Spis treści:

1. Wstęp

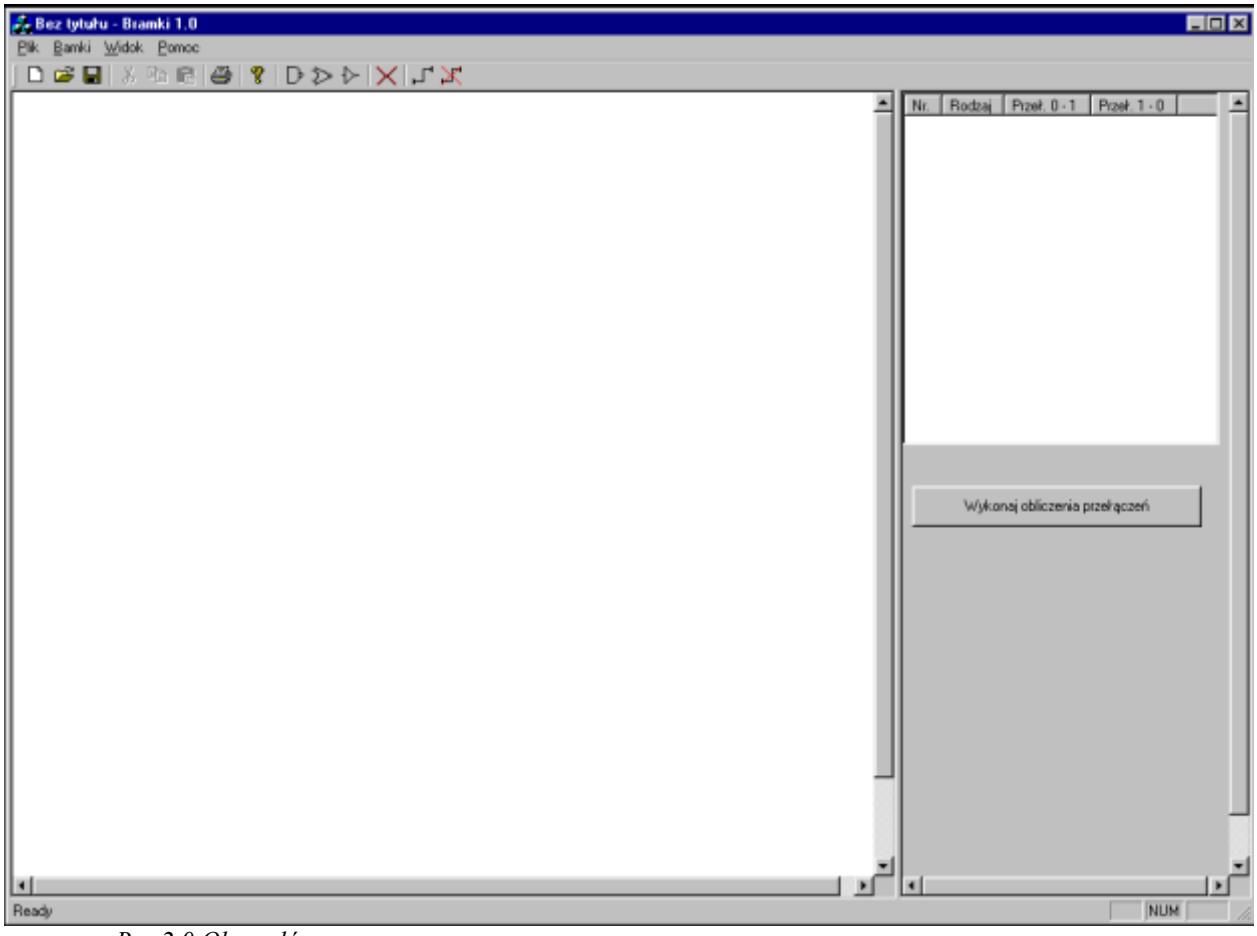
Program „Bramki 1.0” powstał w ramach projektu z przedmiotu „Układy VLSI - laboratorium”. Program składa się z prostego edytora schematów logicznych, dopuszczający trzy podstawowe rodzaje bramek NAND, NOR i NOT. Program potrafi tworzyć schemat na podstawie podanego równania, importować dane w formacie „Netlist” z programów Protel ™ oraz OrCAD ™. Program umożliwia wyliczenia ilości przełączeń stanów poszczególnych bramek na podstawie schematu.

Program został stworzony w technice obiektowej przy użyciu „Microsoft Visual Studio” i pracuje w środowisku Windows 98, oraz Windows NT 4.0 (z Internet Explorer w wersji 4.0 lub wyższej).

2. Opis edytora schematów

Edytor schematów jest wizualnym środowiskiem, które umożliwia tworzenie prostych schematów logicznych składających się z trzech podstawowych bramek NAND, NOR i NOT.

Po uruchomieniu programu użytkownik widzi okno edycyjne służące do tworzenia schematu:



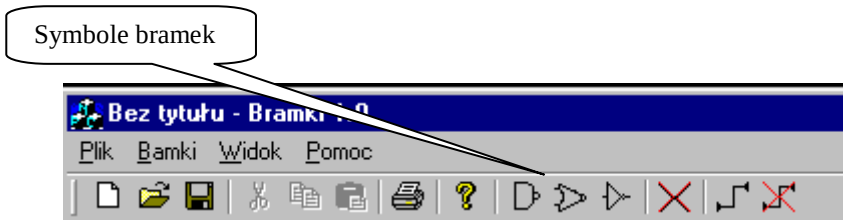
Rys 2.0 Okno główne programu

Okno podzielone zostało na dwie części.

Lewa strona służy do graficznej edycji schematu, prawa wyświetla informacje o istniejących bramkach w systemie.

2.1 Dodawanie bramek do schematu:

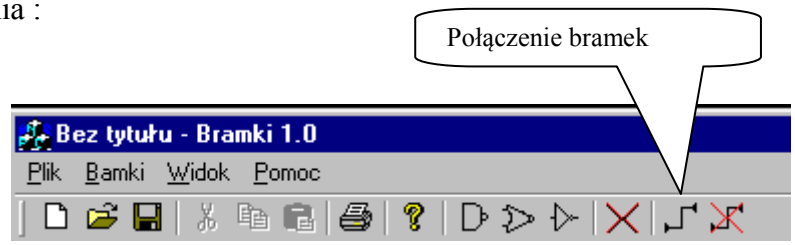
Aby dodać bramkę do schematu należy z paska narzędzi lub menu *Bramki* wybrać odpowiedni rodzaj. Kursor zmieni kształt symbolizujący rodzaj bramki, należy umieścić symbol bramki na wybranym miejscu schematu i wcisnąć lewy klawisz myszy. Aby zaniechać umieszczania bramki należy wcisnąć klawisz *Escape.*



Rys 2.1 Menu i pasek narzędzi programu

2.2 Łączenie bramek

Łączenie bramek odbywa się poprzez wskazanie bramek między którymi będzie tworzone połączenie. W tym celu z paska narzędzi należy wybrać symbol dodawania połączenia :

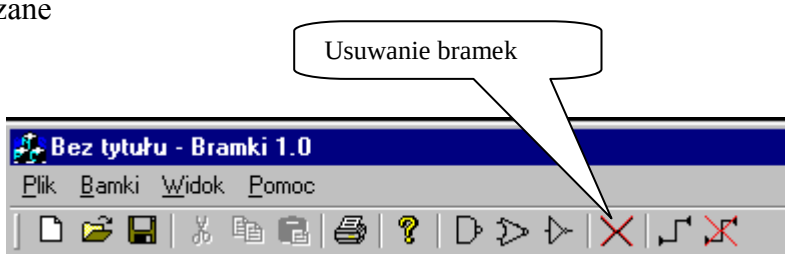


Rys 2.2 Dodawanie połączeń

Następnie wskazać bramkę, której wyjście będzie wejściem do następnej bramki, po czym wskazać bramkę do której będziemy dodawać wejście.

2.3 Usuwanie bramek

Usuwanie bramek odbywa się przez wybranie z paska narzędzi symbolu usunięcia bramki i wskazania wybranej bramki. Razem z bramką usuwane są wszystkie połączenia z nią związane

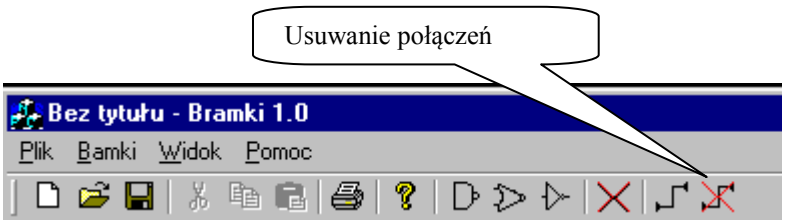


Rys .2.3 Usuwanie bramek

2.4 Usuwanie pojedynczych połączeń

Usuwanie połączeń odbywa się w bardzo podobny sposób jak dodawanie połączeń. Należy wybrać z paska narzędzi symbol usuwania połączeń. Następnie wskazać pierwszą bramkę od której jest połączenie , później bramkę następną.

W przypadku gdy bramki nie są połączone, lub są połączone w drugą stronę zostanie wygenerowany odpowiedni komunikat i usuwanie zostanie zaniechane.



Rys .2.4 Usuwanie połączeń

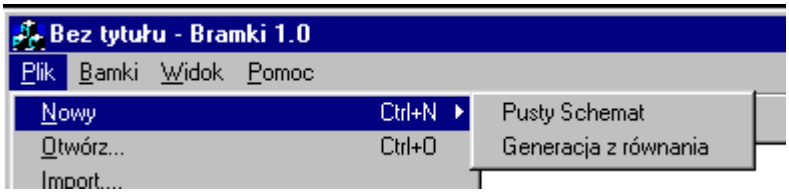
2.5 Dodawanie wejść do schematu

Wejścia są dodawane do schematu poprzez wybranie z menu *Bramki* polecenia *Wstaw-> Wejście do systemu* i umieszczenia symbolu na schemacie. Wejście usuwana jest w ten sam sposób jak usuwane są bramki.

Rys 2.5 Menu „Wejście do systemu”

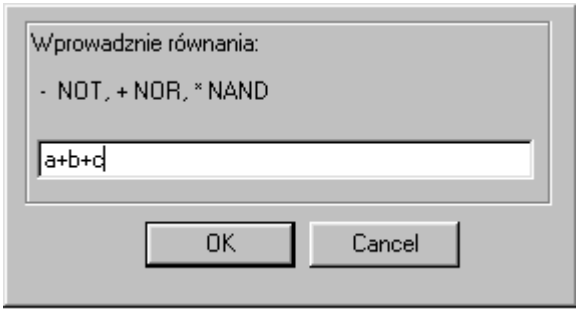
3. Generowanie schematu z równania

Zamiast edytować schemat ręcznie mamy możliwość generacji schematu za pomocą równania. Równanie wprowadzane jest w oknie dialogowym uruchamianym poleceniem z menu *Plik- >Nowy->Generacja z równania*



Rys 3.1 Menu „Generacja z równania”

Po wybraniu polecenia uruchamiane jest okno dialogowe z polem do edycji równań. W edycji zostały przyjęte następujące symbole operacji: - NOT, + NOR, * NAND



Rys 3.2 Edycja równania

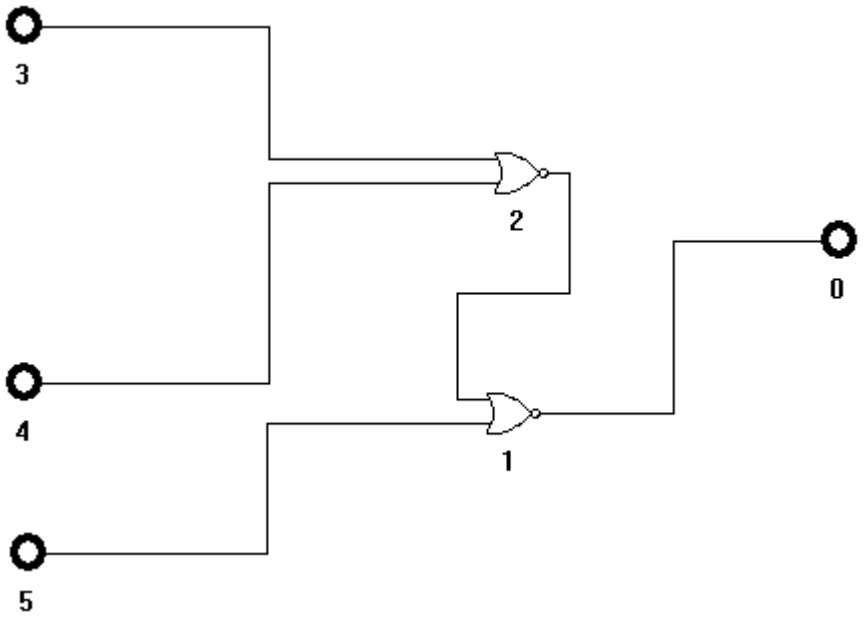
3.1 Rozbiór równania

Równanie podawane jest w postaci infiksowej, w związku z tym, algorytm rozbioru równania generuje drzewo, którego korzeniem jest bramka wyjściowa.

Następnie równanie przetwarzane jest od lewej strony, szukając operatorów o najniższym priorytecie, przy znalezieniu takiego operatora tworzona jest nowa gałąź, która jest ponownie rozbierana na podstawowe części. Po całkowitym rozbiorze równania obliczane są najniższe podwyrażenia a ich wyniki służą do obliczania podwyrażeń znajdujących się wyżej.

Po znalezieniu rodzaju operacji dodawana jest stosowna bramka, a po przeparsowaniu dalej do bramek dodawane są odpowiednie połączenia.

Na końcu do bramek przyłączane są wejścia systemu.



Rys 3.2.1 Schemat uzyskany z równania a+b+c

4.0 Import danych w formacie Netlist

Import danych pozwala na wgranie do programu prostych net list w formatach Protel'a oraz w formacie OrCad/PCB II. Rozpoznanie formatu odbywa się automatycznie na podstawie danych w pliku. Ponieważ program operuje jedynie na bramkach NAND, NOR i NOT w net liście mogą się pojawić tylko takie nazwy bramek. W momencie gdy zostanie stwierdzona inna nazwa proces ładowania jest przerywany z błędem. Kolejnym założeniem jest unikalność nazw bramek. W przypadku gdy pojawiają się dwie bramki o tej samej nazwie działanie programu może być niepoprawne.

Akceptowany format Protela:

```
[
nazwa_bramki

symbol bramki tylko: NAND, NOR i NOT


]   ta sekcja może być powtórzona wielokrotnie ale nazwa musi być unikalna
(
nazwa połączenia
nazwa_bramki-numer_końcówki
...
)
```

Przy czym jeśli bramka jest typu NAND lub NOR to jeśli numer końcówki jest równy trzy to jest on uznawany za wyjście, natomiast dla NOT wyjściem jest końcówka o numerze 2. Pozostałe końcówki są uznawane za wejścia. W jednej sekcji połączenia może być tylko jedno wejście.

Akceptowalny format OrCad'a:

```
( {OrCAD PCB II Netlist Format}
 ( 00000001  nazwa_bramki symbol_bramki_NOT
  ( 1 Netnazwa_poprzedzającej_bramki_1 )
  ( 2 Netnazwa_bramki_2 )
 )
  ( 00000002 nazwa_bramki symbol_bramki_NAND_NOR
  ( 1 Netnazwa_poprzedzającej_bramki_1 )
  ( 2 Netnazwa_poprzedzającej_bramki_2 )
  ( 3 Netnazwa_bramki_3 )
 )
...
)
```

Bardzo ważne jest by pierwsza linia wyglądała dokładnie jak na specyfikacji, ponieważ na tej podstawie wykrywany jest format OrCad'a. Ważne jest również zachowanie właściwych odstępów tzn. nazwa bramki musi się zaczynać na 14 pozycji. Zakończenie listy dla bramki musi mieć format „spacja)". W każdej z pozycji połączenia nazwa połączenia musi zaczynać się od liter „Net" po czym następuje nazwa bramki (do „_").

5.0 Wykonywanie przeliczeń ilości przełączeń

W celu obliczeń ilości przełączeń bramek, obiekt Cbramka został rozszerzony o następujące atrybuty:

- int IloscPrzelaczen_0_1;
  zapamiętywana w nim jest ilość zmian wyjścia bramki z stanu ‘0’ do ‘1’.
- int IloscPrzelaczen_1_0;
  zapamiętywana w nim jest ilość zmian wyjścia bramki z stanu ‘1’ do ‘0’.
- long Iteracja;
  zmienna w której zapisany jest numer sygnału wejściowego, który dotarł do bramki i dla którego dana bramka ustaliła już swoje wyjście.
- int WartoscWyjscia;
  wartość wyjścia konkretnej bramki.

Na początku obliczeń wszystkie bramki są zerowane przy pomocy metody

```
void CBramka::ZerujBramke()
```

Ustawia ona ilości przełączeń na 0, wartość na wyjścia ustalana jest na stan niski (logiczne zero). Numer aktualnej iteracji ustawiany jest na –1, co ma oznaczać że nie rozpoczęło się obliczanie ilości przełączeń.

Następnie w funkcji

```
void CSchemat::ObliczPrzelaczenia()
```

dokonywane jest obliczenie kolejnych stanów bramek w wprowadzonych schemacie. Wykonywane jest to według następującego algorytmu:

1. Obliczana jest liczba wejść w schemacie oraz zapamiętywane są ich numery.
2. Zerowane są wszystkie bramki należące do schematu.
3. W przypadku gdy nie zostało wprowadzone żadne wejście algorytm kończy działanie.
4. Obliczany jest zakres sygnałów, które będą podawane na wejścia (od 0 do 1 << iloscWejsc)
5. W pętli symulowane jest podawanie kolejnych sygnałów na wejścia schematu (ustawiany jest atrybut iteracja na wartość bieżącego sygnału, która zarazem jest numerem iteracji – oznacza ona że na wejściach został ustalony już sygnał):

```
long maska = 0x0001;
for(i=0; i < iIloscWejsc; i++)
{
     int iPozycjaWejscia = caPozycjeWejsc.GetAt(i);
     CBramka bramka =
cbBramkiSchematu.GetAt(iPozycjaWejscia);
     if ( (maska & czas) != 0 )bramka.WartoscWyjscia = 1;
     else                      bramka.WartoscWyjscia = 0;
     bramka.Iteracja = czas;
     maska = maska << 1;
}
```

6. Następnie ustalamy wartości kolejnych bramek: dla każdej bramki, która nie ma jeszcze ustalonego sygnału na wyjściu dla bieżącego sygnału (warunek: `b.Iteracja < czas`) oraz nie jest to bramka typu wejście lub wyjście układu sprawdzamy czy posiada ona ustalone wszystkie wejścia (czy bramki, których wyjścia są połączone z wejściami aktualnie analizowanej bramki zostały już obliczone. W przypadku gdy możemy dokonać obliczenia stanu danej bramki wywoływana jest funkcja dokonująca obliczeń oraz

ustawiająca atrybut `Iteracja` na bieżącą wartość zmiennej `czas`. Ten sam algorytm wykonujemy dla każdej z bramek, dopóki cały schemat nie zostanie obliczony.

7. Następnie zmieniany jest sygnał wprowadzany na wejście schematu i powtarzany jest punkt 5 i 6 algorytmu. Wykonywanej jest to dopóki na wejściach układu nie pojawią się wszystkie możliwe kombinacje stanów.