Technika cyfrowa,
Zjawiska szkodliwe,
Symulacja

PNIEWSKI Roman [1]
KORNASZEWSKI Mieczysław[2]

ALGORYTMY SYMULACJI UKŁADÓW CYFROWYCH

Współczesne systemy sterowania z zastosowaniem techniki cyfrowej, ze względu na znaczne zwiększenie stopnia integracji wymagają weryfikacji poprawności układu cyfrowego już na etapie projektu wstępnego. Jest to realizowane przez symulację opisu logicznego układu. W artykule przedstawiono metody symulacji systemów cyfrowych.

ALGORITHMS OF THE DIGITAL CIRCUITS SIMULATION

Modern control systems using digital techniques, due to the significant increase in the degree of integration require validation of a digital circuit at the stage of preliminary design. This is done by simulating the logical description of the system. The article presents methods for simulation of digital systems.

1. WSTĘP

Od współczesnych systemów sterowania cyfrowego wymaga się aby działały niezawodnie. Szczególnie istotne jest bezawaryjne działanie cyfrowych systemów SRK (Sterowanie Ruchem Kolejowym). Systemy te charakteryzują się dużą redundancją, niezbędną ze względu na wymagane bezpieczeństwo (poprawne działanie lub doprowadzenie do stanu bezpiecznego przy dowolnym uszkodzeniu). Dodatkowo, rozbudowa układów sterowania cyfrowego znacznie utrudnia weryfikację poprawności działania systemów. Opracowanie niezawodnych systemów cyfrowych wymaga weryfikacji poprawności działania układu już we wstępnej fazie projektu, wykorzystuje się do tego celu symulację sieci cyfrowej zamodelowanej z wykorzystaniem algebry Boole'a. W artykule omówiono metody symulacji systemów cyfrowych. Na podstawie przedstawionych algorytmów został opracowany program pozwalający na wykrywanie hazardów w układach cyfrowych.

[1] Politechnika Radomska, Wydział Transportu i Elektrotechniki; 26-600 Radom; ul. Malczewskiego 29. *Tel: + 48 48 361-77-28, Fax: + 48 48 361-77-42, E-mail: r.pniewski@pr.radom.pl*
[2] Politechnika Radomska, Wydział Transportu i Elektrotechniki; 26-600 Radom; ul. Malczewskiego 29. Tel: + 48 48 361-77-28, Fax: + 48 48 361-77-42, E-mail: m.kornaszewski @pr.radom.pl

2. METODY OPISU UKŁADÓW CYFROWYCH

Dowolny system cyfrowy można opisać na wiele różnych sposobów. Najczęściej stosuje się metody opisu funkcji logicznej w postaci:

- Tablica prawdy (truth table)
- Mapa Karnaugh
- Formy normalne
- Formy wielomianowe
- Drzewa i diagramy decyzyjne
- Reprezentacja sześcienna

Tablica prawdy po lewej stronie zawiera wszystkie zmienne, a po prawej rozwiązanie funkcji logicznej.

Mapa Karnaugh ma szerokie zastosowanie w minimalizacji funkcji. Operacje na niej dzięki prostym zasadom stają się szybkie a efekty minimalizacji optymalne. Mapa Karnaugh sprawdza się jedynie dla niedużej liczby zmiennych, ponieważ potrzebujemy 2 do potęgi n kratek. Np. przy 6 i więcej zmiennych tabela staje się zbyt duża (1024 kratki).

Formy normalne to formy zapisywane za pomocą umownych oznaczeń np.:
f = x1 \/ ~(x2)*(x3), gdzie ~ to NOT * to AND.

Formy wielomianowe można podzielić na:
- Wielomiany Reed-Muller - to wielomiany przedstawione tylko za pomocą operacji AND XOR i NOT. Przy pomocy tych trzech funkcji da się przedstawić każdą funkcję logiczną.
- Wielomiany arytmetyczne - to normalne wielomiany (operacje arytmetyczne + i -), które zwracją wynik 0 lub 1 w przypadku, gdy zmienne są 0 lub 1 np. f = x1 + x2 - 2*x1*x2 to to samo co f = x1 XOR x2 (sprawdź na tabeli prawdy).

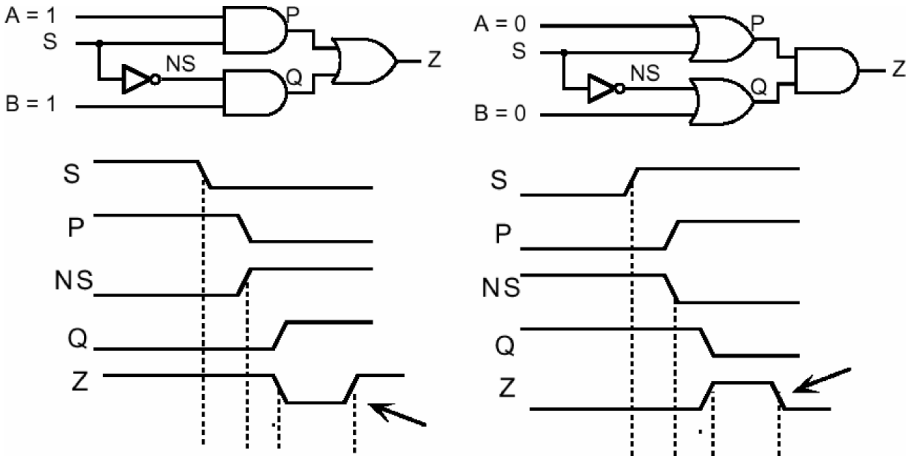
Każdej funkcji logicznej odpowiada wiele wielomianów arytmetycznych i tak samo jednemu wielomianowi arytmetycznemu może odpowiadać wiele funkcji logicznych. Wielomian arytmetyczny uznaje się za lepszy wtedy gdy jego współczynniki są mniejsze oraz gdy jego produkty (składniki tj. np. x1*x2*x3 - produkt 3 zmiennych) są mniejsze.

3. ZJAWISKA SZKODLIWE W UKŁADACH CYFROWYCH

W układach cyfrowych mogą wystąpić dwa rodzaje zjawisk szkodliwych, powodujące błędne działanie układu:

a) hazardy,
b) wyścigi (gonitwy) krytyczne

Zjawisko hazardu, objawia się pojawianiem przypadkowych impulsów na wyjściu układu (w stanach przejściowych). Źródłem niepożądanego stanu na wyjściu układu są nieidealne właściwości przełączające. Wyróżniamy hazardy statyczne w zerze i w jedynce (rys.1).



Rys.1. Hazard statyczny w „1” i w „0”

Wykrycie wyścigów krytycznych, wymaga znajomości wymaganej funkcji realizowanej przez układ cyfrowy, i może być dokonane przy wykorzystaniu dowolnego z algorytmów (występuje inna sekwencja stanów niż założona w projekcie).

4. SYMULACJA UKŁADÓW

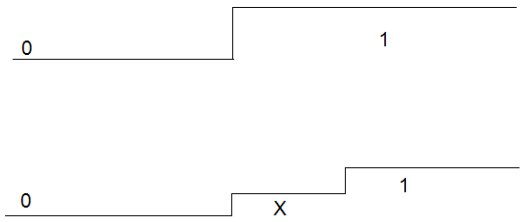
4.1 Algorytmy symulacji układów cyfrowych

Przy weryfikacji systemów cyfrowych, do sprawdzenia prawidłowości działania układu wykorzystuje się symulację komputerową. Algorytm symulacji (zrealizowany w postaci programu) opisuje zachowanie sieci logicznej. Algorytmy symulacji można sklasyfikować następująco:

synchroniczna lub asynchroniczna.
trójstanowa lub dwustanowa,
kompilowana lub interpretacyjna (sterowana tablicowo),

Przy symulacji synchronicznej każda operacja zmiany stanu i próbkowania sygnałów z określonych węzłów sieci logicznej jest sterowana przez „wewnętrzny” zegar symulatora. Wynika z tego, że w symulacji synchronicznej modelowana jest sieć „idealna” bez opóźnień. Przy symulacji asynchronicznej wartości sygnałów we wszystkich węzłach są sprawdzane i uaktualniane natychmiast po każdej zmianie stanu. Metoda ta wymaga wielokrotnego sprawdzenia wszystkich sygnałów w sieci logicznej, aż do osiągnięcia stanu ustalonego we wszystkich punktach układu (po każdorazowej zmianie sygnału wejściowego). Pomimo złożoności obliczeniowej i czasochłonności metoda ta jest powszechnie wykorzystywana w programach CAE ponieważ, powyżej przeciwieństwie do metody synchronicznej, poprzez uwzględnienie czasów propagacji sygnałów przez bramki pozwala na wykrywanie hazardów (nie we wszystkich przypadkach). Oferowane na rynku symulatory (zawarte w systemach CAE) pozwalają na wykrycie hazardu w ściśle określonych warunkach symulacji (czasy propagacji sygnałów przez bramki cyfrowe), nie gwarantują one jednak wykrycia tego zjawiska po zmianie czasów propagacji (np. po zmianie technologii).

Jedną z metod pozwalających na wykrycie hazardów w układach cyfrowych jest symulacja wykorzystująca logikę trójwartościową (metoda ta choć wykorzystuje symulację synchroniczną pozwala na wykrywanie zjawisk szkodliwych). Symulacja trójwartościowa jest realizowana przez dwukrotny przebieg przez dane. Przy każdej zmianie sygnału wejściowego, przyjmuje się, że sygnał zmieniając się z 0 na 1 lub z 1 na 0 przechodzi przez stan pośredni X (rys2). Do symulacji wykorzystuje się zmodyfikowane tabele przejść bramek logicznych. W tabeli Tab.1. przedstawiono przykładowe funkcje przejść (trójstanowe) dla dwuwejściowych bramek NAND I NOR.



Rys.2. Przebieg sygnału w symulacji dwuwartościowej i trójwartościowej

Tab.1. Trójwartościowa tabela przejść bramek NAND I NOR

WEJŚCIA	NAND	NOR
00	1	0
0X	1	X
01	1	1
X0	1	X
XX	X	X
X1	X	1
10	1	1
1X	X	1
11	0	1

Ostatni z podziałów, na metody kompilowane i interpretacyjne odnosi się do technicznej strony realizacji procesu symulacji. W metodzie interpretacyjnej kod programu symulującego sieć logiczną nie zmienia się po zmianie konfiguracji układu. Dane o układzie przedstawione są w postaci powiązanych struktur, które są przeglądane i analizowane przez program symulacyjny. W metodach kompilacyjnych struktura powiązań w sieci logicznej jest zamieniana na równoważny opis w określonym języku programowania (np. C). Kod uzyskany po skompilowaniu pozwala na znaczne przyspieszenie obliczeń związanych z symulacją systemu. Metoda ta jest szczególnie korzystna przy złożonych sieciach logicznych, ponieważ czas zużyty na kompilację projektu może być wielokrotnie mniejszy od zysku czasowego w procesie symulacji (skompilowany program wykonuje się kilkadziesiąt razy szybciej od interpretowanego). Większość programów symulacyjnych wykorzystuje model interpretowany, sterowany tablicowo (tak przynajmniej wynika z opisów i prezentowanych ograniczeń w symulacji

działania układów). Jednym z programów (znanych autorom), który wykorzystuje metodę kompilacyjną jest QUCS (Quite Universal Circuit Symulator). QUCS to darmowy program (dla systemu Linux - powstała również wersja dla Windows) przeznaczony do symulacji układów elektronicznych. Do symulacji układów analogowych wykorzystano biblioteki SPICE, natomiast symulacja układów cyfrowych odbywa się przez konwersję opisu układu do języka „C" i wykorzystanie zewnętrznego kompilatora.

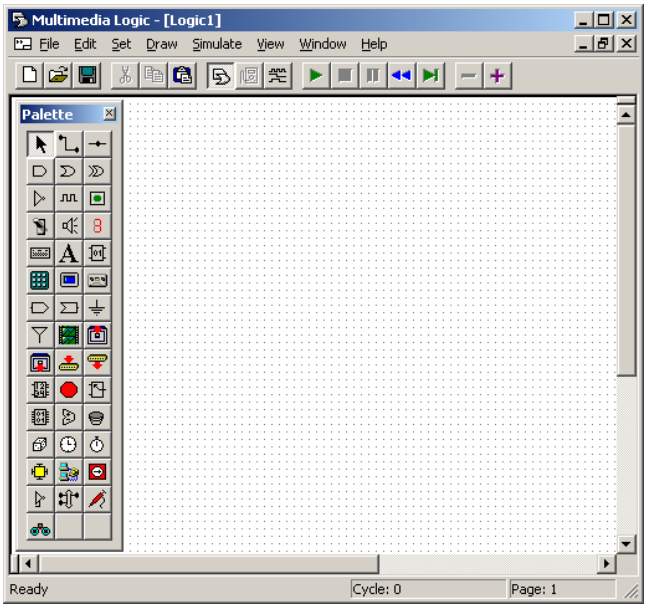
4.1 Oprogramowanie do symulacji układów

Przy opracowaniu oprogramowania do symulacji, autorzy wykorzystali kody źródłowe programu **MultimediaLogic.** MultimediaLogic to bezpłatny program do projektowania i symulacji działania układów cyfrowych. Pod adresem http://www.softronix.com/ znajdują się pliki instalacyjne oraz kod źródłowy programu. Kod źródłowy programu został zmodyfikowany dla potrzeb symulacji trójwartościowej. Poniżej pokazano przykład procedur przeznaczonych do wyznaczania odpowiedzi 2-wejściowych bramek NAND i NOR odpowiadające tabeli 1 (wartości logiczne uwzględniane w systemie:0,1 i -1 reprezentujące stan przejściowy):

```
//funkcja NAND
char nand(char we1,char we2)
{
if(we1==0)
  return(1);
if(we2==0)
  return(1);
if((we2==1) & (we1==1)
  return(0);
return(-1);
}

//funkcja NOR
char nor(char we1,char we2)
{
if(we1==1)
  return(0);
if(we2==1)
  return(0);
if((we2==0) & (we1==0)
  return(1);
return(-1);
}
```

Oprogramowanie firmy Sofrtronix zostało napisane z wykorzystaniem pakietu VisualC. W programie wykorzystano algorytm symulacji synchronicznej. Dostosowanie do potrzeb symulacji trójwartościowej polegało na zmianie modeli układów cyfrowych i dwukrotnym wywołaniu procesu symulacji. Główne okno aplikacji MultimediaLogic zostało przedstawione na rysunku 3.



Rys. 3. Okno główne programu MultimediaLogic.

5. WNIOSKI

Przedstawione w artykule metody symulacji układów cyfrowych pozwalają na wykrycie błędów już w trakcie opracowania projektu. Wyeliminowanie błędów w tej fazie projektu znacznie zmniejsza koszty wdrożenia systemu. Szczególnie symulacja behawioralna wstępnego projektu (na etapie opracowania specyfikacji) pozwala na znaczne ograniczenie kosztów i przyspiesza proces projektowania.

6. BIBLIOGRAFIA

[1] Kalisz J.: *Podstawy elektroniki cyfrowej*, WKiŁ 2008.
[2] Łuba T., Zbierzchowski B.: *Komputerowe projektowanie układów cyfrowych*, WKiŁ 2005.
[3] Majewski W., Albicki A.: *Algebraiczna teoria automatów*, WNT 1980
[4] Pniewski R., Strzyżakowski Z.: *Wykrywanie zjawisk szkodliwych w systemach cyfrowych*. Materiały konferencyjne PTSK Kazimierz Dolny 2007
[5] http://www.softronix.com/ - strona programu MultimediaLogic