Michał GETKA[1], Michał KARPOWICZ[1]

Instytut Automatyki i Informatyki Stosowanej, Politechnika Warszawska (1)

doi:10.15199/48.2018.04.32

Aspekty energooszczędnego sterowania wydajnością pracy procesora systemu komputerowego zgodnego z ACPI w systemie Linux*

Streszczenie. *Zwiększanie efektywności energetycznej systemów komputerowych jest działaniem pożądanym zarówno ze względów ekologicznych jak i praktycznych. Procesor CPU oraz jego system chłodzący mają istotny udział w całościowym poborze mocy urządzenia. Standard ACPI definiuje mechanizmy pozwalające na zmianę poziomów wydajności pracy procesora. W artykule omówiono elementy architektury jądra systemu Linux oraz wybrane aspekty specyfikacji standardu ACPI, które są istotne z punktu widzenia implementacji energooszczędnego regulatora poziomów wydajności pracy procesora.*

Abstract. *Efforts enhancing efficiency of computing systems are desired for both ecological and practical reasons. CPU activity and its influence on cooling system has significant impact on the overall device energy consumption. Mechanisms required to alter the CPU performance states have been defined by the ACPI standard. This study discusses important aspects of both the Linux kernel architecture and the ACPI specification that are relevant for the implementation of energy efficient CPU power control systems.* ***(Aspects of energy efficient performance control of an ACPI-compliant computer system processor in Linux)***

Słowa kluczowe: ACPI, architektura SMP, technika DVFS, energooszczędne systemy komputerowe, Linux
Keywords: ACPI, SMP, DVFS, energy efficient computing, Linux

Wprowadzenie

Doskonale znane prawo Moore'a stanowi, że liczba tranzystorów w układzie scalonym mikroprocesora podwaja się co około 24 miesiące. Wraz ze wzrostem mocy obliczeniowej, związanej z liczbą tranzystorów układu scalonego, wzrasta pobór energii elektrycznej. W przypadku dużych centrów danych z ekonomicznego punktu widzenia wskazane jest wobec tego, aby rozbudowie infrastruktury zwiększającej zakres świadczonych usług towarzyszyły inwestycje w rozwiązania pozwalające na zwiększenie wydajności energetycznej systemu.

Energooszczędność układów elektronicznych ma również wielkie znaczenie w segmencie konsumenckim. Zasilane przez baterie urządzenia przenośne, np. laptopy, telefony lub tablety, są nieodłączną częścią codziennego życia. Zmniejszanie ilości energii elektrycznej zużywanej przez tego typu urządzenia pozwala wydłużać czas ich pracy pomiędzy cyklami ładowania. Towarzyszące temu ograniczenie poziomu wydzielania ciepła pozwala natomiast na zmniejszenie gabarytów układów chłodzących i miniaturyzację urządzeń.

Producenci komponentów systemów komputerowych odpowiadają na zapotrzebowanie rynku proponując urządzenia zawierające mechanizmy zarządzania zużyciem energii umożliwiające układom mikroprocesorowym najnowszych generacji pracę na różnych poziomach wydajności. W celu zwiększenia wydajności energetycznej układów mikroprocesorowych wydajność pracy procesorów jest uzależniania od chwilowego zapotrzebowania na moc obliczeniową niezbędną dla zapewnienia oczekiwanej funkcjonalności systemu. Zazwyczaj problem energooszczędnego sterowania pracą procesora polega na maksymalizacji wskaźnika postaci:

$$\text{wydajność energetyczna} = \frac{\text{liczba zadań obliczeniowych}}{\text{wykorzystana energia}}, \tag{1}$$

przy ograniczeniach podyktowanych wymaganiami dotyczącymi jakości usług dostarczanych przez system komputerowy [1, 2, 3, 4].

Artykuł ten prezentuje przegląd kluczowych zagadnień związanych z mechanizmami kontroli poziomów wydajności procesora w systemach komputerowych zgodnych ze specyfikacją ACPI (Advanced Configuration and Power Interface) [5]. Omówiony został również moduł `cpufeq` jądra systemu operacyjnego Linux wykorzystujący wspomniane mechanizmy w celu energooszczędnego zarządzania poziomami wydajności jednostki obliczeniowej. Szczególny nacisk położony został na omówienie cech specyfikacji oraz funkcjonalności modułów związanych z obsługą systemów wielordzeniowych.

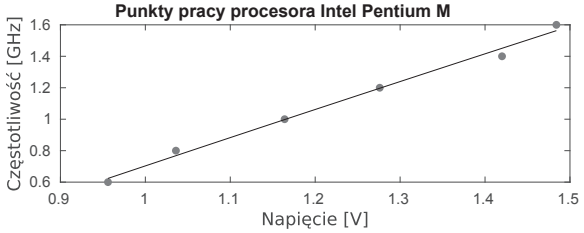
Standard ACPI a sterowanie wydajnością procesora

ACPI (*Advanced Configuration and Power Interface*) jest otwartym standardem definiującym model obsługi poszczególnych komponentów systemu komputerowego. Jako obsługę rozumie się ich inicjalizację oraz pośrednictwo w komunikacji systemu operacyjnego z wbudowanymi mechanizmami zarządzania energią. W ramach standardu ACPI definiuje się nie tylko interfejsy obsługi wspomnianych mechanizmów, ale również określa się pewne aspekty specyfikacji ich działania [5].

Mechanizmy definiowane przez ACPI muszą być wspierane zarówno przez system operacyjny jak i *firmware* systemu komputerowego (BIOS). Przy uruchamianiu systemu, *firmware* warstwy sprzętowej ładuje do pamięci operacyjnej struktury danych, zwane tablicami. Tablice te zawierają informacje o urządzeniach dostępnych w systemie oraz o metodach ich obsługi. Sterowanie jest następnie przekazywane do systemu operacyjnego. Podsystem realizujący mechanizmy ACPI przetwarza zawartość tablic do postaci przestrzeni nazw ACPI (*ACPI namespace*) składającej się ze zbioru obiektów powiązanych z poszczególnymi urządzeniami i funkcjonalnościami. Obiekty definiowane są w ramach bloków definicji (*definition blocks*) przez producentów komponentów sprzętowych z wykorzystaniem języka ASL (*ACPI Source Language*), który jest kompilowany do postaci kodu bajtowego AML (*ACPI Machine Language*), wykonywanego przez interpreter wbudowany w system operacyjny [6].

Specyfikacja ACPI definiuje hierarchię stanów, w których znajdować się może system komputerowy i jego komponenty. Wyróżniono cztery stany globalne definiowane

*Badania finansowane ze środków Narodowego Centrum Nauki w ramach projektu nr 2015/17/B/ST6/01885.



Rys. 1. Charakterystyka minimalnego napięcia pozwalającego na prawidłową pracę przy danej częstotliwości dla procesora Intel Pentium M (opracowanie własne na podstawie [8]).

dla całego urządzenia: uruchomione (G0), uśpione (G1) oraz wyłączone (G2 i G3). W systemie który jest uruchomiony (jest w stanie G0) procesor może być w jednym ze stanów C związanych z mechanizmem usypiania. Stan C0 jest stanem normalnej pracy, stany C1, C2 i dalsze odpowiadają coraz głębszemu uśpieniu. W czasie normalnej pracy procesor może działać z różną wydajnością – różnym jej poziomom odpowiadają stany P, gdzie stan P0 odpowiada najwyższej wydajności. Równolegle do stanów P występują również stany T związane z mechanizmem *throttlingu* używanego do ograniczania wydajności przez mechanizmy ochrony termicznej [5].

Poszczególne stany P odpowiadają różnym poziomom wydajności pracy procesora. We współczesnych procesorach modyfikacja wydajności realizowana jest poprzez modyfikację częstotliwości taktowania oraz napięcia zasilania rdzenia. Z energetycznego punktu widzenia rozwiązanie to jest korzystniejsze niż modyfikowanie wyłącznie częstotliwości, co w praktyce jest tożsame z działaniem mechanizmu *throttling*.

Pobór mocy układu logicznego opartego na technologii CMOS można opisać w następujący sposób:

$$P = CfV^2 + P_{static}, \tag{2}$$

gdzie C oznacza efektywną pojemności przełączania, f częstotliwość, V napięcie zasilające, natomiast P_{static} oznacza niezmienną wartość mocy pobieranej przez mechanizmy związane z upływnością [7]. Kwadratowa zależność poboru mocy i napięcia zasilania powoduje, że korzystne jest ograniczanie napięcia do możliwie niskich wartości. Bramki logiczne układów logicznych wykonanych w technologii CMOS cechują się niewielką pojemnością wejściową. Każde ich przełączenie związane jest z przeładowaniem tej pojemności do wartości napięcia odpowiadającej nowej logicznej wartości wejściowej. W przypadku przejścia od stanu wysokiego do niskiego następuje rozładowanie pojemności do masy układu. Przejście od stanu niskiego do wysokiego związane jest z przeładowaniem pojemności bramki przez napięcie zasilające układ V. Proces ten jest tym szybszy im wyższe jest napięcie V. Oznacza to, że dla danej częstotliwości pracy układu f zdefiniowane jest takie napięcie zasilające V, poniżej którego prawidłowa praca układu nie jest możliwa. W praktyce minimalne napięcie niezbędne do pracy układu z daną częstotliwością zależne jest od jego budowy i stanowi indywidualny parametr konkretnego urządzenia. Przykładową charakterystykę częstotliwości i minimalnego napięcia pracy procesora przedstawiono na Rys. 1.

Każdy stan P odpowiada jednemu punktowi pracy rdzenia procesora wiążącemu częstotliwość taktowania układu z wartością napięcia zasilania. Lista stanów P oraz ich parametry definiowane są przez producenta procesora. Obiekt `_PSS` struktury opisującej procesor w tablicy ACPI zawiera informacj˛e na temat wspieranych przez urz ˛adzenie stanów P. W szczególności brak obiektu `_PSS` ´swiadczy o braku wsparcia dla zarz ˛adzania wydajności ˛a procesora przez urz ˛adzenie w sposób definiowany przez standard ACPI.

W przypadku systemów wielordzeniowych z reguły wyst˛epuj ˛a zale˙zno´sci pomi˛edzy wydajnosci ˛a procesorów logicznych. Poszczególne rdzenie najcz´esciej korzystaj ˛a ze wspólnych linii zasilania. Oznacza to, ˙ze musz ˛a one pracować w tym samym stanie P [9]. W szczególności ma to miejsce w urz ˛adzeniach wspieraj ˛acych technologi˛e *hyper-threading*, kiedy to dwa procesory logiczne widziane przez system operacyjny s ˛a w rzeczywistosci j´ednym fizycznym rdzeniem [10]. Procesory logiczne, które musz ˛a pracowac ná tym samym poziomie wydajnosci P kojarzone s ˛a w ramach *domeny cz˛estotliwosciowej*.

Znajomość zale˙zno´sci pomi˛edzy procesorami logicznymi mo˙ze być niezb˛edna dla prawidłowej realizacji zada-nia stawianego modułowi zarz ˛adzania zu˙zyciem energii pro-cesora. Informacja na ten temat dostarczana jest przez obiekt `_PSD` struktury opisuj ˛acej procesor logiczny w tablicy ACPI.

Bez wzgl˛edu na to czy istniej ˛a zale˙zności pomi˛edzy procesorami logicznymi, ka˙zdy z nich dysponuje swoim rejestrem modyfikuj ˛acym poziom wydajności. Potencjalnie mo˙ze to doprowadzi´c do sytuacji, w której zawartość rejestrów procesorów z jednej domeny cz˛estotliwościowej nie jest spójna. Takie sytuacje obsługiwane s ˛a przez odpowiednie mechanizmy koordynacji. Standard ACPI pozostawia producentom w tej kwestii pewn ˛a swobod˛e. Urz ˛adzenie mo˙ze zawiera´c wewn˛etrzny mechanizm koordynacji lub oczekiwać odpowiedniego zarz ˛adzania od systemu operacyjnego. Informacja o tym, który z wymienionych wariantów jest realizowany w ramach danego urz ˛adzenia, dost˛epna jest równie˙z w obiekcie `_PSD`. Odpowiednie pole przyjmuje jedn ˛a z poni˙zszych warto´sci:

- `HW_ALL`: procesor ma wbudowane mechanizmy koordynacji stanów wydajności w obrębie domeny;
- `SW_ALL`: wymagana jest koordynacja na poziomie systemu operacyjnego, nowa wartosc stanu ´wyda-jnosci dotycz ´aca domeny cz˛estotliwosciowej músi byc wprowad´zona do odpowiedniego rejestru wszystkich procesorów logicznych w ramach tej domeny;
- `SW_ANY`: wymagana jest koordynacja na poziomie systemu operacyjnego, nowa wartość stanu wydajności dotycz ˛aca domeny cz˛estotliwo´sciowej wprowadzana jest do odpowiedniego rejestru dowolnego z procesorów logicznych w ramach tej domeny.

W przypadku wbudowanej koordynacji sprzętowej standard ACPI nie definiuje ˙zadnych wytycznych dla sposobu realizacji tego mechanizmu. Nie istnieje równie˙z ˙zaden obiekt ACPI zawieraj ˛acy informacj˛e o sposobie wyboru ostatecznie wprowadzanego stanu P. W praktyce´ ´stosuje si˛e mechanizmy oparte o głosowanie. Zawartosc odpowiedniego re-jestru ka˙zdego z procesorów logicznych dźiałaj ˛acych w ra-mach tej samej domeny cz˛estotliwosciowej traktowaná jest jako ˙z ˛adanie odpowiedniego poziomu wydajnosci. Ostate-cznie w obr˛ebie domeny cz˛estotliwosciowej wprowadzany jest stán P odpowiadaj ˛acy najwy˙zszej z ˙z ˛adańych wydajnosci [9]. Stosowane są również różne podejscia w kwestii prawa głosu procesorów, które są w różnych stanach uspienia C.

W tym miejscu warto równie˙z, przy´ okazji, zwrócic uwag˛e na potrzeb˛e uwzgl˛ednienia w procesie implemen-

np. ze stabilizacją pracy procesorów oraz monitorowaniem obecności ukrytych procesów złośliwego oprogramowania [11, 12].

Strategie oszczędzania energii

Oszczędność energetyczną systemu komputerowego można uzyskać poprzez wyłączenie komponentu lub ograniczenie jego wydajności. Obie z wymienionych technik znajdują swoje odzwierciedlenie w systemach zarządzania energią współczesnych wielordzeniowych procesorów.

W czasie kiedy rdzeń procesora nie ma przydzielonych żadnych zadań system operacyjny może podjąć decyzje o jego uśpieniu. W przypadku systemów zgodnych ze standardem ACPI samo uśpienie podlega gradacji. Co istotne, im głębszy poziom uśpienia rdzenia, tym mniejsze zużycie energii, ale jednocześnie dłuższy czasu potrzebny na wybudzenie dodatkowo zwiększający prawdopodobieństwo utraty zawartości części podręcznych informacji [5].

Brak jakichkolwiek zadań jest skrajnym przypadkiem, w którym rdzeń może zostać uśpiony. W pozostałych, od częściowego obciążenia aż do pełnego, oszczędność energetyczna może zostać uzyskana przez zastosowanie jednej z poniższych strategii.

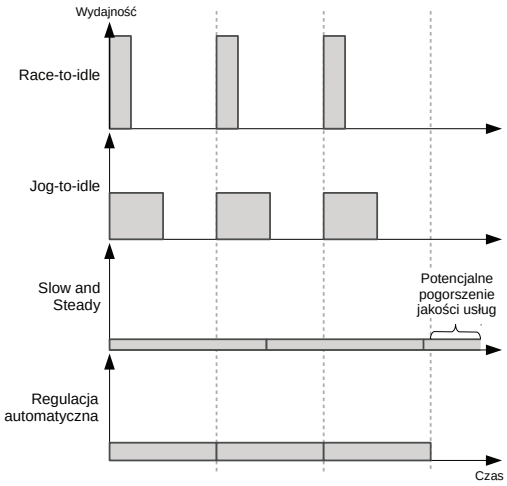
Race-to-idle. Procesor pracuje z maksymalną możliwą szybkością wykonywania instrukcji dążąc do wykonania przydzielonych zadań w możliwie krótkim czasie. Po wykonaniu przydzielonych zadań procesor może zostać uśpiony na dłuższy czas.

Jog-to-idle. Strategia wykorzystuje informacje o punktach pracy urządzenia, dla których wydajność energetyczna osiąga maksimum, lokalne bądź globalne [13]. Administrator dobiera spośród tego zbioru taki punkt, który zapewnia wydajność minimalnie przekraczającą przewidywane maksymalne obciążenie. Procesor wykonuje przydzielone zadania z tak zdefiniowaną wydajnością, a po ich zakończeniu wprowadzany jest w stan uśpienia. Podejście to wymaga dobrej znajomości charakterystyk platformy sprzętowej i obciążenia.

Slow and steady. Celem strategii jest unikanie warunków koniecznych dla przejścia procesora w stan uśpienia. Efekt ten można uzyskać np. dążąc do maksymalizacji obciążenia rdzeni zadaniami [9].

Regulacja automatyczna. Podejście to przewiduje wykorzystanie algorytmów dobierających szybkość pracy procesora do bieżącego zapotrzebowania w oparciu o mechanizm sprzężenia zwrotnego. W dalszej części opracowania w odniesieniu do rozwiązań z tej puli stosowane będzie określenie regulator.

Stany uśpienia, szczególnie głębokiego, cechują się minimalnym zużyciem energii. Z tego powodu strategia *race-to-idle* może być optymalnym rozwiązaniem dla niektórych zadań, szczególnie działających w ramach starszych platform sprzętowych [14, 15]. Współczesne procesory bazują na mechanizmach, wobec których koszt wielokrotnego przechodzenia do stanów uśpienia jest większy niż utrzymywanie procesora na pośrednim poziomie wydajności. W tym kontekście rozwiązania bazujące na regulatorach wydajności wskazywane są jako pozwalające na uzyskanie wyników bliższych optymalnym [13, 7, 17, 16]. Rozwiązania typu *jog-to-idle* oraz *slow and steady* mogą okazać się słuszne w przypadku dobrze zdefiniowanych, stałych w czasie obciążeń. W takim przypadku możliwe jest zdefiniowanie stałego poziomu wydajności pozwalającego zapewnić oczekiwaną jakość usług



Rys. 2. Wizualizacja przebiegów wydajności procesora w czasie, charakterystycznych dla każdej z wymienionych strategii. Pole zawarte pod wykresami poziomów wydajności dla każdego z wykresów jest takie samo, wykonana praca jest identyczna w każdym z przypadków.

świadczonych przez oprogramowanie działające w ramach systemu komputerowego. Rys. 2 wizualizuje modelowe przebiegi wydajności procesora w czasie, ilustrując koncepcje każdej ze strategii.

Sterowanie i koordynacja stanów wydajności procesora w systemie Linux

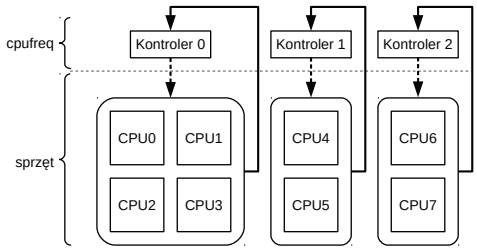
Jądro systemu operacyjnego Linux implementuje mechanizmy ACPI [6], zawiera ono również modularny mechanizm zarządzania energią w ramach modułu `cpufreq`[1]. Moduł ten cechuje się dużą elastycznością w kontekście obsługiwanych architektur sprzętowych. Wspiera on systemy zgodne ze specyfikacją ACPI, udostępnia również moduły obsługi konkretnych systemów wbudowanych opartych o architektury ARM, SPARC i inne. Dostosowanie modułu do pracy z nową architekturą wymaga utworzenia nowego sterownika odpowiedzialnego za komunikację ze sprzętem. Przykładem takiego sterownika jest `acpi-cpufreq` odpowiedzialny za obsługę systemów komputerowych zgodnych ze standardem ACPI.

Część dostępnych na rynku procesorów ma wbudowane mechanizmy zarządzające wydajnością poprzez skalowanie częstotliwości. W takim przypadku moduł `cpufreq`, nie steruje pracą procesora, jedynie definiuje górny i dolny limit częstotliwości. Limity te definiowane są w ramach polityki. Z punktu widzenia modułu `cpufreq` układ wielordzeniowy jest nierozróżnialny wobec obecności w systemie wielu niezależnych procesorów fizycznych. W takich przypadkach definiowanych jest wiele polityk dla każdego niezależnie obsługiwanego rdzenia logicznego lub zespołu rdzeni.

W przypadku procesorów, w których autonomiczne mechanizmy skalowania nie są dostępne lub zdecydowano się je wyłączyć, rola `cpufreq` jest znacznie większa. Do polityki każdej z domen częstotliwościowych przypisywany jest algorytm regulatora skalującego częstotliwość. Moduły implementujące te algorytmy nazywane będą dalej kontrolerami (ang. *governor*). W systemie Linux, niezależnie od architektury sprzętowej, dostępnych jest sześć kontrolerów:

***performance*, *powersave*, *userspace*:** ustawiają poziom wydajności procesora na poziomach odpowiednio najwyższym, najniższym oraz zdefiniowanym przez użytkownika;

[1] Przedstawiony opis został opracowany w oparciu o źródła jądra systemu Linux w wersji 4.11.



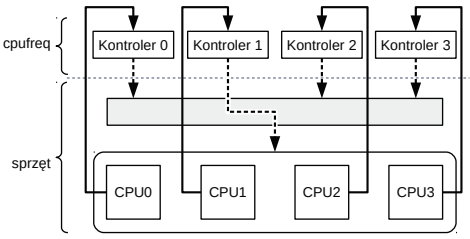
Rys. 3. Przykładowa konfiguracja mechanizmu `cpufreq` dla systemu wielordzeniowego o trzech domenach częstotliwościowych w przypadku realizacji koordynacji poziomów wydajności w wariancie `SW_ALL` lub `SW_ANY`. Linia przerywana – sygnał sterujący, linia ciągła – sprzężenie zwrotne.

***ondemand*, *conservative*, *schedutil*:** ustawiają poziom wydajność procesora w zależności od obserwowanego obciążenia.

Ostatnim komponentem składowym `cpufreq` są wspomniane wcześniej sterowniki procesora. Są to moduły dedykowane konkretnym architekturom sprzętowym, pozwalające na komunikację z konkretnym procesorem w ramach zdefiniowanego API niezależnego od architektury.

Moduł `cpufreq` realizuje swoje zadania stosownie do wymaganego scenariusza koordynacji przy uwzględnieniu przyjętych przez producenta procesora założeń dotyczących mechanizmu koordynacji stanów wydajności rdzeni w ramach domeny częstotliwościowej. Informacja o strukturze domen jest wykorzystywana wyłącznie wtedy, gdy wymagana jest koordynacja na poziomie systemu operacyjnego (`SW_ALL` lub `SW_ANY`). Każda z domen częstotliwościowych jest wtedy obsługiwana przez jedną politykę, a co za tym idzie jeden kontroler. Zostało to zwizualizowane na Rys. 3. Linią przerywaną oznaczono sygnał sterujący będący żądaniem wydajności wypracowanym przez kontroler. Linią ciągłą oznaczono sygnał sprzężenia zwrotnego. Na potrzeby prowadzonej dyskusji, wystarczające jest zdefiniowanie sygnału sprzężenia zwrotnego jako pewnej miary efektywności wykorzystania zasobów sprzętowych procesora przy bieżącym obciążeniu przez aplikacje użytkownika. Każdy z kontrolerów obserwuje odpowiedź obsługiwanej grupy procesorów logicznych na wyznaczoną przez siebie wartość sygnału sterującego.

W przypadku procesora wykorzystującego wbudowane mechanizmy koordynacji (`HW_ALL`) informacja o strukturze domen częstotliwościowych jest ignorowana, każdy procesor logiczny traktowany jest jak niezależna domena częstotliwościowa obsługiwana przez niezależny kontroler. Na każdy rdzeń logiczny w ramach domeny częstotliwościowej przypada jeden kontroler. Ostatecznie dla całej domeny częstotliwościowej zastosowany zostaje sygnał sterujący wybrany przez wbudowane mechanizmy koordynacji, pozostałe sygnały zostają zignorowane. Każdy z kontrolerów w ramach sprzężenia zwrotnego obserwuje wpływ zastosowanej wartości sygnału sterującego na efektywności wykorzystania zasobów obsługiwanego rdzenia logicznego. Nie dysponują one jednak wartościami żądań poziomów wydajności wyznaczonymi przez pozostałe kontrolery. Nie jest zatem możliwe określenie przy jakim rzeczywistym poziomie wydajności uzyskano obserwowany poziom efektywności wykorzystania zasobów. W tym przypadku kontrolery obserwują zatem odpowiedź obiektu sterowania na nieznany sygnał sterujący.



Rys. 4. Przykładowa konfiguracja mechanizmu `cpufreq` dla systemu wielordzeniowego o jednej domenie częstotliwościowej w przypadku realizacji koordynacji poziomów wydajności w wariancie `HW_ALL`. Linia przerywana – sygnał sterujący, linia ciągła – sprzężenie zwrotne, szary blok – wbudowany koordynator poziomów wydajności.

Podsumowanie

Przedstawione opracowanie stanowi wstęp do zagadnienia energooszczędnego sterowania wydajnością pracy procesora. Omówiono podstawowe rodzaje strategii ograniczania poboru mocy procesora oraz przytoczono argumenty przemawiające za zasadnością stosowania algorytmów regulacji automatycznej do rozwiązywania tej klasy problemów. Przeanalizowano specyfikację standardu ACPI pod kątem definicji interfejsów i ograniczeń narzucanych na rozwiązania sprzętowe, mających znaczenie w kontekście modyfikacji poziomów wydajności procesora z poziomu systemu operacyjnego komputera. Przedstawiono moduł `cpufreq` jądra systemu Linux wraz z jego komponentami składowymi. Istotna część funkcjonalności modułu bywa definiowana nie przez warstwę programową a przez mechanizmy wbudowane procesora. Nie znając szerszego kontekstu, opierając się wyłącznie na źródłach systemu Linux, można dojść do błędnych wniosków na temat sposobu realizacji funkcjonalności sterowania poziomami wydajności procesora w ramach tego systemu. Szczególny nacisk położony został na zaprezentowanie sposobu kooperacji pomiędzy modułem `cpufreq` a wbudowanymi mechanizmami sprzętowymi procesorów wielordzeniowych, w przypadku których występują zależności pomiędzy poziomami wydajności pracy zespołów rdzeni logicznych.

***Autorzy**: inż. Michał Getka, dr inż. Michał Karpowicz, Instytut Automatyki i Informatyki Stosowanej, Politechnika Warszawska, ul. Nowowiejska 15/19, 00-665 Warszawa, email: m.getka@stud.elka.pw.edu.pl email: m.karpowicz@elka.pw.edu.pl*

LITERATURA

[1] P. Arabas, M. Karpowicz, "Server power consumption: measurements and modeling with MSRs," *Challenges in Automation, Robotics and Measurement Techniques*. Springer, 2016, s. 233–244.

[2] M. P. Karpowicz, "Energy-efficient CPU frequency control for the Linux system," *Concurrency and Computation: Practice and Experience*, tom 28, nr 2, s. 420–437, 2016, cpe.3476. [Online]. Dost˛epne: http://dx.doi.org/10.1002/cpe.3476

[3] M. P. Karpowicz, P. Arabas, E. Niewiadomska-Szynkiewicz, "Energy-aware multilevel control system for a network of Linux software routers: design and implementation," *Systems Journal, IEEE*, tom PP, nr 99, s. 1–12, 2015.

[4] E. Niewiadomska-Szynkiewicz, A. Sikora, P. Arabas, M. Kamola, M. Mincer, J. Kołodziej, "Dynamic power management in energy-aware computer networks and data intensive computing systems," *Future Generation Computer Systems*, tom 37, s. 284–296, 2014. [Online].

[5] "ACPI Specification Document," www.acpi.info.

[6] L. Brown, "ACPI in Linux," *Linux Symposium*, vol. 51, 2005.

[7] E. Le Sueur, G. Heiser, “Slow down or sleep, that is the question.” in *USENIX Annual Technical Conference*, 2011.
[8] “Enhanced Intel SpeedStep Technology for the Intel Pentium M Processor,” Intel, 2004.
[9] C. Gough, I. Steiner, W. Saunders, *Energy efficient servers: blueprints for data center optimization*. Apress, 2015.
[10] “IA-64 and IA-32 Architectures Software Developer’s Manual,” www.intel.com.
[11] P. Malec, A. Piwowar, A. Kozakiewicz, K. Lasota, “Detecting security violations based on multilayered event log processing,” *Journal of Telecommunications and Information Technol-ogy*, nr 4, s. 30, 2015.
[12] M. Amanowicz, J. Jarmakiewicz, A. Kozakiewicz, M. Karpow-icz, T. Włodarczyk, “Cyberbezpieczenstwo systemów zarz ˛adza-nia sieci ˛a elektroenergetyczn ˛a,” *Przegl ˛ad Telekomunikacyjny+ Wiadomości Telekomunikacyjne*, 2014.
[13] D. H. Kim, C. Imes, H. Hoffmann, “Racing and pacing to idle: theoretical and empirical analysis of energy optimization heuristics,” *Cyber-Physical Systems, Networks, and Applications (CP-SNA), 2015 IEEE 3rd International Conference on*. IEEE, 2015, s. 78–85.
[14] A. Miyoshi, C. Lefurgy, E. Van Hensbergen, R. Rajamony, R. Rajkumar, “Critical power slope: understanding the runtime effects of frequency scaling,” *Proceedings of the 16th international conference on Supercomputing*. ACM, 2002, s. 35–44.
[15] E. Le Sueur, G. Heiser, “Dynamic voltage and frequency scaling: The laws of diminishing returns,” *Proceedings of the 2010 international conference on Power aware computing and sys-tems*. USENIX Association, 2010, s. 1–8.
[16] P. Arabas and M. Karpowicz, “Wykorzystanie informacji z rejestrów procesora do identyfikacji modelu poboru mocy przez serwer,” *Przegl ˛ad Elektrotechniczny*, R. 92, nr 3, s. 34–41, 2016.
[17] M. P. Karpowicz, P. Arabas, E. Niewiadomska-Szynkiewicz, “Design and implementation of energy-aware application-specific CPU frequency governors for the heterogeneous distributed computing systems,” *Future Generation Computer Systems*, 2016. [Online].