Zeszyty Naukowe Akademii Morskiej w Gdyni
Scientific Journal of Gdynia Maritime University

Nr 103/2018, 137–150
ISSN 2451-2486 (online)
ISSN 1644-1818 (printed)
DOI: 10.26408.103.10

Złożony/submitted: 23.12.2017
Zaakceptowany/accepted: 08.02.2018
Opublikowany/published: 31.03.2018

PRZEGLĄD PROGRAMÓW KOMPUTEROWYCH DEDYKOWANYCH PROJEKTOWANIU UKŁADÓW ELEKTRONICZNYCH. CHARAKTERYSTYKA ŚRODOWISKA GECKOCIRCUITS

REVIEW OF COMPUTER PROGRAMS DEDICATED TO THE DESIGN OF ELECTRONIC SYSTEMS. CHARACTERISTICS OF THE GECKOCIRCUITS ENVIRONMENT

Wojciech Stasiak*, Ireneusz Czarnowski
Akademia Morska w Gdyni, Morska 81-87, 81–225 Gdynia,
Wydział Przedsiębiorczości i Towaroznawstwa,
Katedra Systemów Informacyjnych, e-mail: wojtek_stasiak@wp.pl
* Adres do korespondencji/Corresponding author

Streszczenie: Rozwój technologii wytwarzania układów elektronicznych, a także coraz szerszy zakres ich zastosowań, prowadzi do zapotrzebowania na rozwiązania wspomagające ich projektowanie. W artykule dokonano przeglądu wybranych rozwiązań dedykowanych projektowaniu układów elektronicznych. Skoncentrowano się na rozwiązaniach opartych na projektowaniu komputerowym, porównując wybrane środowiska programowe. W szczególności uwagę poświęcono środowisku GeckoCIRCUITS, które omówiono na przykładzie projektu układu przekształtnika podwyższającego napięcie (*boost converter*).

Słowa kluczowe: projektowanie wspomagane komputerowo, GeckoCIRCUITS, projektowanie układów elektronicznych, przekształtnik *boost*.

Abstract: The development of electronics manufacturing technologies, as well as increasingly broad range of their applications, leads to an increase in the demand for solutions that support their design process. The article reviews selected solutions dedicated to the design of electronic circuits and focuses on computed-based solutions, comparing the selected software environments. Particular attention has been paid to the GeckoCIRCUITS environment, which has been discussed basing on example of an electronic boost converter circuit design.

Keywords: computer aided design, GeckoCIRCUITS, design of electronic circuits, boost converter.

1. WSTĘP

Projektowaniem układów elektronicznych nazywa się tworzenie obwodów, składających się z elementów zarówno biernych, jak i czynnych (tj. m.in. źródła prądowe lub napięciowe), wyłączników lub odbiorników, które, tworząc drogę dla prądu

elektrycznego, realizują określoną funkcję [Bolkowski 2012]. Proces komputerowego projektowania układów elektronicznych wiąże się z przedstawianiem układu w graficznej postaci, za pomocą schematu (np. ideowego, elektrycznego lub montażowego). Każdy jego element identyfikuje się odpowiednim dla siebie symbolem graficznym wraz z informacją o parametrach. W procesie komputerowego projektowania pozwala to często na późniejszą edycję poszczególnych komponentów, co ułatwia proces projektowania układu.

Proces tworzenia obwodów elektronicznych nie jest procesem prostym. Już na etapie projektowania układów bierze się pod uwagę nie tylko właściwości fizyczne materiałów, z których będą wykonane elementy układu, ale uwzględnia się inne istotne parametry, wynikające z założeń projektowanego układu, takie jak: rozmiar, waga, niezawodność oraz inne czynniki i ograniczenia. Proces projektowania układów elektronicznych powinien być często postrzegany poprzez pryzmat problemu optymalizacyjnego, tj. pryzmat wyznaczania wielkości optymalnych bądź najlepszych możliwych, ze względu na określoną funkcję oceny (zwaną też funkcją celu), dla projektowanego układu elektronicznego wraz z uwzględnieniem ewentualnych zdefiniowanych ograniczeń [Kovacevic, Friedli i Kolar 2011]. Wraz z rozwojem zastosowań układów elektronicznych oraz ze względu na fakt, że projektowanie ich może być trudne w sensie obliczeniowym, zasadne jest poszukiwanie narzędzi wspomagających ich projektowanie.

Poza samym projektowaniem, nie tylko układów elektronicznych, istotna jeszcze na etapie przedwdrożeniowym staje się weryfikacja zachowania obiektu. Weryfikację tę przeprowadza się poprzez symulację – obecnie symulację komputerową. Symulacja komputerowa pozwala na wnioskowanie o zachowaniu obiektów rzeczywistych, na podstawie obserwacji wyników działania programów komputerowych, badających te układy. Symulacja komputerowa jest tym bardziej przydatna, gdy obserwowanie rzeczywistych układów lub urządzeń na etapie projektowania jest niemożliwe lub trudne i kosztowne [Stupar 2012]. Pozwala także na weryfikację działania projektowanego układu elektronicznego, ze względu na przyjęte założenia oraz ustalone wartości parametrów, gdy problem projektowania układu jest trudny w sensie obliczeniowym i istnieje przeświadczenie, że założenia i ustalone wartości parametrów nie są optymalne, a być może bliskie optymalnym.

Symulacja komputerowa daje zatem możliwości weryfikacji projektowanego układu pod kątem wielu istotnych czynników, ważnych z punktu widzenia domeny jego zastosowania. Z punktu widzenia procesu projektowania symulacja komputerowa jest również postrzegana jako jeden z jego etapów. Takie podejście pozwala na natychmiastowe testowanie nowych konfiguracji układów. Symulacja pozwala na wykrycie potencjalnych błędów oraz ocenę właściwości roboczych układów. Umożliwia też wprowadzanie poprawek. Takie podejście do projektowania, poprzez równoległą symulację pracy projektowanych układów, nie generuje strat i kosztów występujących podczas wielokrotnych testów zbudowanego na próbę układu oraz

jego każdorazowego przebudowywania. Z drugiej strony pozwala na odkrywanie nowych, wcześniej niezidentyfikowanych czynników związanych z projektowanym układem, a które bez użycia opisywanego oprogramowania można by przeoczyć. Atutem symulacji komputerowej jest także eliminacja kosztów, związanych z użyciem często wielu niezbędnych urządzeń, np. do pomiaru, zasilania oraz obciążenia. Dzięki obliczeniom numerycznym, które są składową komputerowego projektowania, istnieje możliwość nie tylko realizowania obliczeń dla samej symulacji projektowanych układów oraz rzeczywistych środowisk ich pracy. Umożliwiają one bowiem również wizualizację parametrów pracy projektowanych układów, a także wizualizację wybranych cech projektowanych układów lub wartości dla wybranych ich parametrów.

Pomimo wielu walorów projektowania opartego na symulacji należy pamiętać, że symulacja komputerowa sama w sobie jest ograniczona i może nie odzwierciedlić nieznanych, ukrytych czynników, mających istotny wpływ na zachowanie układu w rzeczywistych warunkach pracy. Za przykład można tu wskazać rozbudowane procesy, w których po to, by komputer poradził sobie z obliczeniami towarzyszącymi symulacjom modelu układu elektronicznego, obliczenia dotyczące mocy systemu elektronicznego w ramach procesu symulacji komputerowej muszą zostać uproszczone. W takiej sytuacji „ostatnie słowo", tj. weryfikacja dokładności i przydatności wyniku symulacji, często należy do inżyniera projektu i zależy od jego doświadczenia i wiedzy. Dla przykładu, wymagania stawiane układom energoelektronicznym odnoszą się m.in. do gęstości mocy (tj. ilości mocy na jednostkę objętości, wyrażanej w W/m^3, mającej bezpośredni wpływ na rozmiary i objętość układu), wagi, sprawności, niezawodności, a także relatywnie niskich kosztów produkcji [Stupar 2012]. Zaprojektowanie optymalnego układu, przeznaczonego do wykonywania zadanej pracy wymaga zatem dopasowania nie jednego, a wielu jego parametrów. Dlatego też, jak wspomniano powyżej, programy wykorzystywane dziś do symulacji układów elektronicznych – w tym energoelektronicznych – powinny cechować się wielofunkcyjnością, w szczególności zaś powinny umożliwiać optymalizację ich parametrów, przy zadanych uprzednio celach i ograniczeniach. Innymi słowy, ich najważniejszym zadaniem jest wielopłaszczyznowe wsparcie procesu projektowania układów elektronicznych.

Niniejszy artykuł ma charakter przeglądowy. Celem artykułu jest charakterystyka wybranych narzędzi komputerowego projektowania układów elektronicznych, a przede wszystkim przedstawienie środowiska programowego GeckoCIRCUITS. Przedstawiono tu szerzej zastosowanie GeckoCIRCUITS w projektowaniu układów elektronicznych na wybranym przykładzie.

W następnej części artykułu przedstawiono zagadnienie komputerowego wspomagania projektowania układów elektronicznych. W części 3 dokonano przeglądu wybranych programów służących do projektowania i analizy układów elektronicznych, natomiast część 4 zawiera opis zastosowania GeckoCIRCUITS. Ostatnia z części artykułu stanowi podsumowanie.

2. KOMPUTEROWE WSPOMAGANIE PROJEKTOWANIA UKŁADÓW ELEKTRONICZNYCH

Lata 70. XX wieku stanowią początek myślenia o projektowaniu systemów elektronicznych poprzez pryzmat procesów automatyzacji. Mimo to zainteresowanie wykorzystaniem komputerowych narzędzi w procesie projektowania CAE (*Computer Aided Engineering*) oraz EDA (*Electronic Design Automation*) datowane jest na lata 60. ubiegłego stulecia. Przejawem tego zainteresowania było chociażby zainicjowanie corocznych konferencji DAC (*Design Automation Conference*). DAC jest obecnie najstarszą i największą konferencją poświęconą automatyzacji procesów projektowania urządzeń i układów elektronicznych lub inaczej nazywanemu, komputerowemu wspomaganiu projektowania elektroniki (wspomniane EDA) [*Design Automation Conference*]. Następne lata przyniosły szereg opracowań i narzędzi dedykowanych projektowaniu elektroniki. Ważną rolę odegrały tu takie korporacje, jak Hewlett Packard, Tektronix czy Intel.

Odwołując się do historii projektowania układów elektronicznych, należy stwierdzić, że wraz z rozwojem elektroniki, w tym nowych technologii półprzewodnikowych, można było w następstwie obserwować rozwój specjalistycznych i dedykowanych narzędzi automatyzujących procesy projektowania oraz symulacji układów elektronicznych w obszarze ich wdrożeń.

Programy do projektowania układów elektronicznych oparte są na różnych językach programowania i środowiskach obliczeniowych (np. C/C++, Java, MatLab, Spice, Python, Skill, TCL, ULP, VHDL-AMS i inne). Wymienione środowiska pozwalają na tworzenie schematów elektronicznych, a następnie ocenę ich pracy poprzez symulacje i następujące po nich analizy. Część z programów charakteryzuje się dodatkowymi funkcjami projektowania obwodów drukowanych. Zastosowany w nich bogaty interfejs graficzny pomaga użytkownikowi w intuicyjnej obsłudze. W celu zwiększenia dostępności swoich produktów producenci oprogramowania do projektowania układów elektronicznych dbają o to, aby były one dostępne na wiele platform – w sensie systemów operacyjnych (tj. Windows, MacOS, Unix, Linux).

Obecnie można wskazać wiele przykładów środowisk obliczeniowych typu CAD (*Computer Aided Design*), wspomagających projektowanie elektroniki, np. OrCAD PSpice Designer [OrCAD *PSpice Designer*], oraz szereg środowisk typu CAE (*Computer Aided Engineering*) przeznaczonych do symulacji komputerowej, dedykowanych układom elektroniki (np. środowisko PUFF [Compton, Williams i Rutledge 1987]). Najczęściej jednak obie funkcje, tj. projektowania i symulacji, są integrowane w jedno środowisko softwarowe, np. CadSoft EAGLE [*Autodesk Eagle*].

Środowiska CAD i CAE należą do klasy narzędzi i systemów CAx (*Computer Aided Technologies*), wykorzystujących technologię komputerową do pomocy w wielu różnych aspektach zarządzania cyklem życia produktu PLM (*Product Lifecycle Management*). CAD oraz CAE stanowią część systemu komputerowo

zintegrowanego wytwarzania CIM (*Computer Integrated Manufacturing*), łączącego w sobie powiązane ze sobą funkcjonalnie poszczególne systemy CAx [Capanidis i Kowalewski 2012]. System komputerowo zintegrowanego wytwarzania CIM obejmuje zautomatyzowane systemy do planowania, projektowania, finansowania, wytwarzania, zaopatrzenia i sprzedaży w zintegrowanym komputerowo systemie [Capanidis i Kowalewski 2012].

Środowiska komputerowe EDA, przeznaczone do projektowania oraz symulacji komputerowej, cechują się m.in.:

- szybkością przetwarzania danych – cecha istotna zarówno ze względu na efektywną symulację, jak i sprawny dobór parametrów, zapewniających najefektywniejszą pracę zaprojektowanego układu;
- wysoką dokładnością – cecha ważna ze względu na konieczność uzyskiwania atrybutów projektowanego układu bliskich rzeczywistym, tj. mając na uwadze docelowe warunki i środowisko pracy układu;
- wielofunkcyjnością – cecha związana z dostępem do szerokiego zakresu możliwych funkcji, w tym do równoległego badania wielu parametrów, często niezależnych, oraz do optymalizacji parametrów projektowanego układu [Kolar 2014].

W ostatnich latach zauważa się trend na rynku oprogramowania typu EDA. Oprogramowanie typu EDA zostaje włączane do większych, wielofunkcyjnych środowisk projektowania, co jest odpowiedzią na wyzwania i oczekiwania użytkowników. Cecha wielofunkcyjności odwołuje się do środowisk uniwersalnych, pozwalających na badanie układów i ich symulację z punktu widzenia różnych grup parametrów układu, np. dotyczących samych obwodów, właściwości termicznych (obieg ciepła, dynamika płynów) czy elektromagnetycznych [Stupar 2012].

3. PRZEGLĄD WYBRANYCH PROGRAMÓW DO PROJEKTOWANIA I ANALIZY UKŁADÓW ELEKTRONICZNYCH

Projektanci urządzeń elektronicznych mają szeroki wybór narzędzi wspomagających. Narzędzia te często umożliwiają badanie cech projektowanych układów, symulując warunki ich implementacji. Uwzględniając wspomniane wyżej cechy środowiska EDA, wskazać można wiele programów komputerowych, wykazujących właściwości wynikające wprost z tych cech. Projektant, decydując o wyborze środowiska projektowania, uwzględnia wiele aspektów, w tym często dostępność tego środowiska ze względu na jego dystrybucję, mając na uwadze, że gdy jest ono udostępniane na zasadzie wolnej i bezpłatnej licencji, może mieć szereg ograniczeń.

W zależności od charakteru koncepcji dla projektowanego układu elektronicznego, projektant może wybrać oprogramowanie kompleksowe, rozbudowane o opracowanie schematu ideowego układu, symulowanie oraz analizowanie

schematów elektronicznych, a także płytek obwodów drukowanych. Przy prostszych projektach może wybrać mniej skomplikowane środowisko, lepiej dopasowane do wymogów projektu. Jego wybór wiąże się również z dostępnością oprogramowania na wybrane systemy operacyjne (tj. Windows, Unix, Linux, MacOS, itp.).

Poniżej scharakteryzowano wybrane środowiska projektowania, kierując się ich dostępnością oraz popularnością. Wśród środowisk tych znalazły się:

- CadSoft EAGLE [*Autodesk Eagle*];
- Labcenter Electronics Proteus Design Suite [*Labcenter Proteus* VSM];
- OrCAD PSpice Designer [OrCAD *PSpice Designer*];
- ANSYS Simplorer [*Ansys Simplorer*];
- GeckoCIRCUITS [GeckoCIRCUITS].

Środowisko CadSoft EAGLE (*Easily Applicable Graphical Layout Editor*) obejmuje program komputerowy, służący do projektowania układów elektrycznych, składający się z trzech modułów. Rysowanie schematu układu i sprawdzenie jego spójności odbywa się dzięki edytorowi schematów. Edytor płytek zaś pozwala na przenoszenie elementów układu i połączeń na płytkę, a także umożliwia sprawdzenie (poprzez analizę graficzną połączeń) tras sygnałów oraz dobranie długości ścieżek. Trzeci z modułów – Autorouter – pozwala na automatyczne poprowadzenie ścieżek na płytce [*Autodesk Eagle*].

CadSoft EAGLE jest określany mianem programu intuicyjnego, o przyjaznym interfejsie. Ponadto jest on programem dostępnym w wersji darmowej. Dane w tym programie zapisywane są w otwartym formacie XML, co stanowi dodatkowy atut, pozwalając współdzielić dane pomiędzy innymi aplikacjami.

W połowie 2016 roku producent środowiska – CadSoft – został przejęty przez potentata w dziedzinie oprogramowania do projektowania inżynierskiego – firmę Autodesk. Zakup CadSoft EAGLE zapewnił dywersyfikację możliwości tego oprogramowania, zarówno od strony projektu mechanicznego, jak i elektronicznego (MCAD/ECAD).

Ograniczona wersja programu EAGLE Light przeznaczona została do nauki oraz projektów niekomercyjnych [*Eagle Tutorial*]. Umożliwia ona projektowanie płytek drukowanych o wymiarach ograniczonych do 100×80 mm, z dwiema warstwami sygnałowymi.

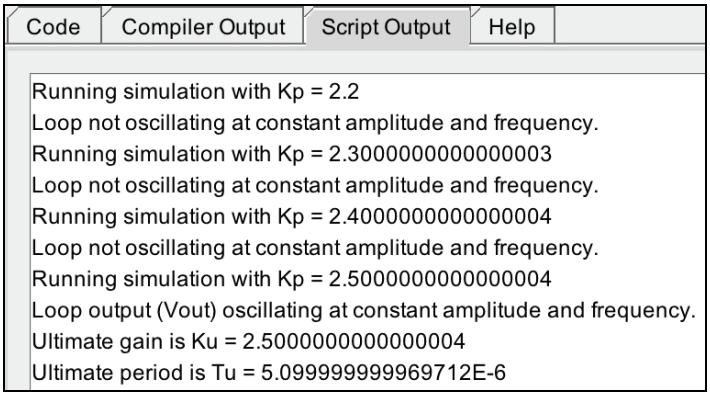
LEPDS (*Labcenter Electronics Proteus Design Suite*) stanowi pakiet oprogramowania, w którego skład wchodzą aplikacje:

- ISIS (*Intelligent Schematic Input System*) – komponent tworzenia schematów układów elektronicznych i symulacji ich działania;
- ARES (*Advanced Routing and Editing Software*) – komponent projektowania płytek drukowanych z funkcjami *Autorouting* oraz *Autoplacemen*t, ułatwiającymi użytkownikowi pracę, poprzez samodzielne ustalanie pozycji ścieżek łączących oraz nanoszonych elementów elektronicznych;

- VSM (*Virtual System Modeling*) – interfejs pozwalający na jednoczesną symulację obwodów elektronicznych, elementów animowanych oraz modeli mikroprocesorów. Ma on na celu analizę całego układu [*Labcenter Proteus* VSM].

LEPDS umożliwia projektowanie i symulowanie układów, opartych m.in. na mikrokontrolerach programowalnych (8051, ARM7, AVR, Motorola, PIC), układów analogowych, cyfrowych i mieszanych. Pakiet oprogramowania posiada wbudowaną bazę podzespołów elektronicznych i zawiera edytor obudów. Program umożliwia również projektowanie wielowarstwowych płytek drukowanych oraz zawiera narzędzia, służące do automatycznego rozmieszczania elementów, wraz z wytyczaniem połączeń elektrycznych, tzw. ścieżek. Program umożliwia także podgląd zaprojektowanych płytek w formie wizualizacji 3D.

Interfejs programu LEPDS z przykładowym projektem układu elektronicznego przedstawiono na rysunku 1.



Rys. 1. Interfejs wyników pośrednich oraz końcowych GeckoSCRIPT

Fig. 1. *Intermediate and final output values in the GeckoSCRIPT interface*

OrCAD PSpice Designer (*Personal Simulation Program with Integrated Circuit Emphasis*) [OrCAD *PSpice Designer*] stanowi zaawansowany program dedykowany projektowaniu i symulacji, przede wszystkim przeznaczony do projektowania obwodów zintegrowanych. Program ten jest kompatybilny ze środowiskiem Simulink programu MatLab. Kompatybilność umożliwia projektantom układów elektromechanicznych (tj. bloki sterujące, silniki, czujniki i konwertery mocy) wykonywanie symulacji, zawierających realistyczne modele elementów elektronicznych.

OrCAD PSpice Designer stanowi część pakietu programów OrCAD PCB Design Suite. Pakiet ten można również traktować jako zbiór uzupełniających się wzajemnie aplikacji, będących rozwinięciem opisywanego symulatora. Składa się m.in. z następujących aplikacji:

- OrCAD Capture CIS – aplikacja przeznaczona do projektowania schematów;
- OrCAD PCB Editor – edytor pozwalający na rozmieszczanie elementów projektowanego układu elektronicznego na płytce drukowanej;
- OrCAD PSpice A/D – aplikacja ułatwiająca interpretację wyników symulacji;
- OrCAD SI – pakiet funkcjonalny przeznaczony do analizy integralności sygnałów w projektowanym układzie.

OrCAD PSpice Designer został również wyposażony w moduł automatycznego dostrojenia parametrów układu elektronicznego w celu spełnienia indywidualnych wymagań projektowych. Zawiera on także bogatą bibliotekę różnych modeli komponentów składowych układów elektronicznych, wliczając około 30 tys. analogowych oraz mieszanych (analogowo-cyfrowych) modeli elementów elektronicznych.

Program OrCAD PSpice Designer posiada także funkcję DEDK (*Device Equations Developer's Kit*), umożliwiającą wdrażanie nowych i niestandardowych równań elementów wewnętrznych. Program pozwala na graficzną prezentację przebiegów symulacji wraz z ich zapisem i dalszą analizą, jeśli jest to potrzebne. OrCAD PSpice Designer to również edytor modeli oraz edytor układów magnetycznych, pozwalający na dobór parametrów projektowanego układu elektronicznego ze względu na zachodzące w nim zjawiska magnetyczne [OrCAD *PSpice Designer*].

ANSYS Simplorer to intuicyjny, wielofunkcyjny program, pozwalający na symulację złożonych układów sterowanych elektronicznie. Technologia logiczna, na której oparto program, umożliwia przeanalizowanie wielu cech rozbudowanych systemów, poczynając od szczegółowych analiz charakterystyk poszczególnych elementów elektronicznych, zastosowanych w projektowanym układzie, po weryfikację wydajności zaprojektowanego układu elektronicznego [*Ansys Simplorer*]. Za pomocą aplikacji Simplorer, już w początkowych fazach tworzenia projektu układu elektronicznego, projektant może poprzez wyniki symulacji śledzić różne jego parametry oraz na bieżąco je korygować. Celem takiej funkcjonalności jest maksymalizowanie efektywności projektowania, czego korzyścią może być m.in. skrócenie czasu, licząc go od koncepcji projektu do wdrożenia układu [*Ansys Simplorer – Esterel*].

Simplorer łączy różnorodne techniki przedstawiania modelowanych układów (za pomocą obwodów elektronicznych, schematów blokowych, tzw. *state machines*, czy z poziomu równań) oraz języki programowania (VHDL-AMS, *Simplorer Modeling Language* oraz C/C+), które poszerzają możliwości projektowe, także przez możliwość programowania nowych komponentów oraz nowych funkcjonalności niezbędnych z punktu widzenia projektowanego układu [*Ansys Simplorer Brochure*].

Kolejnym przykładem oprogramowania, przeznaczonego do projektowania układów elektronicznych oraz symulacji ich pracy, jest GeckoCIRCUITS [IPES *Moodle*]. Środowisko to, w porównaniu do opisanych wcześniej oraz innych środowisk, posiada szereg dodatkowych zalet. Przede wszystkim jest ono przykładem kolejnej, nowej generacji oprogramowania, służącego zintegrowanemu projektowaniu i symulacji pracy układów elektronicznych. Środowisko charakteryzuje się ergonomicznym interfejsem, umożliwiającym także korzystanie z innych środowisk, np. MatLab (Simulink). Program sprawnie przeprowadza wszelkie zadane mu symulacje obwodów elektronicznych. W module symulacji środowiska GeckoCIRCUITS wspierana jest możliwość analizowania pól elektrycznych, magnetycznych i zjawisk elektromagnetycznych.

Środowisko GeckoCIRCUITS zostało oparte na języku JAVA. Interfejs programowy środowiska pozwala na użytkowanie tego programu (jego komponentów związanych z analizowanym układem elektronicznym) jako tzw. apletu osadzonego na stronie WWW, czyli poprzez przeglądarkę internetową. GeckoCIRCUITS nie wymaga zatem instalacji. Ze względu na użyty język programowania JAVA kod źródłowy programu potrzebuje jedynie środowiska uruchomieniowego *Java Runtime Environment*. Kod JAVA pozwala również na programowanie serwerów i apletów układów elektronicznych oraz prowadzenie niezależnych badań i symulacji tych układów poprzez interfejs HTML, co można postrzegać jako pewną funkcjonalność, dającą możliwość, np. pracy w projekcie w tzw. środowisku rozproszonym.

Ponadto, co jest z punktu widzenia poszerzania funkcjonalności projektowania i badania projektowanych układów bardziej korzystne, oprogramowanie firmy Gecko-Simulations wyposażono w pakiet GeckoSCRIPT, który umożliwia zaimplementowanie nowych skryptów – nowych funkcjonalności, oczywiście w języku JAVA.

Gecko-Simulations udostępnia również takie pakiety jak: GeckoMAGNETICS (pakiet służący do projektowania i analizy układów magnetycznych), GeckoEMC (pakiet służący do badania właściwości elektromagnetycznej układów) oraz GeckoHEAT (pakiet służący do badania zjawisk cieplnych w układach energoelektronicznych). Pakiety te mogą być wykorzystywane jako niezależne programy lub moduły uzupełniające dla GeckoCIRCUITS [Drofenik, Müsing i Kolar 2010]. Wielofunkcyjność połączonych pakietów czyni GeckoCIRCUITS konkurencyjnym, mogącym sprostać wielu wymagającym projektom. Dzięki pracy pozwalającej integrować poszczególne jego komponenty i moduły możliwa jest tzw. wielowątkowa symulacja. To rozwiązanie pozwala na dokładniejsze analizy, co stawia program w pozycji konkurenta do pozostałych programów tej klasy.

W kolejnej części artykułu przedstawiono opis projektowania wybranego, przykładowego układu elektronicznego opartego na środowisku GeckoCIRCUITS.

4. GECKOCIRCUITS – PRZYKŁAD ZASTOSOWANIA

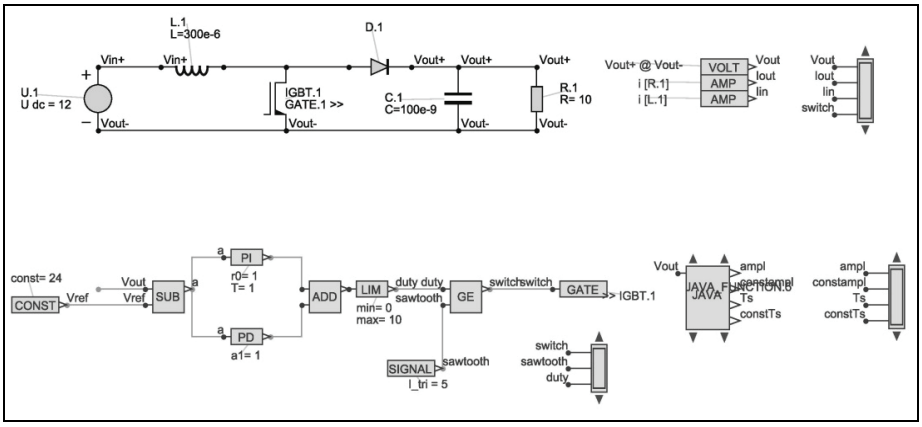
Jak wspomniano wcześniej, zastosowanie tradycyjnej metody iteracji do projektowania układów elektrycznych wymaga dużo wysiłku i czasu. Próba zastosowania innej topologii układu, zmiana trybu pracy, sterowania, systemu modulacji, czy też wymiana poszczególnych komponentów układu elektronicznego bardzo często wymaga rozpoczęcia procesu projektowania niemal od początku. Uzasadniona jest zatem potrzeba ulepszenia metody projektowania układów elektronicznych, w tym poszukiwanie nowych narzędzi.

Programowym narzędziem, pozwalającym sprostać wymogom wspomnianym powyżej, jest GeckoCIRCUITS, którego przykład zastosowania do zaprojektowania układu przekształtnika *boost* przedstawiono poniżej. Układ przekształtnika *boost* w praktyce znajduje zastosowanie m.in. w zasilaniu urządzeń prądu stałego, tj. jako zasilacz urządzeń informatycznych i multimedialnych, a także w hamowaniu rekuperacyjnym z odzyskiem energii KERS (*Kinetic Energy Recovery System*).

4.1. Opis problemu

Za przykład zastosowania GeckoCIRCUITS posłużył układ przekształtnika *boost*, którego zadaniem jest podnoszenie stałego DC (*Direct Current*) napięcia wejściowego na wyższe wyjściowe, również stałe.

Schemat układu, wraz z doborem nastaw regulatora proporcjonalno-całkująco-różniczkującego PID (*Proportional-Integral-Derivative controller*) metodą Zieglera-Nicholsa [Müsing 2014] przedstawiono w interfejsie projektowym GeckoCIRCUITS na rysunku 2 (schemat ideowy układu elektrycznego w górnej części, schemat sterowania układem – w dolnej części).



Rys. 2. Układ przekształtnika *boost* zaprezentowany w środowisku GeckoCIRCUITS

Fig. 2. *Boost converter layout presented in GeckoCIRCUITS*

Projektowany układ służy do podwyższania napięcia w zasilaczach impulsowych stałego napięcia oraz napędach z silnikami prądu stałego. Stosuje się w nim elementy półprzewodnikowe (tranzystor i dioda – oznaczenia odpowiednio IGBT.1 oraz D.1 na rys. 2) oraz co najmniej jeden element magazynujący energię (oznaczenie C.1 na rys. 2). W przekształtniku *boost* za przełączanie między stanem przewodzenia a stanem zaporowym odpowiedzialny jest zazwyczaj tranzystor MOSFET, IGBT lub BJT. Dodatkowe zastosowanie odpowiedniego filtru, składającego się z kondensatorów lub cewek, zapewnia zniwelowanie zniekształceń napięcia wyjściowego [Jankiewicz 2016].

Dla układu idealnego przekształtnika *boost* prawdziwa jest zależność wartości średniej napięcia wyjściowego od współczynnika wypełnienia sterowania tranzystora:

$$U_{wy} = U_{we}\left(\frac{1}{1-d}\right) \tag{1}$$

gdzie:

U_{wy} – napięcie uzyskiwane na wyjściu układu przekształtnika *boost*,
U_{we} – napięcie wejściowe,
d – tzw. współczynnik wypełnienia sterowania tranzystora.

W celu wyznaczenia wymaganego współczynnika wypełnienia, do uzyskania zadanego napięcia wyjściowego można więc zastosować wzór:

$$d = \frac{U_{wy} - U_{we}}{U_{wy}} \tag{2}$$

Powyższe zależności są właściwe przy założeniu m.in. zerowych spadków napięcia na elementach oraz zerowej rezystancji wewnętrznej zastosowanego źródła napięcia stałego, tj. baterie, akumulatory, ogniwa fotowoltaiczne, jak również prostowniki czy generatory prądu stałego.

W przykładzie implementacji postanowiono zmierzyć się z problemem automatycznego dopasowania parametrów sterujących układem przekształtnika *boost* tak, aby w jak najkrótszym czasie uzyskać stabilną pracę zaprojektowanego układu, objawiającą się poprzez stałość amplitudy i okresu oscylacji sygnału napięcia na wyjściu.

4.2. Opis implementacji układu przekształtnika *boost*

W metodzie tradycyjnej wzmocnienie regulatora sterowane jest manualnie, metodą iteracji, aż do uzyskania satysfakcjonujących charakterystyk pracy zaprojektowanego układu. W GeckoCIRCUITS ustalenie parametrów niezbędnych do pożądanej pracy układu może zostać osiągnięte zupełnie inną drogą – poprzez interfejs graficzny i programowy.

W prezentowanym przykładzie opis pracy projektowanego układu, opis poszczególnych elementów w projekcie – ich parametrów, w tym ustalenie ich wartości, znajduje się po stronie kodu projektu, który jest zapisem opartym na składni języka JAVA. Środowisko GeckoCIRCUITS pozwala na użycie i dodanie do projektu dedykowanych bloków funkcjonalnych (w praktyce bloków skryptowych), rozszerzających modelowanie projektowanych układów, jak i jego poszczególnych składowych. Przykładem takiego bloku funkcjonalnego jest blok oznaczony przez „JAVA_FUNCTION.5" (rys. 2).

W analizowanym przykładzie, skrypt zawarty w bloku funkcjonalnym JAVA (rys. 2) odpowiada za manipulowanie oraz wskazanie wahań amplitudy napięcia wyjściowego, odczytu stałości tychże oscylacji, a także ich okresu. Takie rozwiązanie pozwala na symulowanie pracy układu przy jednoczesnym poszukiwaniu optymalnych wartości wzmocnienia K_u oraz okresu oscylacji T_u, które później mogą zostać użyte do dalszych obliczeń.

W prezentowanym na rysunku 2 układzie zawarto również odpowiedni układ regulatora (bloki PI oraz PD), którego rolą jest zapewnienie poprawnej pracy przekształtnika. Użyty regulator PID odczytuje wartość napięcia uzyskanego na wyjściu, przyrównuje do pożądanej, a następnie steruje swoimi nastawami tak, aby napięcie wyjściowe uzyskało (i zachowało) właściwą wartość. Zastosowanie w nim metody Zieglera-Nicholsa pozwala na dobieranie współczynników regulatora PID (K_p, K_i, K_d – wzmocnienia części: proporcjonalnej, całkującej oraz różniczkującej) poprzez pomiar parametrów oscylacji K_u (współczynnik wzmocnienia układu) i T_u (okres oscylacji).

Dostęp do opisu funkcjonalnego poszczególnych elementów projektu, jak i jego bloków funkcjonalnych, w GeckoCIRCUITS odbywa się m.in. przez interfejs programistyczny GeckoSCRIPT. Ten interfejs pozwala na bezpośrednie zaprojektowanie pracy całego układu zgodnie z życzeniem projektanta i odpowiednio z zachowaniem strony merytorycznej projektu. Interfejs ten ułatwia również ustalenie poszczególnych kroków, etapów i parametrów dla całego procesu projektowego. Dedykowane funkcjonalności dla projektowanego układu lub projektowanej symulacji zapisuje się w odpowiedniej strukturze, zwanej strukturą funkcji.

Interfejs pozwala także na tworzenie bloków funkcjonalnych, które można zapisać jako nową funkcję w bibliotece programu, do wykorzystania w nowych projektach.

Interfejs programistyczny GeckoSCRIPT ułatwia opisanie poszczególnych kroków symulacji, w których możliwe jest np. mierzenie napięcia uzyskanego na wyjściu, a po przyrównaniu go do pożądanego poziomu, sterowanie – poprzez regulator PID – tranzystorem tak, aby na wyjściu przekształtnika *boost* użytkownik otrzymywał stabilne, wzmocnione w stosunku do źródła, napięcie prądu stałego. Takie rozwiązanie pozwala na uzyskanie pożądanych wartości na wyjściu regulatora.

4.3. Wyniki symulacji

W wyniku przeprowadzonej symulacji dobrano optymalne parametry dla regulatora PID pracującego na podstawie metody Zieglera-Nicholsa. Nastawy (K_p, K_i oraz K_d) zostają ustawione wstępnie na zero, po czym metodą iteracji zmieniana była wartość wzmocnienia K_u, do momentu spełnienia założenia. Po uzyskaniu stabilności amplitudy i okresu oscylacji, wzmocnienia części proporcjonalnej (K_p), całkującej (K_i) i różniczkującej (K_d) zostały wyliczone zgodnie z tabelą 1.

Oznacza to, że poprzez dobór nastaw regulatora PID uzyskano stabilnie pracujący układ przekształtnika *boost*.

Tabela. 1. Nastawy regulatora PID w metodzie Zieglera-Nicholsa

Table. 1. *PID controller parameters using Ziegler-Nichols method*

Typ regulacji	K_p	K_i	K_d
P	0.50 K_u	–	–
PI	0.45 K_u	1.2 K_p/P_u	–
PID	0.60 K_u	2 K_p/P_u	$K_pP_u/8$

Finalnie po zakończeniu symulacji, ale i w trakcie procesu symulacji, program poinformował użytkownika o każdym kroku i wynikach. Projektant układu elektronicznego na podstawie uzyskanego raportu ma możliwość wglądu w charakterystyki pracy układu. Interfejs programu GeckoCIRCUITS z wynikami pośrednimi pokazano na rysunku 1.

5. PODSUMOWANIE

W artykule przedstawiono opis wybranych środowisk programistycznych, przeznaczonych do projektowania układów elektronicznych. Szerzej zostało opisane środowisko GeckoCIRCUITS. Środowisko to przedstawiono również, wykorzystując skrócony opis jego implementacji do poszukiwania parametrów układu przekształtnika *boost*. Zaprezentowano także wybrane wyniki tej implementacji.

GeckoCIRCUITS jest nową klasą oprogramowania, stanowiącą alternatywę dla pozostałych, m.in. omówionych powyżej. Prosty interfejs GeckoCIRCUITS oraz łatwość implementacji nowych funkcjonalności, wynikająca przede wszystkim z możliwości, jakie dostarcza obiektowy język programowania JAVA, sprawiają, że przy uwzględnieniu wymogu znajomości języka JAVA przez programistę, środowisko wyróżnia się na tle konkurencyjnych środowisk.

Rozbudowa podstawowego środowiska o kolejne aplikacje z pakietu, tj. np. GeckoMAGNETICS, sprawia, że omawiany program spełnia wspomniane wymagania wielofunkcyjności stawiane oprogramowaniu EDA. Ta wielofunkcyjność

GeckoCIRCUITS pozwala na wykorzystanie programu zarówno w prostych, jak i bardziej złożonych projektach.

Niniejszy artykuł stanowi inspirację dla autorów do szerszego poznania środowiska, a w szczególności jego modułów dedykowanych rozwiązywaniu problemów optymalizacyjnych, związanych z projektowaniem układów elektronicznych, oraz prac związanych z poszerzeniem funkcjonalności GeckoCIRCUITS o nowe funkcje dedykowane poszukiwaniu optymalnych parametrów w projektach układów elektronicznych.

6. LITERATURA

Ansys Simplorer, http://www.ansys.com/Products/Systems/ANSYS-Simplorer.

Ansys Simplorer Brochure, http://www.anova.com.tr/dynamicContent/file/ansys-simplorer-brochure.pdf.

Ansys Simplorer – Esterel, http://www.esterel-technologies.com/products/ansys-simplorer/.

Autodesk Eagle, https://www.autodesk.com/products/eagle/overview.

Bolkowski, S., 2012, *Teoria obwodów elektrycznych*, Wydawnictwa Naukowo-Techniczne, Warszawa.

Capanidis, D., Kowalewski, P., 2012, *Przegląd systemów wspomagania procesów konstruowania i wytwarzania,* http://yadda.icm.edu.pl/baztech/element/bwmeta1.element.baztech-97607e36-a04a-4c25-a6c2-a13fd851357c/c/Capanidis_Kowalewski_Przeglad_1_2012.pdf.

Compton, R., Williams, W., Rutledge, D., 1987, *Puff, an Interactive Microwave Computer Aided Design Program for Personal Computers*, IEEE MTT-S International Microwave Symposium Digest, vol. 2, s. 707–708.

Design Automation Conference, https://dac.com/.

Drofenik, U., Müsing, A., Kolar, J.W., 2010, *Novel Online Simulator for Education of Power Electronics and Electrical Engineering*, https://www.pes.ee.ethz.ch/uploads/tx_ethpublications/drofenik_IPEC2010_education.pdf.

Eagle Tutorial, Version 6, https://www.egr.msu.edu/eceshop/pcb/V6_tutorial_en.pdf.

GeckoCIRCUITS, http://www.gecko-simulations.com/geckoCIRCUITS.html.

IPES *Moodle*, http://www.ipes.ethz.ch.

Jankiewicz, Z., 2016, *Układy energoelektroniczne***,** http://www.bezel.com.pl/index.php/urzadzenia-energoelektroniczne/uklady-energoelektroniczne.

Kolar, J., 2014, *Future Challenges for Research and Teaching in Power Electronics,* https://www.pes.ee.ethz.ch/uploads/tx_ethpublications/__Nottingham_2014_FINALFINAL_110714.pdf.

Kovacevic, I., Friedli, T., Kolar, J.W., 2011, *PEEC-Based Modeling of EMI Filters Incl. Parasitics and Geometrical Arrangement of Filter Elements*, https://www.pes.ee.ethz.ch/uploads/tx_ethpublications/10_Kovacevic_PEEC-Based_Modelling_of_EMI_Filters_01.pdf.

Labcenter Proteus VSM, http://www.labcenter.com/products/vsm/vsm_overview.cfm.

Müsing, A., 2014, *Script-based Simulation Control using GeckoSCRIPT,* http://www.gecko-simulations.com.

OrCAD *PSpice Designer,* http://www.orcad.com/products/orcad-pspice-designer/overview.

Stupar, A., 2012, *Simulation-Based Design and Optimization of Power Electronic Systems*, http://www.corpe.et.aau.dk/digitalAssets/59/59025_corpe---eth---stupar--andrija.pdf.