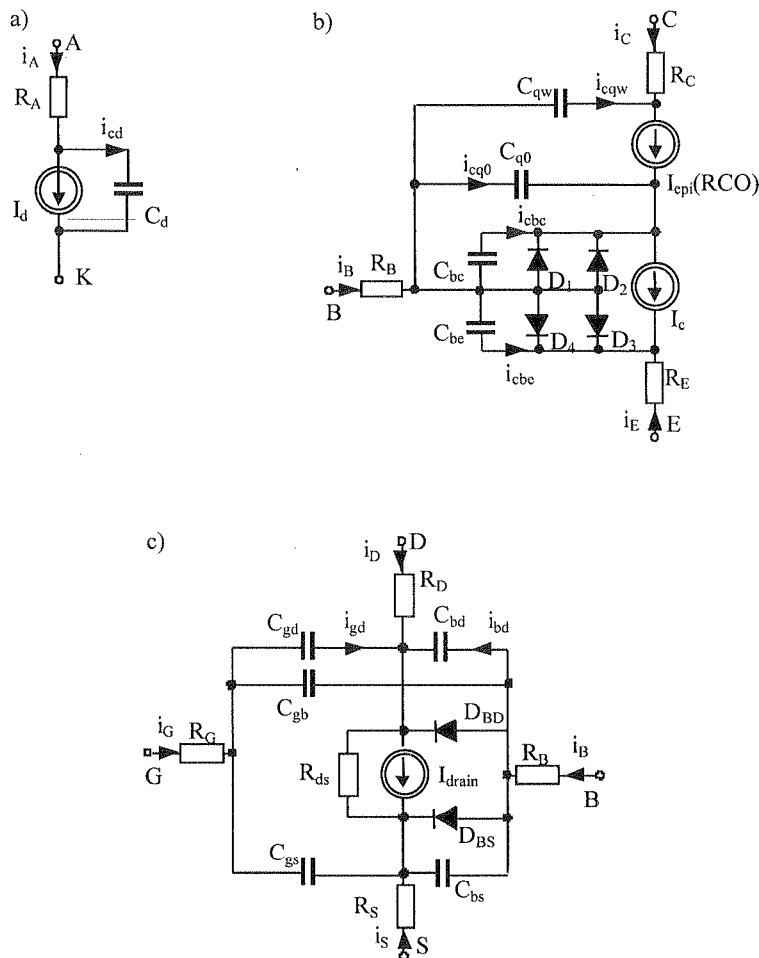
Użytkownik programu SPICE dysponuje wbudowanymi fizycznymi modelami podstawowych elementów półprzewodnikowych, wraz z wartościami ich parametrów dla poszczególnych typów. Na rys. 1 przedstawiono struktury takich modeli dla podstawowych elementów półprzewodnikowych – diody p-n, tranzystora bipolarnego oraz tranzystora MOSFET.

Istotnym problemem z punktu widzenia realizacji analizy elektrotermicznej za pomocą programu SPICE jest wyznaczenie wartości mocy czynnej w elemencie półprzewodnikowym. Moc czynna, z uwagi na istnienie elementów inercyjnych w modelu, różni się od mocy całkowitej – możliwej do wyznaczenia na podstawie prądów i napięć zaciskowych wyliczonych w danym kroku analizy. Z drugiej strony użytkownik programu SPICE nie ma dostępu do wewnętrznych zacisków rozważanego modelu, co uniemożliwia uzyskanie informacji o wartościach prądów gałęziowych związanych z elementami o charakterze rezystancyjnym i pojemnościowym. Dlatego pojawia się problem wiarygodnego wyznaczenia przebiegu mocy czynnej.

Zaproponowany przez autorów algorytm wyznaczania mocy czynnej dla tranzystora MOSFET [12] można łatwo uogólnić na przypadek modeli pozostałych elementów półprzewodnikowych dostępnych w programie SPICE. Zgodnie z tą ideą, moc czynną można wyrazić zależnością

$$P_{cz} = \sum_k i_k^2 \cdot R_{Sk} + \sum_k \left(i_k - \sum_j i_{ckj} \right) \cdot (u_{kj} - i_k \cdot R_{Sk} + i_j \cdot R_{Sj}) \quad (1)$$

gdzie i_k oznacza prąd wpływający do zacisku K-tego, R_{Sk} – rezystancję szeregową podłączoną do tego węzła, i_{ckj} – prąd płynący przez pojemność włączoną między zaciski K oraz J, natomiast u_{kj} to napięcie między tymi zaciskami.



Rys. 1. Modele podstawowych elementów półprzewodnikowych w programie SPICE:
a) diody p-n, b) tranzystora bipolarnego, c) tranzystora MOSFET

A zatem dla przedstawionych na rys. 1 modeli, zależność (1) ma następującą postać szczegółową:

- Dla diody p-n

$$P_{cz} = i_A^2 \cdot R_A + I_d \cdot (u_{AK} - i_A \cdot R_A) \quad (2)$$

- dla tranzystora bipolarnego, przy pominięciu efektu quasi-nasycenia (parametr $RCO = 0$)

$$P_{cz} = i_C^2 \cdot R_C + i_E^2 \cdot R_E + i_B^2 \cdot R_B + (i_{D4} + i_{D3}) \cdot (u_{BE} - i_B \cdot R_B + i_E \cdot R_E) + \\ + (i_{D1} + i_{D2}) \cdot (u_{BC} - i_B \cdot R_B - i_C \cdot R_C) + I_C \cdot (u_{CE} + i_E \cdot R_E - i_C \cdot R_C) \quad (3)$$

gdzie i_{D1} , i_{D2} , i_{D3} , i_{D4} oznaczają prądy płynące przez diody, odpowiednio D_1 , D_2 , D_3 , D_4 .

- dla tranzystora MOSFET

$$P_{cz} = i_D^2 \cdot R_D + i_S^2 \cdot R_S + i_G^2 \cdot R_G + i_B^2 \cdot R_B + I_{drain} \cdot (u_{DS} - i_D \cdot R_D + i_S \cdot R_S) + \\ + i_{DBS} \cdot (u_{BS} - i_B \cdot R_B + i_S \cdot R_S) + i_{DBD} \cdot (u_{BD} - i_B \cdot R_B + i_D \cdot R_D) + \\ + (u_{DS} - i_D \cdot R_D + i_S \cdot R_S)^2 / R_{ds} \quad (4)$$

gdzie i_{DBS} oraz i_{DBD} oznaczają prądy płynące przez diody, odpowiednio D_{BS} oraz D_{BD} .

Jak zatem widać, zaproponowane podejście wymaga znajomości funkcyjnego opisu wydajności poszczególnych źródeł sterowanych i charakterystyk statycznych diod, występujących w modelu rozważanych elementów oraz wartości parametrów tych modeli. Funkcyjny opis takich zależności dostępny jest w bogatej literaturze przedmiotu, np. w pracach [13, 14].

3. ALGORYTM OBLICZEŃ

Realizacja opracowanego przez autorów algorytmu ITER wyznaczania nieizotermicznych dynamicznych charakterystyk elementów półprzewodnikowych wymaga naprzemiennego wykonywania analiz stanów przejściowych w rozważanym elemencie za pomocą programu SPICE oraz obliczeń przebiegu mocy czynnej i temperatury wnętrza elementu półprzewodnikowego, a także formułowania odpowiednich plików wejściowych dla programu SPICE. Algorytm ITER zaimplementowano w opracowanym i uruchomionym przez autorów programie RANEL. Zgodnie z tym algorytmem należy:

1. Sformułować plik wejściowy PLIK.PAR dla programu RANEL, zawierający wartość temperatury otoczenia T_a , czas trwania analizy t_{END} , żadaną dokładność ε obliczeń temperatury wnętrza T_j oraz wartości parametrów przejściowej impedancji termicznej badanego elementu.
2. Sformułować plik wejściowy PLIK0.CIR dla programu SPICE, definiując wartość temperatury wnętrza elementu półprzewodnikowego za pomocą parametru $T_{ABS} = T_a$ oraz parametru DELAY, określającego czas opóźnienia w opisie źródeł

niezależnych, pobudzających badany element, który to parametr przyjmuje wartość początkową DELAY0.

3. Zdefiniować początkowe wartości parametrów analizy, oznaczających czas rozpoczęcia poprzedniego przedziału analizy $t_1 = 0$, czas zakończenia poprzedniego przedziału analizy $t_0 = 0$, czas zakończenia bieżącego przedziału analizy $t_{\text{KON}} = t_{\text{END}}$, licznik przedziałów $k = 0$.
4. Wyznaczyć współrzędne punktu pracy (u , i , T_j) badanego elementu, odpowiadające chwili $t = 0$. W tym celu należy:
 - a) za pomocą programu SPICE przeprowadzić izotermiczną analizę OP,
 - b) na podstawie wyliczonych wartości prądów i napięć zaciskowych elementu, wyliczyć wartość wydzielanej w nim mocy P ,
 - c) wyznaczyć wartość temperatury wnętrza T_j , wykorzystując model termiczny o postaci

$$T_j = T_a + R_{th} \cdot P \quad (5)$$

gdzie R_{th} oznacza rezystancję termiczną elementu,

- d) jeżeli różnica między wyznaczoną temperaturą wnętrza a temperaturą elementu w czasie analizy jest mniejsza od założonej dokładności ϵ , to należy przejść do realizacji punktu 5 algorytmu, w przeciwnym przypadku należy zrealizować punkt 4e,
- e) zmodyfikować w pliku wejściowym wartość temperatury elementu T_{ABS} i przejść do realizacji punktu 4a.
5. Przeprowadzić izotermiczną analizę stanów przejściowych (TRAN) za pomocą programu SPICE do czasu t_{KON} . Wyniki obliczeń w postaci tekstowej są zapisywane w pliku PLIKk.CSD.
6. Wyliczyć w programie RANEL przebieg mocy czynnej wydzielanej w badanym elemencie w przedziale czasu od t_0 do t_{KON} na podstawie odpowiedniego wzoru (wzór (2) dla diody, wzór (3) dla tranzystora bipolarnego lub wzór (4) dla tranzystora MOSFET) oraz danych zapisanych w pliku PLIKk.CSD. Wyniki obliczeń zapisywane są do pliku PLIKk.MOC.
7. Wyliczyć w programie RANEL przebieg temperatury wnętrza badanego elementu za pomocą algorytmu splotowego [5, 11], wykorzystując przebieg mocy z pliku PLIKk.MOC oraz przejściową impedancję termiczną opisaną zależnością [5]

$$Z(t) = R_{th} \cdot \left[1 - \sum_{i=1}^N a_i \cdot \exp\left(-\frac{t}{\tau_{thi}}\right) \right] \quad (6)$$

gdzie a_i jest współczynnikiem wagowym odpowiadającym i -tej termicznej stałej czasowej τ_{thi} , natomiast N – ilością termicznych stałych czasowych. Wówczas temperatura wnętrza elementu T_j może być opisana za pomocą sumy temperatury otoczenia T_a oraz nadwyżek temperatury ΔT_{ji} , związanych z i -tą termiczną stałą czasową.

$$T_j(t) = T_a + \Delta T_{j1}(t) + \Delta T_{j2}(t) + \dots + \Delta T_{jN}(t) \quad (7)$$

gdzie

$$\Delta T_{ji}(t) = \frac{R_{th} \cdot a_i}{\tau_{thi}} \int_0^t \exp\left(-\frac{t-v}{\tau_{thi}}\right) \cdot p(v) \cdot dv \quad (8)$$

Wyniki obliczeń zapisywane są w pliku PLIKk.TEM.

8. Przeanalizować przebieg temperatury wnętrza $T_j(t)$ zapisany w pliku PLIKk.TEM oraz wyznaczyć największą wartość czasu $t_{xk} > t_0$, dla której spełniony jest warunek

$$T_j(t_{xk}) - T_{_ABS} < \varepsilon \quad (9)$$

Jeżeli warunek (9) jest spełniony w całym przedziale $\langle t_0; t_{KON} \rangle$, wówczas należy przyjąć $t_{xk} = t_{KON}$.

9. Zapisać do pliku wynikowego PLIK.WYN przebiegi napięć i prądów, obliczone w punkcie 5 algorytmu, oraz przebieg temperatury wnętrza w przedziale od t_0 do $t_0 + t_{xk}$ obliczony w punkcie 7 algorytmu.
10. Podstawić $t_1 = t_0$, $t_0 = t_0 + t_{xk}$ oraz $k = k + 1$.
11. Sprawdzić, czy został spełniony warunek $t_0 < t_{END}$. Przy spełnieniu powyższego warunku należy kontynuować obliczenia (punkt 12 algorytmu), w przeciwnym razie należy zakończyć pracę algorytmu.
12. Wyznaczyć, za pomocą algorytmu zmiany kroku, przewidywaną wartość czasu zakończenia obliczeń w kolejnym kroku czasowym t_{KON} . W tym celu należy oszacować przyrosty składowych temperatur ΔT_i^k , związanych z poszczególnymi termicznymi stałymi czasowymi $\tau_{th1}, \tau_{th2}, \dots, \tau_{thN}$ w poprzednim (k-tym) przedziale analizy. Jeżeli zatem
 - $\Delta T_i^k \leq 0$, wówczas

$$t_{KONi} = t_0 - \tau_{thi} \cdot \ln\left(\frac{T_{ji}(t_0) - \varepsilon - R_{th} \cdot a_i \cdot P_{cz}(t_0)}{T_{ji}(t_0) - R_{th} \cdot a_i \cdot P_{cz}(t_0)}\right) \quad \text{dla } i = 1, 2, \dots, N \quad (10)$$

- $\Delta T_i^k > 0$, wówczas

$$t_{KONi} = t_x - \tau_{thi} \cdot \ln\left(1 - \frac{T_{ji}(t_0) - T_{jmin} + \varepsilon}{R_{th} \cdot a_i \cdot P_{cz}(t_0)}\right) \quad \text{dla } i = 1, 2, \dots, N \quad (11)$$

przy czym

$$t_x = \tau_{thi} \cdot \ln\left(\frac{T_{ji}(t_0) - T_j(t_1)}{R_{th} \cdot a_i \cdot P_{cz}(t_0) \cdot \left[\exp\left(-\frac{t_1}{\tau_{thi}}\right) - \exp\left(-\frac{t_0}{\tau_{thi}}\right)\right]}\right) \quad (12)$$

$$T_{jimin} = T_{ji}(t_1) - R_{th} \cdot a_i \cdot P_{cz}(t_0) \cdot \left[1 - \exp\left(-\frac{t_1 - t_x}{\tau_{thi}}\right) \right] \quad (13)$$

Do dalszych obliczeń należy przyjąć najmniejszą wartość t_{KONi} .

13. Sformułować plik wejściowy PLIKK.CIR dla programu SPICE, poprzez korektę pliku PLIK0.CIR, obejmującą:

- podstawienie w modelu elementu półprzewodnikowego $T_ABS = T_j(t_0)$,
- podstawienie w opisie wydajności każdego niezależnego źródła prądowego lub napięciowego, pobudzających badany element, $DELAY = DELAY0 - t_0$,
- wprowadzenie warunków początkowych odpowiadających chwili $t = t_0$,
- wprowadzenie zakresu analizy TRAN równego $t_{KON} - t_0$

14. Powtórzyć obliczenia od punktu 5 algorytmu.

Program RANEL umożliwia użytkownikowi wprowadzenie nazwy pliku wejściowego, wyznacza przebiegi mocy czynnej i temperatury wnętrza elementu, tworzy odpowiednie pliki wynikowe, modyfikuje zawartość pliku wejściowego dla programu SPICE, zgodnie z zasadami określonymi w punkcie 13 algorytmu ITER oraz cyklicznie uruchamia ten program.

Przy opracowywaniu algorytmu analizy przyjęto, że w chwili $t = 0$ występuje w analizowanym układzie stan termicznie ustalony, co oznacza, że temperaturę wnętrza elementu można opisać zależnością (5).

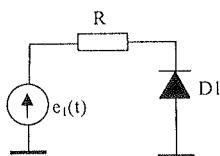
Z kolei, przy konstruowaniu algorytmu zmiany kroku obliczeń przyjęto, że w poszczególnych przedziałach obliczeń moc czynna jest opisana funkcją stałą, a przebieg temperatury wnętrza jest opisany za pomocą funkcji eksponencjalnych. Przy opisie przebiegu tych temperatur wykorzystywane są wartości temperatury na końcach dwóch ostatnich przedziałów analizy, wartość mocy czynnej, wydzielanej na końcu ostatniego przedziału oraz parametry modelu przejściowej impedancji termicznej. Długość przedziału obliczeń jest szacowana z nadmiarem, ponieważ przyjęto założenie, że przyrost temperatury wnętrza spowodowany przez każdą termiczną stałą czasową może być równy założonej dokładności wyznaczania temperatury wnętrza elementu.

Plik wejściowy dla programu SPICE PLIK0.CIR zawiera wyrażenie typu .MODEL dla elementu półprzewodnikowego, z wyspecyfikowaną wartością temperatury wnętrza elementu za pomocą parametru T_ABS, a postać pliku wynikowego jest określona, za pomocą wyrażenia .PROBE/CSDF z nazwami zaciskowych napięć i prądów elementu półprzewodnikowego.

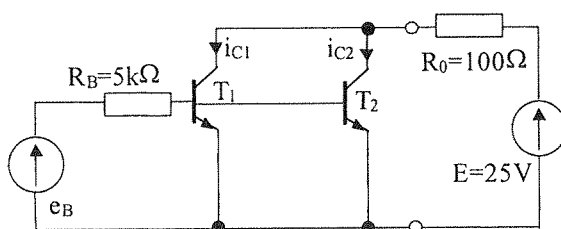
Występowanie w analizach stanów przejściowych stałych czasowych (elektrycznych i termicznych) o znacznie różniących się wartościach (nawet 10 rzędów) oraz szybko zmiennych sygnałów powoduje duże zmiany kroku obliczeń wyznaczanego automatycznie przez program SPICE. W skrajnym przypadku, wyznaczenie zbyt małej wartości kroku obliczeń może być powodem przerwania obliczeń. Tak więc skuteczna analiza elektrotermiczna wymaga starannego doboru wartości opcji analizy określających żadaną dokładność względną napięć i prądów (RELTOL), dokładność bezwzględną napięć i prądów (odpowiednio VNTOL oraz ABSTOL), a także parametru ITL4, określającego ilość dopuszczalnych iteracji w jednym kroku analizy stanów przejściowych.

4. WYNIKI OBLICZEŃ

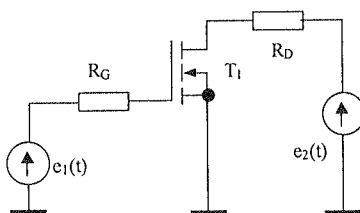
W celu oceny skuteczności i wiarygodności opracowanego algorytmu przeprowadzono symulacje charakterystyk dynamicznych rozważanych wcześniej elementów, pracujących w wybranych, prostych układach elektronicznych, przedstawionych na rys. 2,3,4. Wyniki obliczeń algorytmem ITER (oznaczone na rysunkach linią ciągłą) porównano z wynikami obliczeń, opartych na algorytmie wspólnych iteracji [5] (punkty), a także w celu ilustracji wpływu samonagrzewania na właściwości elementów półprzewodnikowych, zamieszczono dodatkowo wyniki odpowiadające warunkom izotermicznym (linie kreskowe). W analizach przyjęto wartość temperatury otoczenia $T_a = 300K$ oraz następujące wartości opcji w programie SPICE, określających dokładność obliczeń: $RELTOL = 2 \cdot 10^{-4}$, $VNTOL = 7 \cdot 10^{-7}$, $ABSTOL = 10^{-12}$ oraz maksymalną ilość iteracji dla jednego punktu analizy stanów przejściowych $ITL4 = 2000$.



Rys. 2. Układ z diodą przełączaną napięciowo



Rys. 3. Połączenie równoległe tranzystorów bipolarnych



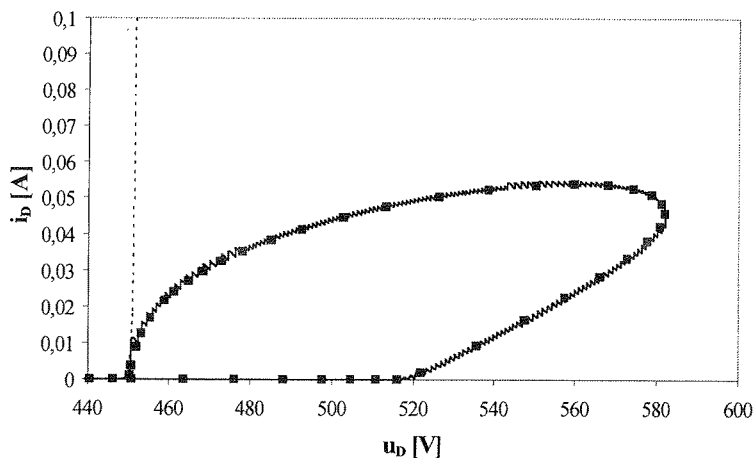
Rys. 4. Układ klucza równoległego z tranzystorem MOSFET

Na rys. 5 przedstawiono obliczone charakterystyki fazowe diody, pracującej w układzie z rys. 2, w którym przyjęto $R = 1 \text{ k}\Omega$ sinusoidalne pobudzenie ze źródła napięciowego $e_1(t)$ o częstotliwości 50Hz oraz amplitudzie 630V, zapewniającej wejście diody w zakres przebicia.

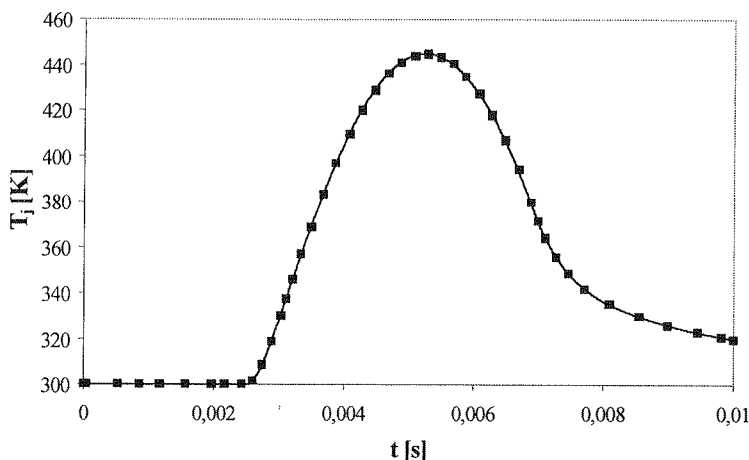
Przyjęto następujące wartości parametrów modelu diody (wymieniono tylko te parametry, których wartości różnią się od wartości domyślnych): $I_S = 26,06$ nA, $N = 2$, $BV = 450$ V, $IBV = 1$ mA, $NBV = 11,6$, $TBV1 = 2 \times 10^{-3}$ K $^{-1}$, $RS = 136$ m Ω , $TT = 100$ ns, $CJO = 100$ pF. Z kolei parametry modelu termicznego miały następujące wartości: $R_{th} = 73$ K/W, $a_1 = 0,905$, $a_2 = 0,045$, $a_3 = 0,05$, $\tau_{th1} = 170$ s, $\tau_{th2} = 4$ ms, $\tau_{th3} = 0,5$ ms.

Jak widać z rys. 5a oraz 5b, uzyskano dobrą zgodność wyników obliczeń elektrotermicznych, otrzymanych za pomocą obu algorytmów (krzywe są praktycznie nierozróżnialne), natomiast przedstawiona na rys. 5a charakterystyka izotermiczna (kreskowa linia pionowa) zdecydowanie różni się zarówno ilościowo, jak i jakościowo (brak pętli) od charakterystyk nieizotermicznych.

a)



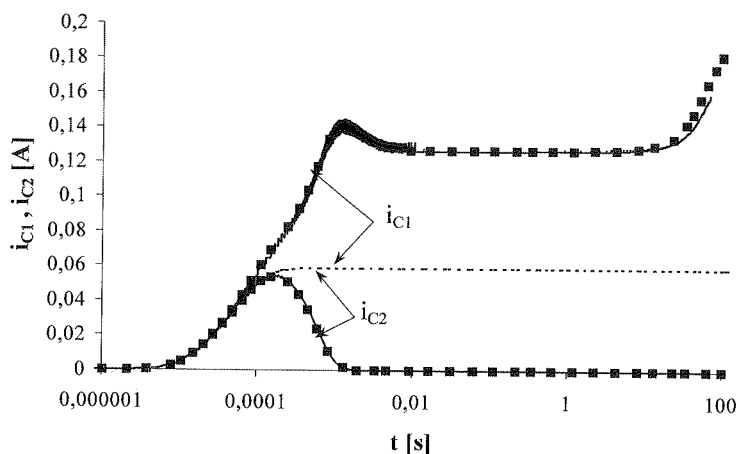
b)



Rys. 5. Izotermiczne oraz nieizotermiczne charakterystyki fazowe diody p-n przy polaryzacji zaporowej (a) oraz czasowy przebieg temperatury wnętrza (b)

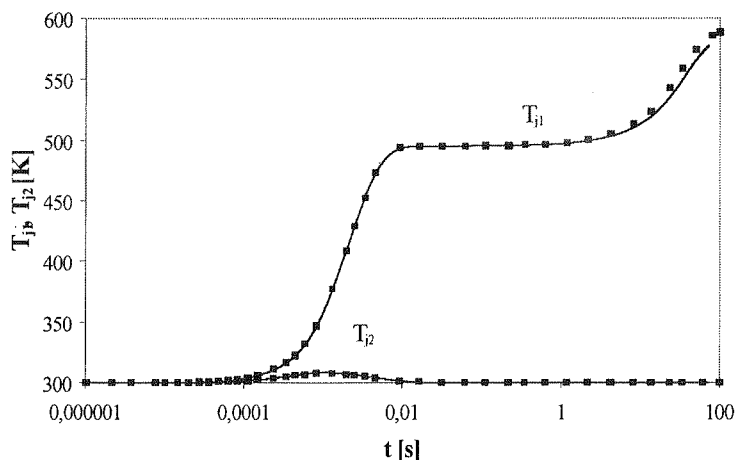
Wyniki symulacji charakterystyk połączonych równolegle tranzystorów bipolarnych sterowanych prądowo (rys. 3) przedstawiono na rys. 6. Przyjęto identyczne wartości parametrów elektrycznych obydwu tranzystorów (wymieniono tylko parametry, których wartości różnią się od wartości domyślnych): $I_S = 2,4536 \times 10^{-14}$ A, $BF = 130$, $BR = 2$, $R_B = 1$ m Ω , $E_G = 1,206$ V, $V_{AF} = 200$ V, $X_{TI} = 2$, $X_{TB} = 0$, $TF = 400$ ns, $TR = 1,5$ μ s, $C_{JE} = 30$ pF, $C_{JC} = 5$ pF. Tranzystory nie są sprzężone termicznie, natomiast różnią się tylko wartościami najkrótszej termicznej stałej czasowej τ_{th1} , która dla tranzystora T1 wynosi 2 ms, a dla tranzystora T2 – 5 ms. Pozostałe parametry modelu termicznego mają następujące wartości: $R_{th} = 250$ K/W, $a_1 = 0,5$, $a_2 = 0,5$, $\tau_{th2} = 100$ s (w zależności (6) przyjęto $N = 2$). Wartości elementów obwodu podano na rys. 4. Źródło napięciowe e_B wytwarza napięcie trapezoidalne o poziomach 0,1V oraz 5V, czasie narastania równym 10 μ s oraz czasie trwania 1500 s.

Jak widać z rys. 6, nawet nieznaczna różnica wartości jednej termicznej stałej czasowej powoduje przejmowanie całości prądu układu przez tranzystor T1, legitymujący się mniejszą wartością τ_{th1} . Z kolei, z analiz izotermicznych wynika, że prądy obu tranzystorów są jednakowe. Przebiegi temperatury wnętrza tranzystorów zamieszczono na rys. 7. Jak widać, temperatura wnętrza tranzystora T1 po przejściu całego prądu układu wzrasta do wartości przekraczającej wartość katalogową temperatury dopuszczalnej, mniejszej typowo poniżej 200°C.



Rys. 6. Wyniki analiz układu połączenia równoległego tranzystorów bipolarnych

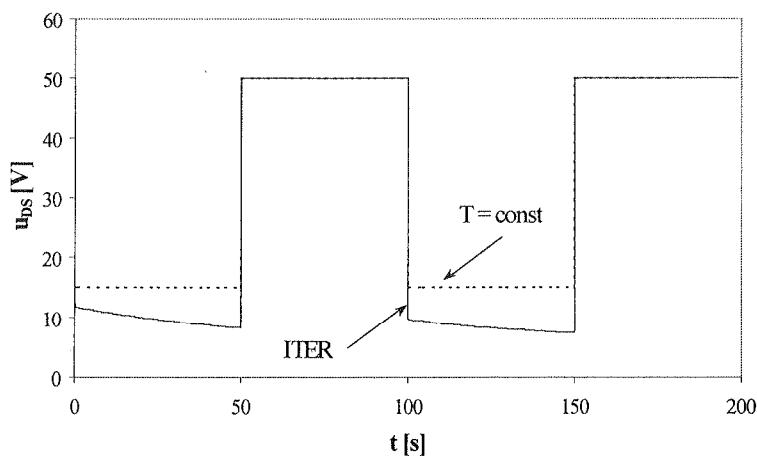
Z kolei, na rys. 8 przedstawiono charakterystyki tranzystora MOS, pracującego w układzie klucza równoległego (rys. 4), w którym przyjęto pobudzenie – źródłem $e_1(t)$ w postaci ciągu impulsów trapezoidalnych o poziomach 0 oraz 3,9 V, czasie trwania 50 s, okresie powtarzania 100 s oraz czasach narastania i opadania równych 100 ns. Pozostałe elementy obwodu mają następujące wartości: $e_2(t) = 50$ V, $R_G = 1$ k Ω , $R_D = 150$ Ω . Wartości parametrów elektrycznych modelu tranzystora IRF 740 zaczerp-



Rys. 7. Obliczone przebiegi temperatur wnętrza tranzystorów T_1 oraz T_2 w układzie z rys. 3

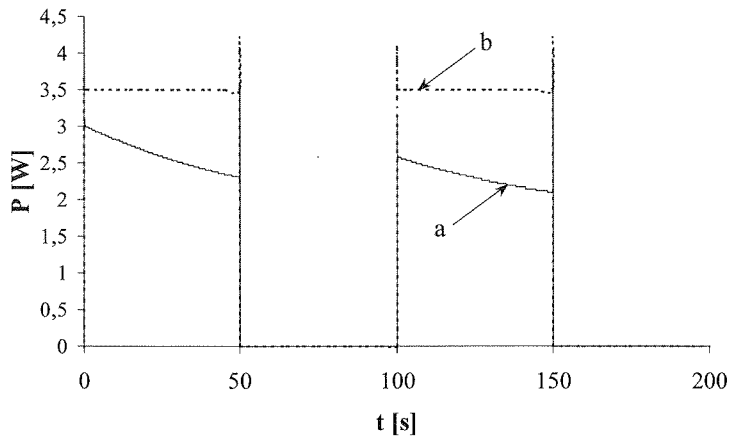
nięto z biblioteki PWRMOS.LIB, natomiast parametry modelu termicznego mają następujące wartości: $R_{th} = 50 \text{ K/W}$, $a_1 = 0,8$, $a_2 = 0,15$, $a_3 = 0,05$, $\tau_{th1} = 100 \text{ s}$, $\tau_{th2} = 0,1 \text{ s}$, $\tau_{th3} = 5 \text{ ms}$.

Jak wynika z rys. 8, nieizotermiczne charakterystyki dynamiczne tranzystora IRF740 (przebiegi napięcia dren-źródło), otrzymane za pomocą algorytmu ITER, w stosunku do swoich odpowiedników izotermicznych w stanie włączenia znacznie się różnią nawet o 40%.



Rys. 8. Czasowe przebiegi napięcia dren-źródło

Z kolei, na rys. 9 przedstawiono przebiegi mocy wydzielanej w tranzystorze IRF 740 uzyskane z wykorzystaniem algorytmu ITER (krzywa a) oraz wyznaczone na podstawie analizy izotermicznej (krzywa b) widać, że przebieg mocy odpowiadający



Rys. 9. Czasowe przebiegi mocy wydzielanej w tranzystorze IRF 740

warunkom izotermicznym znacznie odbiega od przebiegu uzyskanego za pomocą algorytmu ITER, a różnice między nimi przekraczają nawet 40%.

Na rys. 8 i 9 nie zamieszczono wyników analiz elektrotermicznych metodą wspólnych iteracji, ponieważ autorzy nie dysponują dostatecznie dokładnym elektrotermicznym, dynamicznym, skupionym modelem tego elementu.

5. UWAGI KOŃCOWE

W pracy przedstawiono nowy algorytm wyznaczania nieizotermicznych charakterystyk dynamicznych elementów półprzewodnikowych metodą rozdzielonych iteracji. Zaproponowany algorytm pozwala na bezpośrednie wykorzystanie w analizie elektrotermicznych stanów przejściowych, wbudowanych modeli elementów półprzewodnikowych wraz z dostępnymi w odpowiednich bibliotekach wartościami ich parametrów. Zamieszczone w pracy porównanie wyników obliczeń charakterystyk dynamicznych rozważanych wcześniej elementów, pracujących w wybranych, prostych układach elektronicznych, uzyskane za pomocą algorytmu ITER oraz algorytmu wspólnych iteracji [5], świadczy o wiarygodności i poprawności działania algorytmu ITER.

Nieznaczne modyfikacje algorytmu pozwoliłyby na uwzględnienie również wzajemnych sprzężeń termicznych pomiędzy elementami półprzewodnikowymi, co byłoby istotne przy badaniu właściwości elementów półprzewodnikowych, umieszczonych w układach scalonych lub na wspólnym radiatorze.

6. BIBLIOGRAFIA

1. W. Petegem, B. Geeraerts, W. Sansen, B. Graindourze: *Electrothermal Simulation and Design of Integrated Circuits*. IEEE Journal on Solid-State Circuits, Vol. 29, No. 2, 1994, p. 143.

2. W. Petegem, D. Wachter, W. Sansen: *Electrothermal Simulation of Integrated Circuits*. 6-th IEEE SEMI-THERM, Phoenix 1990, p. 70.
3. A. Kos: *Modelowanie hybrydowych układów mocy i optymalizacja ich konstrukcji ze względu na rozkład temperatury*. Kraków, Wydawnictwo AGH, 1994.
4. W. Janke, W. Pietrenko: *SPICE-aided Simulation of Electro-thermal Interactions in Transient States*. XVI Nat. Conference Circuit Theory and Electronic Circuits, Kołobrzeg 1993, Vol. 2, p. 290.
5. J. Zarębski: *Modelowanie, symulacja i pomiary przebiegów elektrotermicznych w elementach półprzewodnikowych i układach elektronicznych*. Prace Naukowe Wyższej Szkoły Morskiej w Gdyni, Gdynia, 1996.
6. E. Schurack, T. Latzel, W. Rupp, A. Gotwald: *Nonlinear Effects in Transistors Caused by Thermal Power Feedback: Simulation and Modeling in SPICE*. IEEE Int. Symp. on Circuits and Systems – ISCAS'92, San Diego 1992, p. 879.
7. K. Zamłyński: *Modelowanie termiczne sprzężenia zwrotnego w monolitycznych układach scalonych przy pomocy programu SPICE-2*. Rozprawy Elektrotechniczne, 1990, t. 35, z. 3, s. 673.
8. A. Napieralski: *Komputerowe projektowanie układów półprzewodnikowych mocy ze szczególnym uwzględnieniem ich właściwości termicznych*. Zeszyty Naukowe Politechniki Łódzkiej, 1988, nr 562.
9. K. Górecki: *Elektrotermiczny makromodel tranzystora Darlingtona do analizy układów elektronicznych*. Praca doktorska, Politechnika Gdańska, 1999.
10. J. Zarębski, K. Górecki, P. Jasicki: *Modelowanie nieizotermicznych charakterystyk dynamicznych tranzystora VDMOS w programie SPICE*. XXIV Międzynarodowa Konferencja z Podstaw Elektrotechniki i Teorii Obwodów IC-SPETO 2001, Ustroń, 2001, Vol. 2, s. 465.
11. J. Zarębski: *The Time Dependence Calculations of the Junction Temperature of Semiconductor Devices*. International Journal of Microcomputer Applications, Vol. 13, No. 2, June 1994, p. 62.
12. K. Górecki, J. Zarębski: *Nowy algorytm wyznaczania mocy czynnej tranzystora MOS w programie SPICE*. Kwartalnik Elektroniki i Telekomunikacji, t. 47, Nr 3, 2001, s. 373.
13. J. Izydorczyk: *Pspice. Komputerowa symulacja układów elektronicznych*. Gliwice, Helion 1993.
14. P. Antognetti, G. Massobrio: *Semiconductor Devices Modeling with SPICE*. New York, McGraw-Hill Book Company, 1988.

J. ZARĘBSKI, K. GÓRECKI

A NEW ALGORITHM OF SEMICONDUCTOR DEVICES NONISOTHERMAL DYNAMIC CHARACTERISTICS CALCULATIONS IN SPICE

Summary

This paper deals with the problem of SPICE modelling of the dynamic characteristics of semiconductor devices with the thermal phenomena taken into account. Such characteristics, called here as the nonisothermal ones differ from the isothermal characteristics defined at the constant value of temperature and being calculated directly from SPICE. However, to get the nonisothermal characteristics of semiconductor devices, a new algorithm ITER of the thermal and electrical device models solution has been proposed. In this algorithm the new calculation method of the real power dissipated in semiconductor devices is also proposed. The real power values are indispensable to compute the junction temperature with the use of the thermal model at the successive time steps of calculations. A very well agreement between the calculation results of the nonisothermal characteristics of the selected semiconductor devices performed by the use of ITER and the known algorithm of the common iterations revealed the usefulness and credibility of the new algorithm.

Keywords: SPICE, electrothermal analysis, modelling of semiconductor devices

PWM controller electrothermal macromodel dedicated for SPICE

JANUSZ ZARĘBSKI, KRZYSZTOF GÓRECKI

Department of Marine Radioelectronics, Gdynia Maritime Academy

81-225 Gdynia, ul. Morska 83

e-mail: zarebski@wsm.gdynia.pl

gorecki@wsm.gdynia.pl

Otrzymano 2001.07.20

Autoryzowano 2001.09.03

The paper deals with the problem of electrothermal modelling of PWM controllers for SPICE. A new form of the electrothermal macromodel of SG3525A controller including the dependence of the output pulse width and of the supply current on the voltage error, the supply voltage and the junction temperature is proposed. The parameter values of macromodel have been obtained from measurements. To illustrate the correctness and usefulness of the macromodel, some results of measurements and calculations of SG3525A controller operating in the catalogue test circuit are given as well.

Keywords: PWM controller, selfheating, electrothermal models

1. INTRODUCTION

A very important class of the contemporary power electronic circuits are switching power electronic converters (SPEC), as e.g. inverters, pulse stabilisers etc. While considering typical operating conditions of semiconductor devices – especially the power switching devices and the other components (subcircuits) used in SPEC, the thermal phenomena as the selfheating and mutual thermal interactions should be taken into account. They influence the important parameter values and the reliability of SPEC considerably. As a result of the thermal phenomena, the device junction (inner) temperature T_j differs, sometimes considerably, from the ambient temperature T_a .

In the computer-aided design and analysis of SPEC the proper and accurate models of the whole circuits or theirs components including, among others, the thermal phenomena are needed. Such models, called electrothermal ones, e.g. [1], are formulated first of all for the power devices as the power BJTs, VDMOS, thyristors, etc.

However, there are no trends to formulate the electrothermal models of low-power or signal devices and circuits operating in SPEC. The authors proved, that the temperature

affects strongly the performance of a class of PWM controllers, especially its parameter values and characteristics – called nonisothermal ones, e.g. [10]. The investigation results of the ambient temperature influence on the PWM controller along with the simple isothermal macromodel of this device was presented in [2], whereas the steady-state electrothermal macromodel of the considered device was proposed in [3].

In the paper, being the continuation of the previous one [3], the electrothermal macromodel of the PWM SG3525A controller dedicated for SPICE, appropriated to the electrothermal transient analysis is proposed. The results of measurements and calculations illustrating the correctness and usefulness of the macromodel, are presented as well. The indispensable macromodel parameter values have been obtained from measurements of the proper isothermal characteristics.

In Sec. 2 the construction and principle of operation of the considered controller is described, whereas in Sec. 3 the structure of the new electrothermal macromodel of SG3525A controller is discussed. In Sec. 4 a new formal descriptions of the controller supply current and the output pulses time duration are proposed. The results of the measurements and analysis being of the confirmation of the macromodel correctness are presented in Sec. 5.

2. THE STRUCTURE OF SG3525A CONTROLLER

The main task of PWM controllers being one of the most important part of the each pulse stabiliser is generation of the rectangular waves of the proper pulse-duty factor controlling of semiconductor power switches in a way assuring a fixed value of the output voltage in the case of changing of the output current, the input supply voltage and the temperature. The SG3525A controller electrical block diagram is shown in Fig. 1. As seen, the considered controller demands the use of the small number of the outside components indispensable to ensure its operating conditions and the working parameter values of the regulation.

Shaped properly the output pulses of the time duration t_w , proportional to the difference of the error voltage V_{err} and the reference voltage V_{ref} available at the input of the error amplifier (EA), existing in the output of the comparator CMP, excite the output circuit (OC) of the demand current efficiency. The output circuit allows to get two synchronised pulse rectangular trains shifted to each other by 180° .

The frequency of the pulse rectangular trains can be controlled in the range from 100 Hz to 200 kHz by the external components R_T , C_T , whereas the deadtime t_d value is determined by the resistor R_D . The value of the supply voltage (V_{SUP}) should be selected in the range from 8 V to 35 V.

The p-n junction represented by the diode D built-in the oscillator block (OSC) was selected by the authors to measure the transient thermal impedance of the controller by means of the method based on the other one – destined for this controller thermal resistance measurements [4].

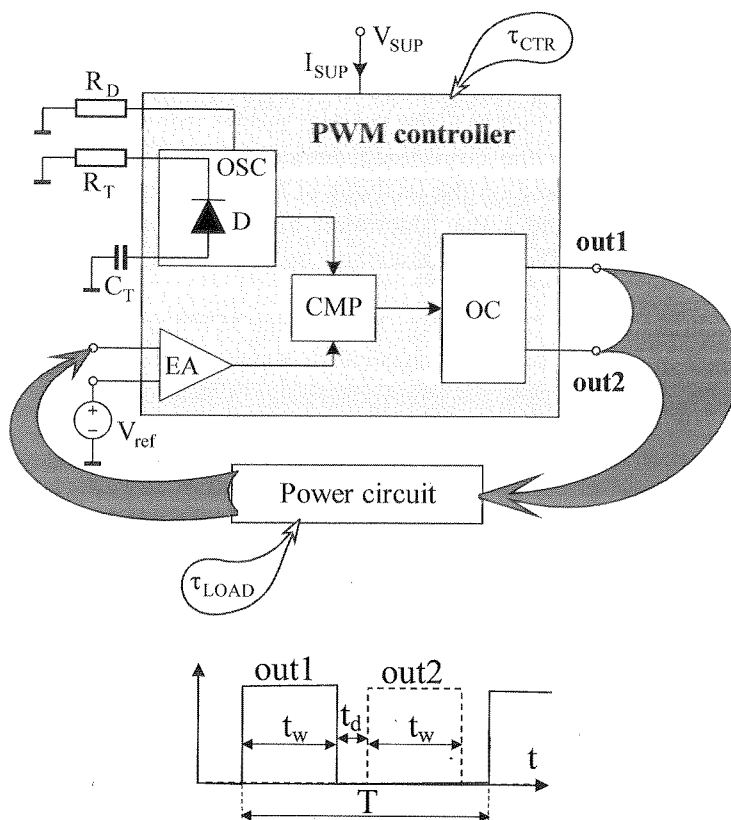


Fig. 1. The block diagram of SG3525A controller

The very important role, from point of view of the proposed model and the analysis results, plays the relationship between the electrical time constants τ_{CTR} and τ_{LOAD} of the controller and the whole controller load represented by the power circuit, respectively.

Because typically (in the real case) the inequality $\tau_{LOAD} \gg \tau_{CTR}$ is fulfilled, then one can assume V_{err} to be constant during the time of existing of the output signals.

3. THE MACROMODEL FORM

The main task to be performed by authors at the stage of the macromodel conception was to create it in the simplest form acceptable by SPICE. Moreover, this macromodel was designed to ensure the satisfying representation of the shapes and the parameter values of the output signals, in the all catalogue ranges of change of the signal frequency, supply voltage and the circuit inner temperature.

According to the principles of electrothermal model formulation [5], the worked out macromodel, showed in Fig. 2, consists of three essential blocks (A, B and C), representing the electrical model along with the parameters dependent on temperature (block A), the thermal model (block B) and the model of the real power dissipated in the considered device (block C). The structure of the block A has been created based on the analysis of PWM controller working principle described in Sec. 2. To model the functional blocks distinguished in Fig. 1 the controlled sources of the efficiency described by the set of functions available in SPICE is used.

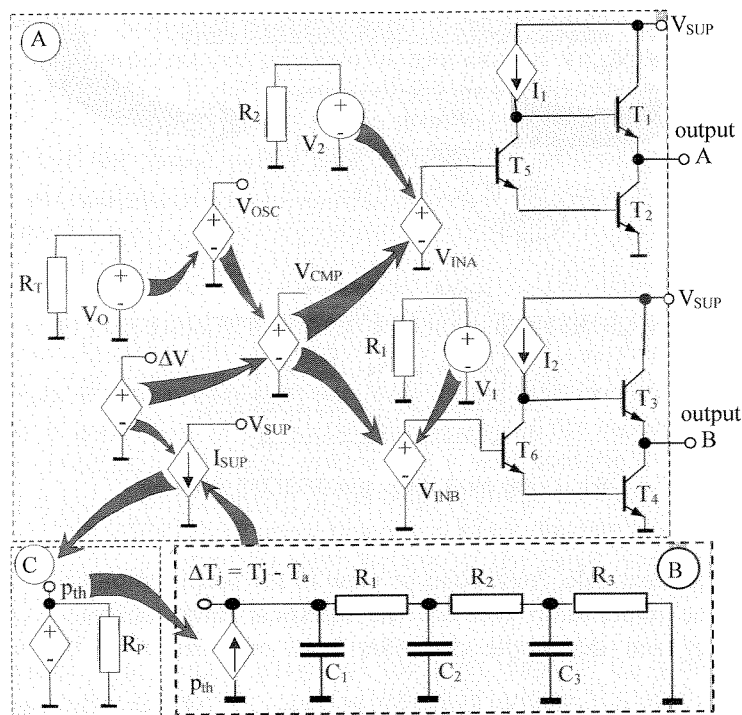


Fig. 2. The electrothermal macromodel of SG3525A controller

The output circuit is realized as Totem-Pole configuration (see block A in Fig. 2), consisting of two pairs of BJTs T_1 - T_2 and T_3 - T_4 , which are controlled by rectangular wave from sources V_{INA} and V_{INB} . These sources are connected to the base terminals of transistors T_5 and T_6 . In this macromodel, BJTs are described by the SPICE built-in isothermal model.

The source of the output voltage ΔV represents the error amplifier. Its output value is proportional to the difference of the error and reference voltages. Moreover, because for

$\Delta V < \Delta V_{\min}$ we get $t_w = 0$, whereas for $\Delta V > \Delta V_{\max}$ there is $t_w = t_{w\max} = \text{const}$, therefore the output value of this source can be described by

$$\Delta V = \text{LIMIT}(V_{\text{err}} - V_{\text{ref}}, \Delta V_{\min}, \Delta V_{\max}) \quad (1)$$

where according to SPICE User's Guide [6]

$$\text{LIMIT}(x, x_{\min}, x_{\max}) = \begin{cases} x_{\min} & \text{for } x < x_{\min} \\ x & \text{for } x \in (x_{\min}, x_{\max}) \\ x_{\max} & \text{for } x > x_{\max} \end{cases} \quad (2)$$

As it results from the PWM principle of operation [7, 8], the oscillator represented here by V_{OSC} source should generate the triangle pulses of the period time equal to $T/2$ and the slope γ equal to that of the isothermal characteristics $V_{\text{err}}(t_w)$ in the linear range.

As at $t = 0$, $V_{\text{err}} \neq 0$, therefore the time-independent parameter β has to be added to the linear dependence describing of V_{OSC} output value. Moreover, as it results from [2], for $V_{\text{err}} < V_{\text{errmin}}$, the time of the pulse duration $t_w = 0$ and on the other hand for $V_{\text{err}} > V_{\text{errmax}}$, $t_w = \text{const}$, therefore two parameters – V_{errmin} and V_{errmax} , limiting the linear range of $V_{\text{err}}(t_w)$ characteristic, have to be included in the model. Finally, the following description of V_{OSC} output value can be formulated in the form

$$V_{\text{OSC}} = \text{LIMIT}(\gamma \cdot (t + \beta), V_{\text{errmin}}, V_{\text{errmax}}) \quad (3)$$

where

$$\beta = t_w - \gamma^{-1} \cdot V_{\text{err}} \quad (4)$$

Since the slope of the triangle pulses can be changed in the analysis run, therefore the standard PULSE function cannot be used for definition of the output value V_{OSC} . This problem has been solved by the use of the additional artificial voltage source described by

$$V_0 = \frac{2 \cdot t}{T} - \text{trunc}\left(\frac{2 \cdot t}{T}\right) \quad (5)$$

Now, one can introduce V_0 variable instead to variable t in Eq. (3).

The comparator (CMP) is represented by the source V_{CMP} described by the equation

$$V_{\text{CMP}} = \text{IF}(V_{\text{OSC}} - \Delta V < 0, 0, 1) = \begin{cases} 0 & \text{for } V_{\text{OSC}} > \Delta V \\ 1 & \text{for } V_{\text{OSC}} < \Delta V \end{cases} \quad (6)$$

As the output signal of V_{INA} and V_{INB} sources are shifted by 180° , therefore V_{CMP} voltage has to control the bases of T_5 and T_6 transistors every other period. The output signals of V_{INA} and V_{INB} sources are of the form

$$V_{\text{INA}} = V_1 \cdot V_{\text{CMP}} \cdot V_{\text{SUP}} \quad (7a)$$

$$V_{\text{INB}} = V_2 \cdot V_{\text{CMP}} \cdot V_{\text{SUP}} \quad (7b)$$

where the output signals of V_1 and V_2 of the form of the rectangular waves shifted to each other by 180° are described by the period time equal to T , the time duration equal

to $T/2$ and the voltage levels equal to 0 and 1 V, respectively. These two voltage signals are synchronous to voltage V_0 .

As results from the previous investigations of the considered controller [2, 3] the time pulse duration t_w (see Eq. (4)) and the current I_{SUP} consumed by the controller depend strongly on the error voltage V_{err} , supply voltage V_{SUP} , junction temperature T_j and the output signal frequency f_w . This problem is discussed in Sec.4 in detailed way.

In turn, the thermal model (block B) describes the dependence of the rise of the junction temperature ΔT_j on the dissipated power p_{th} , what can be described by the convolution integral of the form

$$\Delta T_j = T_j - T_a = \int_0^t Z'(t-\nu) \cdot p_{th}(\nu) \cdot d\nu \quad (8)$$

where $Z'(t)$ denotes the time derivation of the transient thermal impedance $Z(t)$, typically described by means of a sum of exponential functions of the form [5]

$$Z(t) = R_{th} \cdot \left[1 - \sum_{i=1}^N a_i \cdot \exp\left(-\frac{t}{\tau_{thi}}\right) \right] \quad (9)$$

In Eq. (9) R_{th} is the thermal resistance, τ_{thi} – i -th thermal time constant, a_i are the balance factors of the sum equal to 1 and N is the number of the thermal time constants.

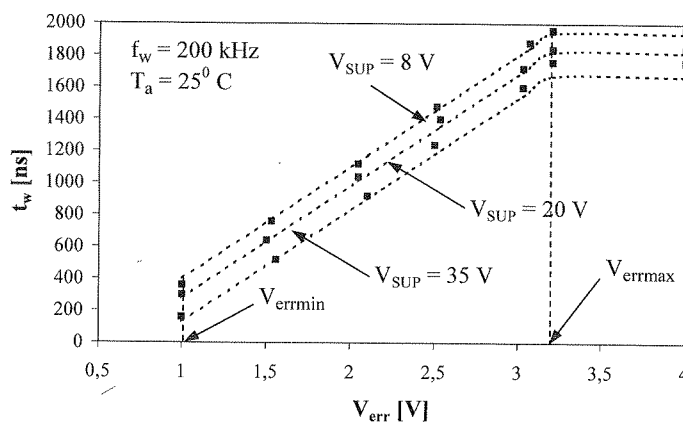
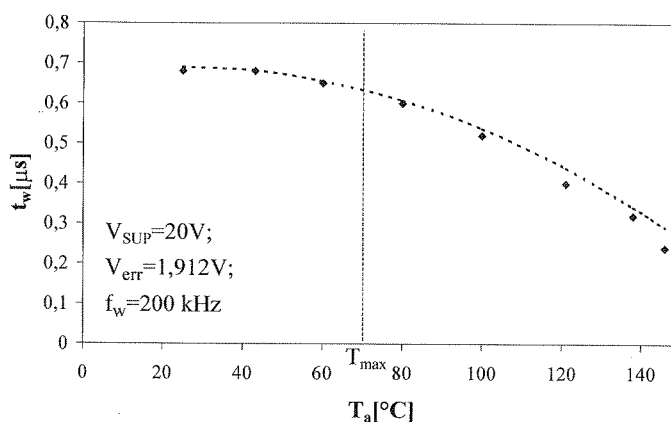
Due to in SPICE analysis any variable value from the last step of time is not available for a user, thus the thermal model is realized in the form of the controlled voltage source p_{th} connected with RC Cauer network, in which: $R_i = R_{th} \cdot a_i$ and $C_i = \tau_{thi}/R_i$. Indispensable thermal model parameter values can be calculated after the method described in [5] on the base of the thermal impedance measurements.

At last, the dependence of the power p_{th} on the controller supply voltage and current (block C) is represented by the circuit of the parallel connected resistor and the controlled voltage source of the output value equal to the average value of the V_{SUP} and I_{SUP} product.

4. A NEW DESCRIPTION OF THE SUPPLY CURRENT AND THE OUTPUT PULSES TIME DURATION

The full form of the proposed macromodel demands to determine the functional dependencies of the time pulse duration t_w and the current consumed by controller I_{SUP} depended strongly on the error voltage V_{err} , the supply voltage V_{SUP} , the junction temperature T_j and the output signal frequency f_w . To formulate such description three types of the isothermal characteristics have been measured:

- The dependence of the output pulse time duration t_w on the error voltage and the supply voltage,
- The dependence of the time pulse duration t_w on the ambient temperature T_a at the supply voltage $V_{SUP} = \text{const}$ and the error voltage $V_{err} = \text{const}$,

Fig. 3. The dependence of t_w on V_{err} and V_{SUP} Fig. 4. The dependence of t_w on T_a

- The dependence of the current consumed by controller I_{SUP} on the error voltage V_{err} for selected values of the supply voltage V_{SUP} and the ambient temperature T_a . The measured isothermal characteristics (point), are shown in Figs 3 ÷ 5.

On the base of the measured results (Figs 3 ÷ 5) the searched dependencies have been proposed as follows.

$$t_w = \frac{V_{err}}{\gamma \cdot f_w \cdot T_{w0}} + b + c \cdot (V_{SUP} - U_{SUP0}) + d \cdot (T_x - T_0)^2 \quad (10)$$

$$I_{SUP} = I_{SUP01} + e \cdot (V_{SUP} - V_{SUP0}) + I_{SUP02} \cdot \sqrt{T_{w0} \cdot f_w} \cdot (1 + f \cdot (T_x - T_0)) \cdot \\ \cdot LIMIT(1000 \cdot (V_{err} - V_{err0}), 0, 1) + (g + h \cdot (V_{SUP} - V_{SUP0}) + i \cdot (T_x - T_0)) \cdot \\ \cdot LIMIT((V_{err} - V_{err0}), 0, 2.5) \quad (11)$$

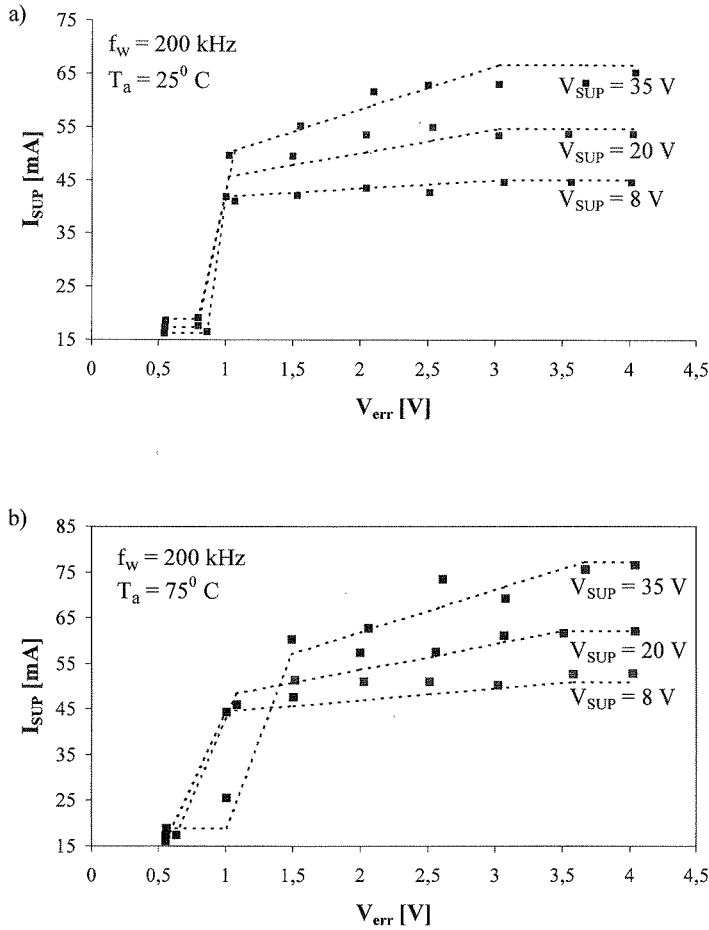


Fig. 5. The dependence of I_{SUP} on V_{err} at the selected values of V_{SUP} at two T_a values

where T_x denotes the ambient temperature T_a for the isothermal analysis and the junction temperature T_j for the electrothermal one, T_0 is the reference temperature and b , c , d , V_{SUP0} , I_{SUP01} , I_{SUP02} , T_{w0} , V_{err0} , e , f , g , h , i are the macromodel parameters.

The proposed dependencies given by Eqs. (10) and (11) are of the form of the formal model including fifteen parameters. To get the good matching between measurements (isothermal characteristics) and analytical curves (dashed lines), the following values of the model parameters have been found: $\gamma = 1.4286$ V/ μs , $b = -300$ ns, $c = -10$ ns/V, $V_{SUP0} = 8$ V, $d = 30$ ps/ K^2 , $e = 0.3$ mA/V, $I_{SUP01} = 16.2$ mA, $I_{SUP02} = 25.7$ mA, $f = 2$ $\mu\text{A/K}$, $T_0 = 298$ K, $V_{err0} = 0.95$ V, $g = 1.5$ mS, $h = 0.25$ mA/ V^2 , $i = 20$ $\mu\text{A/K}$, $T_{w0} = 5$ μs , $V_{err\min} = 1$ V, $V_{err\max} = 3.2$ V.

5. VERIFICATION OF THE MACROMODEL

The macromodel was implemented to SPICE with the use of ABM option. To confirm the correctness of the macromodel some analysis and measurements of the electrothermal performance of SG3525A controller, operating in the catalogue test circuit [7] (Fig.6) have been carried out. In this circuit two resistors R_{01} and R_{02} between the output terminals and the supply voltage represent the whole load of the controller. The value of R_D resistor is equal to zero, whereas R_{X1} , R_{X2} , R_{X3} resistors enable to change the error voltage V_{err} .

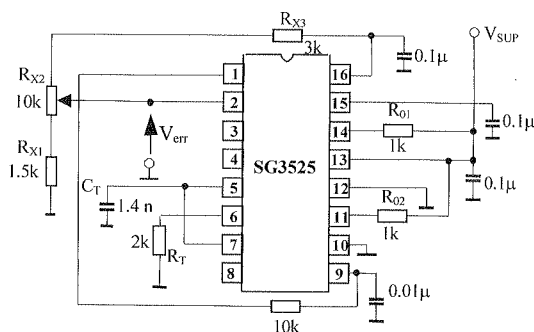


Fig. 6. SG3525A test circuit

SG3525A controller electrothermal and isothermal transients analyses, for selected values of error voltage V_{err} , supply voltage V_{SUP} and output signal frequency f_w have been carried out. In these analyses the fixed values of error voltage and supply voltage have been taken into account.

To make shorter the time of calculations, the all thermal time constants τ_{thi} and the all balance factors a_i have to be assumed equal to $10 \mu s$ and equal to 0.333, respectively. The value of the thermal resistance equal to $50 K/W$, taken into account in analyses, has been measured after the method described in [4]. Then the values of RC components (see Fig. 2) are: $R_1 = R_2 = R_3 = 16.67 \Omega$, $C_1 = C_2 = C_3 = 600 nF$.

To illustrate the influence of the selfheating on the controller electrothermal transients, in Fig. 7 the relative differences between the isothermal and nonisothermal time duration t_w at the selected periods of output signal, is presented. It is seen, that the selfheating changes considerably the time duration of the signals, and the differences between isothermal and nonisothermal case can exceed even 60%.

Very often the real characteristics of the controller at the steady-state are one of the most important information for engineers. Therefore, the next examples illustrate the calculated and measured characteristics of SG3525A controller at the thermal steady-state. Of course, to get the further presented $t_w(V_{err})$, $P(V_{err})$ and $\Delta T_j(V_{err})$ characteristics, the full transient analysis had to be performed for the each point on these characteristics.

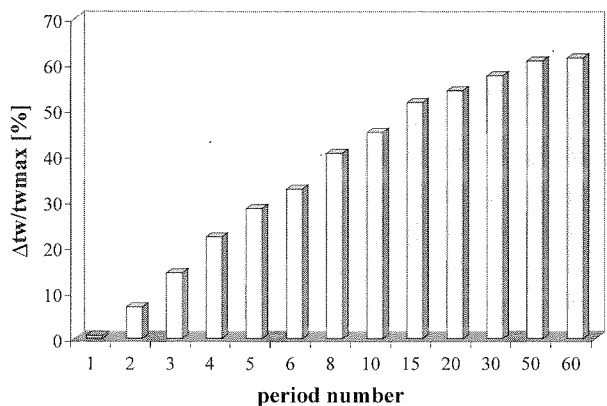


Fig. 7. The time dependence of the difference between the isothermal and nonisothermal time duration t_w , normalised to value of t_w at steady-state

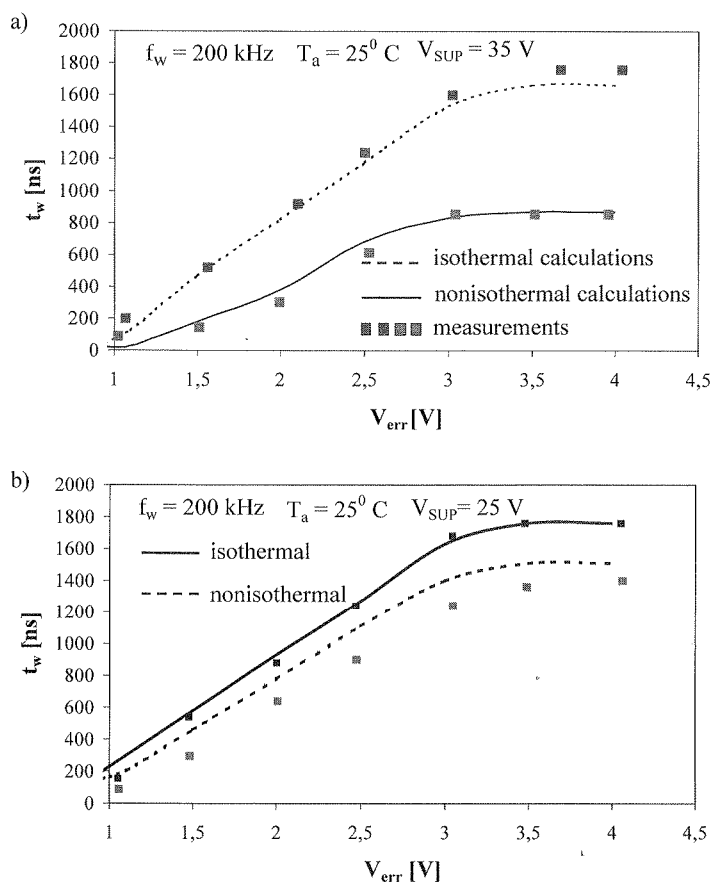


Fig. 8. The isothermal and nonisothermal dependencies of t_w on V_{err} at two V_{SUP}

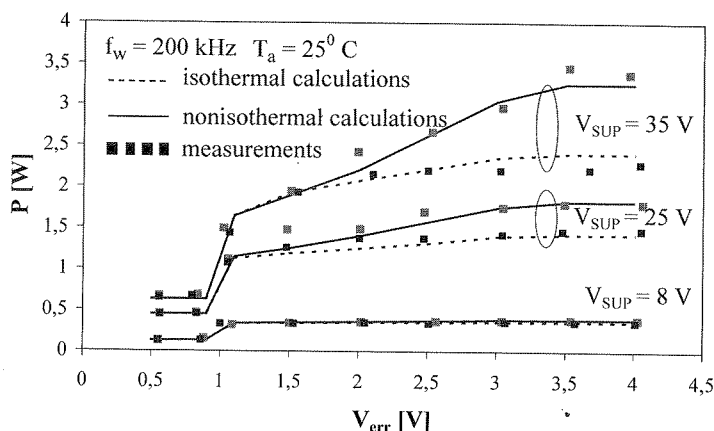


Fig. 9. The isothermal and nonisothermal dependencies of the power on the voltage error

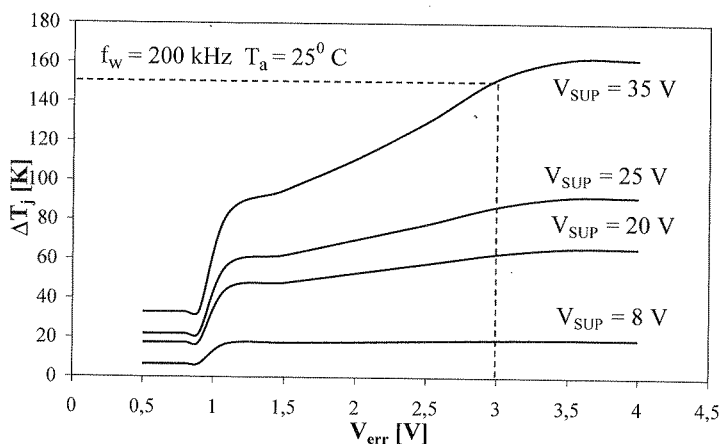


Fig. 10. The dependence of rise of the controller junction temperature over the ambient one on the voltage error

In Fig. 8 the dependencies of the nonisothermal (at the steady state) and for comparison, the isothermal one, the pulse time duration on the error voltage V_{err} for the selected controller working conditions, that means the output signal frequency $f_w = 200$ kHz and the two values of supply voltage $V_{SUP} = 35$ V (Fig. 8a) and $V_{SUP} = 25$ V (Fig. 8b), are presented. As it is seen, the macromodel characteristics fit good to the measurement results and the biggest differences between the isothermal and nonisothermal results appear at the higher values of voltages V_{err} . The selfheating changes considerably the time duration of the signals, and the differences between isothermal and nonisothermal characteristics exceed even 60%. These differences decrease at the smaller values of the supply voltage.

In Fig. 9 the measured and calculated dependencies of the power dissipated on the error voltage – both in the isothermal and nonisothermal case are presented. Whereas, in Fig. 10 the calculated rise of the controller junction temperature over the ambient one, for $f_w = 200$ kHz, is shown.

As it results from Fig. 9, the power dissipated inside the controller working under the catalogue conditions overcomes its catalogue maximum value ($P_{\max} = 1$ W), even in the isothermal case, whereas the selfheating increases this value from about 10% ($V_{\text{SUP}} = 8$ V, $V_{\text{err}} = 3$ V) to about 50% ($V_{\text{SUP}} = 35$ V, $V_{\text{err}} = 3$ V). In the latter case the rise of the junction temperature ΔT_j is about 150 K, what can be recognise as the typical value of the maximum temperature for the silicon bipolar devices.

6. CONCLUSIONS

As it was shown in the paper, the selfheating affects the essential performances of the investigated controller considerably and the electrothermal macromodel proposed in the paper describes its nonisothermal characteristics and parameters correctly.

As it results from the performed investigations the influence of selfheating on SG3525A controller features depends on its working conditions. The influence of the junction temperature on the characteristics of the considered device is especially visible at the high values of the supply voltage, the error voltage and the output signal frequency.

The proposed form of macromodel which has risen on the base of the SG3525A controller investigations can be utilized for the other types of PWM controllers, but the detail description of the time pulse duration t_w and the current consumed by controller I_{SUP} can be different, as well as the macromodel parameters values should be found in the each case.

7. ACKNOWLEDGEMENTS

The authors would like to thank prof. W.J. Stepowicz for valuable discussions leads to improving of the final version of this paper.

8. REFERENCES

1. J. Zarębski: *A New Form of the BJT Model Including Electrothermal Interactions*. 11-th European Conf. on Circuit Theory and Design – ECCTD'93, Davos 1993, Vol. 1, p. 431.
2. J. Zarębski, K. Górecki, P. Jasicki: *Modelling of the PWM Controller in SPICE*. Int. Conference on Fundamentals of Electrotechnics and Circuit Theory IC-SPETO'2001, Ustroń, 2001, Vol. 2, s. 461 (in Polish).

3. J. Zarębski, K. Górecki, P. Jasicki: *The Electrothermal Macromodel of PWM Controller for SPICE*. 8-th International Conference Mixed Design of Integrated Circuits and Systems MIXDES 2001, Zakopane, 2001, p. 279.
4. K. Górecki, J. Zarębski, P. Jasicki: *The Circuit and the Method for Measurement of the Thermal Resistance of the Switched Power Supplies Controllers*. Patent application No. P344803.
5. J. Zarębski: *Modelowanie, symulacja i pomiary przebiegów elektrotermicznych w elementach półprzewodnikowych i układach elektronicznych*. Prace Naukowe WSM w Gdyni, Gdynia, 1996.
6. MicroSim Pspice A/D. Reference Manual. MicroSim Corp., 1997.
7. N. Mohan, T.M. Undeland, W.P. Robbins: *Power Electronics: Converters, Applications, and Design*. New York, John Wiley & Sons, 1989.
8. M.H. Rashid: *Power Electronics. Circuits, Devices, and Applications*. Englewood Cliffs, New Jersey, Prentice Hall, 1988.
9. Regulating Pulse Width Modulators SG3525A, Catalogue data, STMicroelectronics, 1998.
10. W.J. Stepowicz: *D.C. Nonisothermal Output Characteristics of a Bipolar Transistor Including Breakdown*. Bulletin of the Polish Academy of Sciences, Technical Sciences, Vol. XXX, No. 5-6, 1982, p. 85.

J. ZARĘBSKI, K. GÓRECKI

ELEKTROTERMICZNY MAKROMODEL STEROWNIKA PWM DEDYKOWANY DLA PROGRAMU SPICE

Streszczenie

W pracy przedstawiono problem modelowania sterowników PWM na potrzeby analizy z wykorzystaniem programu SPICE z uwzględnieniem samonagrzewania. Zaproponowano nowy elektrotermiczny makromodel sterownika SG3525A firmy STMicroelectronics, uwzględniający zależność szerokości impulsów wyjściowych oraz prądu zasilającego badany układ od napięcia błędu, napięcia zasilania i temperatury wnętrza układu. Wartości parametrów elektrycznych zostały wyznaczone na podstawie pomiarów odpowiednich charakterystyk izotermicznych, natomiast parametry termiczne wyznaczane są z pomiarów wykorzystujących krzywą chłodzenia układu oraz charakterystykę termometryczną dostępną dla użytkownika złącza p-n. W pracy zamieszczono szereg wyników pomiarów i obliczeń rozważanego sterownika, potwierdzających poprawność i skuteczność zaproponowanego makromodelu.

Słowa kluczowe: sterowniki PWM, samonagrzewanie, model elektrotermiczny

M

M

J.

M

K

K

J.

J.

Sp

In

Re

Li

M

M

J.

M

K.

K.

J.

J.

Co

Inf

Ye

Th

SPIS TREŚCI

M. Gajer: Wybrane zagadnienia analizy i projektowania rozproszonych systemów komputerowych czasu rzeczywistego	401
M. Gajer: Zagadnienia przydziału i planowania zadań w systemach czasu rzeczywistego	433
J. Stanclik: Zwiększenie dokładności makromodelu scalonych wzmacniaczy mocy małej częstotliwości	463
M. Czarnecki: Adaptacyjna kompensacja wpływu tłumienia wprowadzonego przez opady deszczu dla systemów satelitarnych	477
K. Wiatr, E. Jamro: Implementacja szybkich układów mnożących w strukturach FPGA	495
K. Noga: Jakość transmisji binarnej w kanale Hoyta	515
J. Zarębski, K. Górecki: Nowy algorytm wyznaczania nieizotermicznych charakterystyk dynamicznych elementów półprzewodnikowych w programie SPICE	525
J. Zarębski, K. Górecki: Elektroniczny makromodel sterownika PWM dedykowany dla programu SPICE	539
Spis treści	553
Informacje dla Autorów	554
Roczny spis treści	559
Lista recenzentów	565

CONTENTS

M. Gajer: The chosen problems of analysis and design of distributed real-time systems	401
M. Gajer: Problems of task scheduling and allocation in real-time systems	433
J. Stanclik: Accuracy increasing of macromodel for integrated circuit audio power amplifiers	463
M. Czarnecki: Adaptive compensation of the influence of rain fades for satellite communications systems	477
K. Wiatr, E. Jamro: High speed multiplication circuits implemented in FPGAs chips	495
K. Noga: Performance of binary transmission in Hoyt channel	515
J. Zarębski, K. Górecki: A new algorithm of semiconductor devices nonisothermal dynamic characteristics calculations in SPICE	525
J. Zarębski, K. Górecki: PWM controller electrothermal macromodel dedicated for SPICE	539
Contents of Electronics and Telecommunications Quarterly	553
Information for the Authors	554
Year contents of Electronics and Telecommunications Quarterly	559
The List of the reviewers	565

INFORMACJE DLA AUTORÓW

Redakcja przyjmuje do publikowania prace oryginalne, przeglądowe i monograficzne wchodzące w zakres szeroko pojętej elektroniki. Ponieważ KWARTALNIK ELEKTRONIKI I TELEKOMUNIKACJI jest czasopismem Komitetu Elektroniki i Telekomunikacji Polskiej Akademii Nauk, w związku z tym na jego łamach znajdują się prace naukowe dotyczące podstaw teoretycznych i zastosowań z zakresu elektroniki, telekomunikacji, mikroelektroniki, optoelektroniki, radiotechniki i elektroniki medycznej.

Artykuły powinny charakteryzować oryginalne ujęcie zagadnienia, własna klasyfikacja, krytyczna ocena (teorii lub metod), omówienie aktualnego stanu, lub postępu danej gałęzi techniki oraz omówienie perspektyw rozwojowych.

Artykuły publikowane w innych czasopismach nie mogą być kierowane do druku w Kwartalniku Elektroniki i Telekomunikacji w drugiej kolejności zgłoszenia.

Objętość artykułu nie powinna przekraczać 30 stron po około 1800 znaków na stronie, w tym rysunki i tabele.

Wymagania podstawowe.

Artykuły należy nadsyłać na wyraźnym, jednostronnym, czarno-białym wydruku komputerowym. Wydruk w formacie A4 powinien mieć znormalizowaną liczbę wierszy i znaków w wierszu (30 wierszy po 60 znaków w wierszu), w dwóch egzemplarzach, w języku polskim lub angielskim wybranym przez autora. Do wydruku powinna być dołączona dyskietka z elektronicznym tekstem artykułu. Preferowane edytory to WORD 6 lub 8. Układ artykułu (w wersji podstawowej) musi być następujący:

- Tytuł.
- Autor (imię i nazwisko autora/ów).
- Miejsce pracy (nazwa instytucji, miejscowość, adres, + ew. adres elektroniczny (e-mail)).
- Zwięzłe streszczenie powinno być w języku takim, w jakim jest pisany artykuł (wraz ze słowami kluczowymi).
- Tekst podstawowy powinien mieć następujący układ:
 1. WPROWADZENIE
 2. np. TEORIA
 3. np. WYNIKI NUMERYCZNE
 - 3.1.
 - 3.2.
 4.
 5.
 6. PODSUMOWANIE
 7. ew. PODZIĘKOWANIE
 8. BIBLIOGRAFIA
- Układ streszczenia w dodatkowej wersji językowej powinien być następujący:

TYTUŁ (w języku angielskim – o ile artykuł pisany jest w języku polskim i na odwrot),
AUTOR (inicjał imienia i nazwisko).

Obszerne do 3600 znaków streszczenie (wraz z słowami kluczowymi) w języku:

- a) angielskim, gdy artykuł pisany jest w języku polskim,
- b) polskim, gdy artykuł pisany jest w języku angielskim.

Streszczenie to powinno pozwolić czytelnikowi na uzyskanie istotnych informacji zawartych w pracy. Z tego względu w streszczeniu tym mogą być cytowane numery istotnych wzorów, rysunków i tabel zawartych w podstawowej wersji językowej.

- Wszystkie strony muszą mieć numerację ciągłą.

Sposób pisania tekstu.

Tekst powinien być pisany bez używania wyróżnień, a w szczególności nie dopuszcza się spacjiowania, podkreślania i pisania tekstu dużymi literami z wyjątkiem wyrazów, które umownie pisze się dużymi literami (np. FORTRAN). Proponowane wyróżnienia Autor może zaznaczyć w maszynopisie zwykłym ołówkiem za pomocą przyjętych znaków adiustacyjnych np. podkreślenie linią przerywaną oznacza spacjiowanie (rozstrzelanie), podkreślenie linią ciągłą – pogrubienie, podkreślenie wężykiem – kursywą.

Tekst powinien być napisany z podwójnym odstępem między wierszami, tytuły i podtytuły małymi literami. Marginesy z każdej strony powinny mieć około 35 mm. Wielkość czcionki wydruku powinna być zbliżona co najmniej do wielkości czcionki maszyny do pisania (minimum 12 punktów). Przy podziale pracy na rozdziały i podrozdziały, ich cyfrowe oznaczenia nie powinny być większe niż III stopnia (np. 4.1.1.).

Sposób pisania tabel.

Tabele powinny być pisane na oddzielnych stronach. Tytuły rubryk pionowych i poziomych powinny być napisane małymi literami z podwójnym odstępem między wierszami. Przypisy (notki) dotyczące tabel należy pisać bezpośrednio pod tabelami. Tabele należy numerować kolejno liczbami arabskimi, u góry każdej tabeli podać tytuł dwujęzyczny. W pierwszej kolejności w podstawowej wersji językowej, a później w dodatkowej wersji językowej. Tabele umieścić na końcu maszynopisu. Przyjmowane są tabele algorytmów i programy na wydrukach komputerowych. W tym przypadku zachowany jest ich oryginalny układ. Tabele powinny być cytowane w tekście.

Sposób pisania wzorów matematycznych.

Roźmieszczenie znaków, cyfr, liter i odstępów powinno być zbliżone do roźmieszczenia elementów druku. Wskaźniki i wykładniki potęg powinny być napisane wyraźnie i być prawidłowo obniżone lub podwyższone w stosunku do linii wiersza podstawowego. Znaki nad literami i cyframi, całkami i in. symbolami (strzałki, linie, kropki, daszki) powinny być umieszczone dokładnie nad tymi elementami, do których się odnoszą. Numery wzorów cyframi arabskimi powinny być kolejne i umieszczone w nawiasach okrągłych z prawej strony. Nazwy jednostek, symbole literowe i graficzne powinny być zgodne w wytycznymi IEE (International Electronical Commision) oraz ISO (International Organization of Standarization).

Powołania.

Powołania na publikację powinny być umieszczone na ostatnich stronach tekstu pod tytułem „Bibliografia”, opatrzone numeracją kolejną bez nawiasów. Numeracja ta powinna być zgodna z odnośnikami w tekście artykułu. Przykłady opisu publikacji:

- periodycznej 1. F. Valdoni: A new milimetre wave satelitę. E.T.T. 1990, vol. 2, no 5, pp. 141–148
- nieperiodyczne 2. K. Andersen: A resource allocation framework. XVI International Symposium, Stockholm (Sweden), may 1991, paper A 2.4
- książki 3. Y.P. Tvidis: Operation and modeling of the MOS transistors. New York, McGraw-Hill, 1987, p. 553

Materiały ilustracyjne.

Rysunki powinny być wykonane wyraźnie, na papierze gładkim lub milimetrowym w formacie nie mniejszym niż 9×12 cm. Mogą być także w postaci wydruku komputerowego (preferowany edytor Corel Draw). Fotografie lub diapozytywy przyjmowane są raczej czarno-białe w formacie nie przekraczającym 10×15 cm. Na marginesie każdego rysunku i na odwrocie fotografii powinno być napisane ołowiem imię i nazwisko Autora oraz skrót tytułu artykułu, do którego są przeznaczone oraz numer rysunku. Spis podpisów pod rysunki i fotografie powinien być umieszczony na oddzielnej stronie. Podpisy pod rysunkami (fotografiami) powinny być dwujęzyczne: w pierwszej kolejności w podstawowej wersji językowej, a później w dodatkowej wersji językowej. Rysunki powinny być cytowane w tekście.

Uwagi dodatkowe.

Na odrębnej stronie powinny być podane następujące informacje:

- adres do korespondencji z kodem pocztowym (domowy lub do miejsca pracy),
- telefon domowy i/lub do miejsca pracy,
- adres e-mailowy (jeśli autor posiada).

Autorowi przysługuje bezpłatnie 20 odbitek artykułu. Dodatkowe egzemplarze odbitek, lub cały zeszyt Autor może zamówić u wydawcy na własny koszt.

Autora obowiązuje korekta autorska, którą powinien wykonać w ciągu 3 dni od daty otrzymania tekstu z Redakcji oraz zwrócić osobiście, lub listownie pod adres Redakcji. Korekta powinna być naniesiona na przekazanych Autorowi szpaltach na marginesach ew. na osobnym arkuszu w przypadku uzupełnień tekstu większych niż dwa wiersze. W przypadku nie zwrócenia korekty w terminie, korektę przeprowadza Redakcja Techniczna Wydawcy. Redakcja prosi Autorów o powiadomienie ją o zmianie miejsca pracy i adresu prywatnego.

INFORMATION FOR AUTHORS OF K.E.T

The editorial staff will accept for publishing only original monographic and survey papers concerning widely understood electronics. Because of the fact that KWATRALNIK ELEKTRONIKI I TELEKOMUNIKACJI is a journal of the Committee for Electronics and Telecommunications of Polish Academy of Science, it presents scientific works concerning theoretical bases and applications from the field of electronics, telecommunications, microelectronics, optoelectronics, radioelectronics and medical electronics.

Articles should be characterised by original depiction of a problem, its own classification, critical opinion (concerning theories or methods), discussion of an actual state or a progress of a given branch of a technique and discussion of development perspectives.

An article published in other magazines can not be submitted for publishing in K.E.T.

The size of an article can not exceed 30 pages, 1800 character each, including figures and tables.

Basic requirements

The article should be submitted to the editorial staff as a one side, clear, black and white computer printout in two copies. The article should be prepared in English or Polish. Floppy disc with an electronic version of the article should be enclosed. Preferred wordprocessors: WORD 6 or 8.

Layout of the article.

– Title.

– Author (first name and surname of author/authors).

– Workplace (institution, address and e-mail).

– Concise summary in a language article is prepared in (with keywords).

– Main text with following layout:

- Introduction
- Theory (if applicable)
- Numerical results (if applicable)
- Paragraph 1
- Paragraph 2
-
-
- Conclusions
- Acknowledgements (if applicable),
- References

– Summary in additional language:

- Title (in Polish, if article was prepared in English and vice versa)
- Author (first name initials and surname)
- One page summary (in Polish, if article was prepared in English and vice versa).

Pages should have continuous numbering.

Main text

Main text can not contain formatting such as spacing, underlining, words written in capital letters (except words that are commonly written in capital letters). Author can mark suggested formatting with pencil on the margin of the article using commonly accepted adjusting marks.

Text should be written with double line spacing with 35 mm left and right margin. Titles and subtitles should be written with small letters. Titles and subtitles should be numbered using no more than 3 levels (i.e. 4.1.1.).

Tables

Tables with their titles should be placed on separate page at the end of the article. Titles of rows and columns should be written in small letters with double line spacing. Annotations concerning tables should be placed directly below the table. Tables should be numbered with Arabic numbers on the top of each table. Table can consist algorithm and program listings. In such cases original layout of the table will be preserved. Table should be cited in the text.

Mathematical formulas

Characters, numbers, letters and spacing of the formula should be adequate to layout of main text. Indexes should be properly lowered or raised above the basic line and clearly written. Special characters such as lines, arrows, dots should be placed exactly over symbols which they are attributed to. Formulas should be numbered with Arabic numbers placed in brackets on the right side of the page. Units of measure, letter and graphic symbols should be printed according to requirements of IEE (International Electrotechnical Commission) and ISO (International Organisation of Standardisation).

References

References should be placed at the end of the main text with the subtitle „References”. References should be numbered (without brackets) adequately to references placed in the text. Examples of periodical [1], non-periodical [2] and book [3] references:

1. F. Valdoni: A new millimetre wave satellite. E.T.T. 1990, vol.2, no 5, pp. 141–148
2. K. Anderson: A resource allocation framework. XVI International Symposium, Stockholm (Sweden), May 1991. paper A 2.4
3. Y.P. Tvidis: Operation and modelling of the MOS transistors. New York, McGraw-Hill, 1987, p. 553

Figures

Figures should be clearly drawn on plain or millimetre paper in the format not smaller than 9×12 cm. Figures can be also printed (preferred editor – CorelDRAW). Photos or diapositives will be accepted in black and white format not greater than 10×15 cm. On the margin of each drawing and on the back side of each photo author name and abbreviation of the title of article should be placed. Figure's captions should be placed on separate page. Figures should be cited in the text.

Additional information

On the separate page following information should be placed:

- mailing address (home or office),
- phone (home or/and office),
- e-mail.

Author is entitled to free of charge 20 copies of article. Additional copies or the whole magazine can be ordered at publisher at the one's expense.

Author is obliged to perform the author's correction, which should be accomplished within 3 days starting from the date of receiving of the text from the editorial staff. Corrected text should be returned to the editorial staff personally or by mail. Correction marks should be placed on the margin of copies received from the editorial staff or if needed on separate pages. In the case when the correction is not returned in said time limit, correction will be performed by technical editorial staff of the publisher.

In case of changing of workplace or home address Authors are asked to inform the editorial staff.

POLSKA AKADEMIA NAUK
KOMITET ELEKTRONIKI I TELEKOMUNIKACJI

KWARTALNIK
ELEKTRONIKI I TELEKOMUNIKACJI

ELECTRONICS AND
TELECOMMUNICATIONS
QUARTERLY

SPIS TREŚCI – TOM 47/2001

WYDAWNICTWO NAUKOWE PWN
WARSZAWA 2001

SPIS TREŚCI DO TOMU 47 – 2001

Nr 1

A. A. Petrovsky, J. Baszun: Projektowanie cyfrowych przestrajanych filtrów pasmowych – synteza i zastosowania	5
A. A. Petrovsky, J. Baszun: Efekt skończonej długości słowa w rekursywnych przestrajanych filtrach pasmowych	27
A. A. Kucharski: Cylindryczne rezonatory dielektryczne w zastosowaniach antenowych – analiza z wykorzystaniem równań całkowych	43
D. Kania: Realizacja układów kombinacyjnych w strukturach MACH	65
W. Podmiotko, A. Szymański: prądowo-napięciowy, niskoszumny scalony wzmacniacz impulsów	75
J. Kowalski, T. Kacprzak: Analizowanie i projektowanie sieci neuronowych komórkowych realizowanych w technice układowej wzmacniaczy transkonduktacyjnych jedno- i wielowejściowych	87
S. Kozieł, S. Szczepański: Projektowanie liniowych operacyjnych wzmacniaczy transkonduktacyjnych (OTA) CMOS z wykorzystaniem techniki sumowania/odejmowania prądów	121

Nr 2

T. Ostrowski: Ciała skończone i kody Fire'a	145
R. Golański: Adaptacja częstotliwości próbkowania jako metoda kompandowania w modulacjach różnicowych	161
J. Falkiewicz: Zastosowanie algorytmu kratowego typu LS w adaptacyjnym podpasmowym systemie usuwania echa akustycznego	183
A. Jakubiak, A. Zieliński: Prawdopodobieństwo fałszywego algomu detektora log-t w obecności skolerowanych zakłóceń typu K	209
A. Jakubiak, M. Muczko: Klasyfikator Kullbacka-Leihlera w obecności skolerowanych zakłóceń biernych	221
K. Wiatr, E. Jamro: Układy mnożne przez stały współczynnik implementowane w układach programowalnych FPGA	233
B. Stec, Z. Chudy, L. Kachel: Szerokopasmowy układ formowania wiązki antenowej dookólnego systemu namiaru źródeł promieniowania	255
Informacje dla Autorów	274

Nr 3

J. Halawa: Dynamika zamkniętego układu sterowania z regulatorem Reswicka	285
K. Noga: Prawdopodobieństwo błędnego bloku dla transmisji w kanale z wolnymi zanikami Nakagami i wielostanowej modulacji	293

J. Kołodziej, A. Kos: Estymacja strat energii w układach cyfrowych CMOS	303
B. Dudojć, J. Mindykowski: Opis i metoda oceny właściwości metrologicznych i eksploatacyjnych dwuprzewodowych torów pomiarowych temperatury w obszarach zagrożonych wybuchem na statkach morskich	323
B. Dudojć, J. Mindykowski: Błędy doatkowe i ich wyznaczenie w pomiarach temperatury w obszarach zagrożonych wybuchem na statkach morskich	355
K. Górecki, J. Zarębski: Nowy algorytm wyznaczania mocy czynnej tranzystora MOS w programie SPICE	377

Nr 4

M. Gajer: Wybrane zagadnienia analizy i projektowania rozproszonych systemów komputerowych czasu rzeczywistego	401
M. Gajer: Zagadnienia przydziału i planowania zadań w systemach czasu rzeczywistego	433
J. Stanlik: Zwiększenie dokładności makromodelu scalonych wzmacniaczy mocy małej częstotliwości	463
M. Czarnecki: Adaptacyjna kompensacja wpływu tłumienia wprowadzonego przez opady deszczu dla systemów satelitarnych	477
K. Wiatr, E. Jamro: Implementacja szybkich układów mnożących w strukturach FPGA	495
K. Noga: Jakość transmisji binarnej w kanale Hoyta	515
J. Zarębski, K. Górecki: Nowy algorytm wyznaczania nieizotermicznych charakterystyk dynamicznych elementów półprzewodnikowych w programie SPICE	525
J. Zarębski, K. Górecki: Elektrotermiczny makromodel sterownika PWM dedykowany dla programu SPICE	539
Spis treści	553
Informacje dla Autorów	554
Roczny spis treści	561
Lista recenzentów	565

CONTENTS

Nr 1

A. A. Petrovsky, J. Baszun: Design of digital tunable bandpass filters – synthesis and applications	5
A. A. Petrovsky, J. Baszun: Finite word length effects in tunable recursive bandpass filters	27
A. A. Kucharski: Cylindrical dielectric resonators in antenna applications – integral equations analysis	43
D. Kania: Technology mapping algorithm for MACH devices	65
W. Podmiotko, A. Szymański: Low noise, current-voltage integrated charge amplifier	75
J. Kowalski, T. Kacprzak: Analysis and designing of cellular neural networks based on transconductance amplifier – OTA and multiple input transconductance amplifier – MIOTA	87
S. Kozieł, S. Szczepański: Design of linear CMOS OTA using a current addition/subtraction technique	121

Nr 2

T. Ostrowski: Finite fields and fire codes	145
R. Golański: Nonuniform sampling as a companding method in differential modulations	161
J. Falkiewicz: Application of attice LS-type algorithm subband adaptive echo cancellation system	183

A. Jakubiak, A. Zieliński: False alarm probability of log-t detector in correlated K-distributed clutter	209
A. Jakubiak, M. Muczko: The Kullback-Leibler classifier in correlated clutter	221
K. Wiatr, E. Jamro: Constant coefficient multiplication in FPGA structures	233
B. Stec, Z. Chudy, L. Kachel: Wideband antennas beam forming circuit in circle direction finding system	255
Information for the Authors	274

Nr 3

J. Halawa: Dynamics of closed-loop system with Reswicks controller	285
K. Noga: Probability of block error from transmission in slow Nakagami fading and MNCFSK	293
J. Kołodziej, A. Kos: Estimation of energy losses in CMOS digital circuits	303
B. Dudojć, J. Mindykowski: Description and evaluation technique of metrology and exploitation properties of two wires temperature measurement lines in ship's hazardous areas	323
B. Dudojć, J. Mindykowski: Additional errors and their detection in temperature measurements in ship's hazardous areas	355
K. Górecki, J. Zarębski: A new algorithm of MOSFET real power calculations in SPICE	377

Nr 4

M. Gajer: The chosen problems of analysis and design of distributed real-time systems	401
M. Gajer: Problems of task scheduling and allocation in real-time systems	433
J. Stanlik: Accuracy increasing of macromodel for integrated circuit audio power amplifiers	463
M. Czarnecki: Adaptive compensation of the influence of rain fades for satellite communications systems	477
K. Wiatr, E. Jamro: High speed multiplication circuits implemented in FPGAs chips	495
K. Noga: Performance of binary transmission in Hoyt channel	515
J. Zarębski, K. Górecki: A new algorithm of semiconductor nonisothermal dynamic characteristics calculations in SPICE	525
J. Zarębski, K. Górecki: PWM controller electrothermal macromodel dedicated for SPICE	539
Contents of Electronics and Telecommunications Quarterly	553
Information for the Authors	554
Year contents of Electronics and Telecommunications Quarterly	561
The List of the reviewers	565

LISTA RECENZENTÓW KWARTALNIKA ELEKTRONIKI I TELEKOMUNIKACJI

Dr inż. *Jarostaw Arabas*
Politechnika Warszawska,
Instytut Systemów Elektronicznych

Prof. dr hab. inż. *Jerzy Baranowski*
Politechnika Warszawska,
Instytut Systemów Elektronicznych

Mgr. inż. *Piotr Bora*
Wojskowa Akademia Techniczna
Warszawa

Dr hab. inż. *Grzegorz Ciesielski*
Politechnika Łódzka
Instytut Elektrotechniki Teoretycznej,
Metrologii i Materiałoznawstwa

Dr hab. inż. *Witold Czarnecki*
Politechnika Warszawska
Instytut Telekomunikacji

Prof. nzw. dr hab. *Andrzej Dąbrowski*
Politechnika Warszawska
Instytut Telekomunikacji

Prof. dr hab. inż. *Janusz A. Dobrowolski*
Politechnika Warszawska,
Instytut Systemów Elektronicznych

Dr inż. *Jarostaw Domaszewicz*
Politechnika Warszawska,
Instytut Telekomunikacji

Dr inż. *Przemysław Dymarski*
Politechnika Warszawska,
Instytut Telekomunikacji

Prof. dr hab. inż. *Jerzy Frączek*
Politechnika Śląska
Instytut Automatyki

Dr inż. *Zbigniew Gajo*
Politechnika Warszawska,
Instytut Systemów Elektronicznych

Prof. dr hab. inż. *Bogdan Galwas*
Politechnika Warszawska,
Instytut Mikroelektroniki i Optoelektroniki

Mgr inż. *Marcin Golański*
Politechnika Warszawska
Instytut Telekomunikacji

Dr inż. *Konrad Hejn*
Politechnika Warszawska
Instytut Systemów Elektronicznych

Dr hab. inż. *Ewa Hermanowicz*
Politechnika Gdańska
Wydział Elektroniki i Telekomunikacji
i Informatyki

Prof. dr hab. inż. *Ryszard Jachowicz*
Politechnika Warszawska,
Instytut Systemów Elektronicznych

Dr inż. *Andrzej Jakubiak*
Politechnika Warszawska,
Instytut Telekomunikacji

Prof. dr hab. *Andrzej Jakubowski*
Politechnika Warszawska
Instytut Mikroelektroniki
i Optoelektroniki

Dr inż. *Stanisław Jankowski*
Politechnika Warszawska
Instytut Systemów Elektronicznych

Dr. hab. inż. *Swietlana Januszkiewicz*
Politechnika Szczecińska,
Wydział Informatyki

Dr. inż. *Jacek Jarkowski*
Politechnika Warszawska
Instytut Systemów Elektronicznych

Dr inż. *Krzysztof Jasiński*
Politechnika Warszawska,
Instytut Telekomunikacji

Prof. dr hab. inż. *Janusz Jaworski*
Politechnika Lubelska,
Katedra Automatyki

Dr hab. inż. *Wojciech Kabaciński*
Politechnika Poznańska,
Instytut Elektroniki i Telekomunikacji

Prof. dr hab. inż. *Zbigniew Kaczkowski*
Polska Akademia Nauk,
Instytut Fizyki, Warszawa

Dr inż. *Ryszard Kisiel*
Politechnika Warszawska,
Instytut Mikroelektroniki i Optoelektroniki

Dr inż. *Tomasz Kosiło*
Politechnika Warszawska,
Instytut Radioelektroniki

Dr inż. *Zbigniew Kowalczyk*
Przemysłowy Instytut Telekomunikacji
Gdańsk–Wrzeszcz

Mgr inż. *Henryk Kowalski*
Politechnika Warszawska,
Instytut Informatyki

Prof. dr hab. inż. *Andrzej Kraśniewski*
Politechnika Warszawska,
Instytut Telekomunikacji

Prof. dr hab. inż. *Małgorzata Kujawińska*
Politechnika Warszawska,
Instytut Mikromechaniki i Fotoniki

Dr inż. *Stawomir Kukliński*
Politechnika Warszawska
Instytut Telekomunikacji

Dr hab. inż. *Wiesław Kuźmicz*
Politechnika Warszawska
Instytut Mikroelektroniki
i Optoelektroniki

Prof. dr hab. inż. *Juliusz Lech Kulikowski*
Polska Akademia Nauk, Warszawa
Instytut Biocybernetyki i Inżynierii Biomedycznej

Prof. dr hab. inż. *Zygmunt Kuśmierek*
Politechnika Łódzka, Instytut Elektrotechniki
Teoretycznej, Meteorologii i Materiałoznawstwa

Dr inż. *Lech Lewandowski*
Politechnika Warszawska
Instytut Systemów Elektronicznych

Prof. dr hab. inż. *Tadeusz Łuba*
Politechnika Warszawska,
Instytut Telekomunikacji

Mgr inż. *Grzegorz Mazur*
Politechnika Warszawska,
Instytut Informatyki

Prof. dr hab. inż. *Marian Miłek*
Politechnika Zielonogórska,
Instytut Metrologii Elektrycznej

Prof. dr hab. inż. *Tadeusz Morawski*
Politechnika Warszawska,
Instytutu Radioelektroniki

Prof. dr hab. inż. *Andrzej Napieralski*
Politechnika Łódzka,
Instytutu Elektroniki

Prof. dr *Wincenty Pajewski*
Polska Akademia Nauk
Instytut Podstawowych Problemów Techniki

Dr hab. inż. *Zdzisław Papir*
Akademia Górniczo-Hutnicza, Kraków
Katedra Telekomunikacji

Dr inż. *Andrzej Paszkiewicz*
Politechnika Warszawska,
Instytut Telekomunikacji

Dr hab. inż. *Jan Petykiewicz*
Politechnika Warszawska,
Instytut Fizyki

Dr. in. *Andrzej Pfizner*
Politechnika Warszawska,
Instytut Mikroelektroniki i Optoelektroniki

Prof. dr hab. inż. *Anna Piotrowska*
Instytut Technologii Elektronicznej
Warszawa

Prof. dr hab. *Włodzimierz Pogribny*
Akademia Techniczno-Rolnicza, Bydgoszcz
Instytut Telekomunikacji

Mgr inż. *Mariusz Rawski*
Politechnika Warszawska
Instytut Telekomunikacji

Dr inż. *Miroslaw Rojewski*
Politechnika Gdańska,
Instytut Systemów Informacyjnych

Prof. nadzw. dr hab. *Stanisław Rostonec*
Politechnika Warszawska
Instytut Radioelektroniki

Prof. dr hab. inż. *Jerzy Siuzdak*
Politechnika Warszawska,
Instytut Telekomunikacji

Prof. nadzw. dr hab. *Władysław Skarbek*
Politechnika Warszawska
Instytut Radioelektroniki

Dr inż. *Andrzej Skorupski*
Politechnika Warszawska
Instytut Informatyki

Mgr inż. *Krzysztof Skowroński*
Politechnika Warszawska,
Instytut Systemów Elektronicznych

Prof. dr hab. inż. *Władysław Soluch*
Instytut Technologii Materiałów
Elektronicznych

Prof. dr hab. inż. *Marek Stabrowski*
Politechnika Warszawska,
Instytutu Elektrotechniki Teoretycznej
i Miernictwa Elektrycznego

Dr hab. inż. *Witold J. Stepowicz*
Politechnika Gdańska
Wydział Elektroniki, Telekomunikacji
i Informatyki

Dr inż. *Jerzy Szabatin*
Politechnika Warszawska,
Instytut Systemów Elektronicznych

Dr inż. *Jacek Szymanowski*
Politechnika Warszawska,
Instytut Automatyki i Informatyki Stosowanej

Dr inż. *Lech Śliwa*
Politechnika Warszawska,
Instytut Podstaw Elektroniki

Prof. dr hab. inż. *Leszek Trybus*
Politechnika Rzeszowska,
Katedra Automatyki i Informatyki

Prof. dr hab. inż. *Aleksander Urbaś*
Politechnika Warszawska,
Instytut Systemów Elektronicznych

Mgr inż. *Krzysztof Włostowski*
Politechnika Warszawska
Instytut Telekomunikacji

Plk. dr hab. inż. *Marian Wnuk*
Wojskowa Akademia Techniczna
Warszawa

Prof. dr hab. inż. *Jacek Wojciechowski*
Politechnika Warszawska,
Instytut Systemów Elektronicznych

Prof. nzw.,
dr hab. inż. *Andrzej Wojtkiewicz*
Politechnika Warszawska,
Instytut Systemów Elektronicznych