ИНФОРМАТИКА

2004 **июль-сентябрь** **№3**

УДК 519.7

П.Н. Бибило, Д.А. Кочанов

ПОСТРОЕНИЕ VHDL-МОДЕЛЕЙ СХЕМ ДЛЯ ВЫЧИСЛЕНИЯ ФУНКЦИЙ, ЗАДАННЫХ В ТАБЛИЧНОЙ ФОРМЕ

Предлагаются средства для создания VHDL-моделей аналитических функций с использованием стандартных математических пакетов. Аналитические функции реализуются в виде макроэлементов цифровых сверхбольших интегральных схем.

Введение

Возможности современной микроэлектронной технологии позволяют реализовать на одном кристалле законченные вычислительные и управляющие системы. При проектировании цифровых заказных СБИС (сверхбольших интегральных схем) возникает проблема автоматизации проектирования макроблоков (макроэлементов), предназначенных для вычисления разнообразных математических функций [1, 2].

Важнейшими аспектами решения проблемы автоматизации проектирования логических схем, реализующих математические функции в цифровых СБИС, являются:

– точность приближения функций;

– способ приближения (вычисления) функций;

– язык описания объекта проектирования (макроблока) в САПР;

– формальная модель описания макроблока на выбранном языке описания.

В цифровых СБИС любые функции должны быть реализованы логическими схемами в том или ином базисе логических элементов. На характеристики (сложность, быстродействие) схемы влияет как требуемая точность вычисления функции, так и способ приближения. Для вычисления элементарных функций в специализированных СБИС предложены различные методы, например метод «цифра за цифрой» [1].

В данной работе решается проблема создания VHDL-моделей функционирования макроэлементов, предназначенных для вычисления аналитических функций. Для исходного описания функций используется их аналитическое задание в математических пакетах (системах), для результирующего – табличная форма представления, от которой осуществляется переход к VHDL-модели. Прежде чем дать обоснование принятого подхода, приведем краткие сведения о программных математических системах и языке VHDL.

1. Математические системы Mathematica, Maple, MathCad

Системы Mathematica, Maple, MathCad являются наиболее известными в мире математическими пакетами [3-5]. Они представляют широкие возможности для символьных и численных вычислений, позволяют решать разнообразные математические задачи с необходимой точностью, имеют развитую двухмерную и трехмерную графику, а также встроенные языки программирования высокого уровня. Эти возможности сделали их незаменимыми средствами для исследователей, так как математические пакеты позволяют пользователю сосредоточиться на постановке задачи (математической формулировке проблемы) и на анализе полученных результатов.

2. Язык VHDL

Язык VHDL (Very high speed integrated circuits Hardware Description Language) [6] является одним из самых известных в мире и используемых на практике языком описания проектов цифровых систем, реализуемых на СБИС различных классов – специализированных (заказных), полузаказных и программируемых пользователями интегральных схем (ПЛИС). Все современ-

ные САПР используют язык VHDL в качестве входного. Проблемы моделирования и синтеза логических схем по спецификациям на языке VHDL подробно изложены в книге [7].

3. Создание VHDL-модели аналитической функции

Табличный способ задания функций является универсальным. При кодировании численных значений булевыми (двоичными 0 1) кодами табличное представление функции заменяется таблицей истинности системы булевых функций. Выбор табличного представления функции ведет, по существу, к тому, что любая исходная математическая функция заменяется при СБИС-реализации многовыходной логической схемой. Известно, что теория логического синтеза схем наиболее развита именно для комбинационных логических схем. Однако табличное представление функций в силу своей универсальности имеет как достоинства, так и недостатки. Важнейшим недостатком является большой размер таблиц, если требуется высокая точность вычислений. Поэтому данный подход может оправдать себя при «быстром» проектировании, когда требуется создать работающую систему или ее макет в кратчайшие сроки и когда приемлемой является размерность решаемой задачи для выбранного микроэлектронного базиса, а также для моделирования работы макроблока в составе некоторой системы.

В пользу выбора какого-либо математического пакета в качестве средства начального проектирования говорят возможности быстрого, эффективного и, что еще более важно, корректного решения задачи вычисления функций с требуемой точностью. Если пользователь работает в математическом пакете и выбрал табличный способ представления результатов вычислений, то его в этом случае не беспокоит ни способ приближения функции, ни требуемая точность приближения – эти проблемы решаются с помощью математического пакета. Обосновывать выбор языка VHDL, по существу, нет необходимости, так как VHDL является мировым стандартом.

Традиционный подход для построения комбинационной схемы, реализующей требуемые функции, следующий [8]. Проектировщик пишет программу на каком-либо универсальном языке программирования (С, Паскаль и т. д.) для вычисления данной функции и представляет результат в табличной форме, где десятичному значению аргумента соответствует десятичное значение функции. Затем переходит к двоичным кодам и получает таблицу истинности системы булевых функций. Данная система может быть реализована на постянном запоминающем устройстве (ПЗУ) в составе заказной СБИС либо в виде нерегулярной логической схемы в библиотеке проектирования заказной СБИС, макроблок – в составе ПЛИС либо в составе полузаказной СБИС на основе базовых матричных кристаллов (БМК). Чтобы использовать САПР, нужно представлять таблицу истинности либо эквивалентное логическое описание на входном языке используемой САПР.

Предлагаемый подход состоит в использовании специально написанных программ для преобразования заданной таблицы истинности в десятичной системе счисления в формат представления логической функции на языке VHDL (рис. 1). При этом таблица истинности может быть подготовлена не только при помощи различных математических пакетов, но и вручную.

Вначале при помощи какого-либо математического пакета (или вручную) подготавливается табличное задание требуемой функции в десятичной системе счисления. Затем на основе этой таблицы программа-генератор *dtob.exe* создает ее представление в двоичной системе счисления с определяемым разработчиком числом двоичных разрядов после запятой, т. е. строит таблицу истинности. После этого вторая программа-генератор *tbl_txt.exe* на основе построенной таблицы истинности создает алгоритмическое описание требуемой функции на языке VHDL.

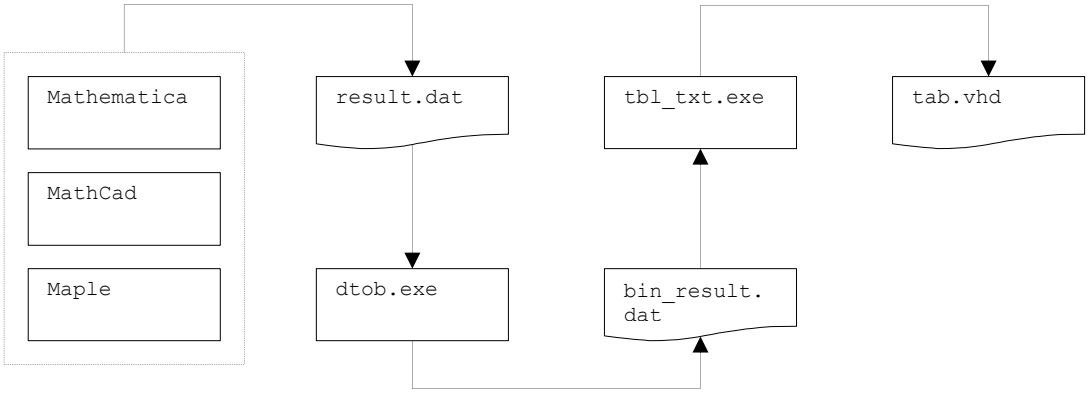

Рис. 1. Процесс построения VHDL-модели

4. Примеры создания VHDL-моделей аналитических функций

П р и м е р 1. Пусть требуется реализовать в составе СБИС макроблок, вычисляющий абсолютное значение функции $y = \sin x$, при этом x принимает значения на численном интервале $[0, \frac{\pi}{2}]$. Пусть требуемая точность вычислений значений аргумента и значений функции ограничена n двоичными разрядами.

В листингах 1, 2 и 3 приводятся алгоритмы вычисления функции *y=sin(x)* в системах Mathematica, Maple и MathCad соответственно. Каждая команда задается в отдельной строке.

Листинг 1. **Программа вычисления функции y = sin(x) в системе Mathematica**

```
fun[x_]:= Sin[x]
a:= 0
b:= Pi/2
n:= 3
Export["result.dat", ({#, fun[#]}&/@Range[N[a], N[b], N[2^(-n)]])]
```

Листинг 2. **Программа вычисления функции y = sin(x) в системе Maple**

```
fun:=sin
a:=0;
b:=Pi/2;
n:=3;
lst:=Matrix(round(evalf((b-a)/(2^(-n)))),2);
for i from a by 1 to trunc(evalf((b-a)/(2^(-n)))) do
lst[i+1,1]:=a+i*evalf(a+(2^(-n)));
lst[i+1,2]:=eval(fun(lst[i+1,1]));
end do;
ExportMatrix("result.dat", lst);
```

Листинг 3. **Программа вычисления функции y = sin(x) в системе MathCad**

$fun := \sin$

$a := 0$

$b := \frac{\pi}{2}$

$n := 3$

$j := 0..1 \quad i := 0..trunc[\frac{b-a}{2^{-n}}]$

$x_{i,0} := a + i \cdot 2^{-3} \quad x_{i,1} := \sin(x_{i,0})$

$WRITEPRN("result.dat") := x$

В каждом из листингов 1 – 3 четыре первые команды задают вычисляемую функцию *sin(x)*, переменные *a*, *b* (которые определяют начальную и конечную точки изменения аргумента), а также переменную *n,* которая задает точность (число двоичных разрядов после запятой) представления аргумента. Последующие команды рассчитывают табличное представление функции в десятичной системе счисления и сохраняют результат в файле *result.dat*.

Результатом работы каждого из приведенных выше алгоритмов является табличное представление функции в виде текстового файла *result.dat*. Оно приведено в листинге 4.

Листинг 4. **Задание функции sin(x) в десятичной форме записи (файл result.dat)**

```
0.      0.
0.125   0.12467473338522769
0.25    0.24740395925452294
0.375   0.36627252908604757
0.5     0.479425538604203
0.625   0.5850972729404622
0.75    0.6816387600233341
0.875   0.7675435022360271
1.      0.8414709848078965
1.125   0.9022675940990952
1.25    0.9489846193555862
1.375   0.9808930570231557
1.5     0.9974949866040544
```

Затем при помощи команды (выполняемой в командной строке, например, операционной системы MS DOS)

```
dtob.exe result.dat bin_result.dat x:n y:m
```

«отсекается» требуемое число разрядов: оставляется *n* разрядов (после точки) в задании аргументов функции и *m* разрядов в задании значений функции. Двоичное представление сохраняется в файле *bin_result.dat*. В листинге 5 приведен файл *bin_result.dat* для n=3 и m=4.

Листинг 5. **Задание функции sin(x) в двоичном коде (файл bin_result.dat)**

```
.000  .0000
.001  .0001
.010  .0011
.011  .0101
.100  .0111
.101  .1001
.110  .1010
.111  .1100
1.000 .1101
1.001 .1110
1.010 .1111
1.011 .1111
1.100 .1111
```

После отсечения происходит некоторая потеря точности значений функции. Точность зависит от параметра m, который задается разработчиком. На рис. 2 изображен график функции *sin(x)* и множество точек значений функции, построенных по двоичному представлению (см. листинг 5).

Чтобы проинтерпретировать листинг 5 как таблицу истинности системы булевых функций, требуется добавить знаковые разряды для представления значений аргумента и функции. Кроме того, требуется определенное число разрядов для представления целой части чисел.

Табличное задание функции в двоичном коде интерпретируется как таблица истинности системы m булевых функций. В таблице истинности (табл. 1) приведена система четырех ($m = 4$) булевых функций *y1*, *y2*, *y3*, *y4* вместе со знаковым разрядом *y0*. В знаковом разряде 0 обозначает положительное значение функции, 1 – отрицательное. Таблица строится по листингу 5. Старшинство разрядов в двоичном представлении является общепринятым – старшим считается первый разряд слева. Это справедливо как для представления значений аргумента, так и для представления значений функции.

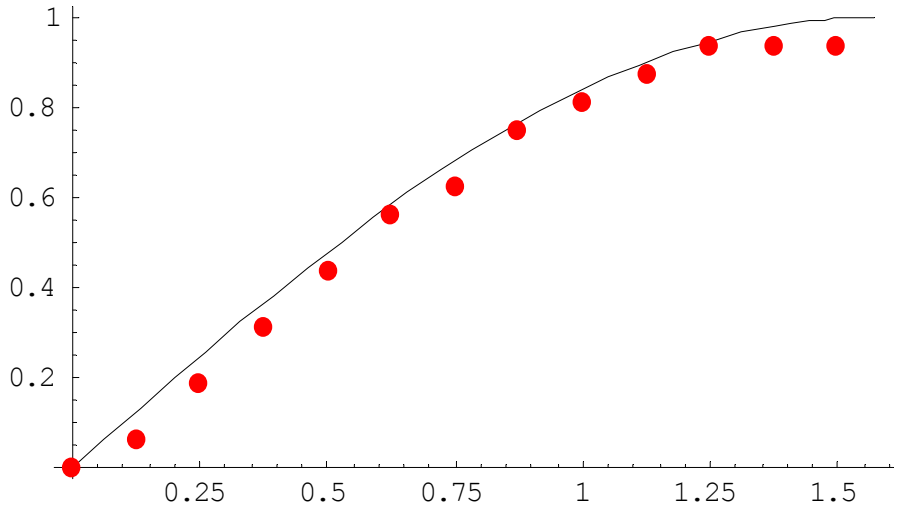

Рис. 2. График функции *sin(x)* и множество точек,
по которым производится приближение функции

Таблица 1

Табличное задание функции *sin(x)*
в двоичном коде вместе со знаковым разрядом

x0	x1	x2	x3	x4	y0	y1	y2	y3	y4
0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	1
0	0	0	1	0	0	0	0	1	1
0	0	0	1	1	0	0	1	0	1
0	0	1	0	0	0	0	1	1	1
0	0	1	0	1	0	1	0	0	1
0	0	1	1	0	0	1	0	1	0
0	0	1	1	1	0	1	1	0	0
0	1	0	0	0	0	1	1	0	1
0	1	0	0	1	0	1	1	1	0
0	1	0	1	0	0	1	1	1	1
0	1	0	1	1	0	1	1	1	1
0	1	1	0	0	0	1	1	1	1

В рассматриваемом примере все значения аргумента и все значения функции положительные, поэтому столбцы *x0*, *y0* являются нулевыми. Чтобы на основе двоичной таблицы истинности получить VHDL-описание схемы, необходимо воспользоваться программой *tbl_txt.exe*, произведя следующий ее вызов из командной строки:

```
tbl_txt.exe bin_result.dat tab.vhd
```

Результатом работы этой команды будет файл *tab.vhd*, содержащий VHDL-описание логической схемы, которая реализует требуемую функцию. В частности, для табл. 1 файл *tab.vhd* приведен в листинге 6. При построении VHDL-описания файл *bin_result.dat* интерпретируется как таблица истинности.

Листинг 6. **Представление таблицы истинности на языке VHDL (tab.vhd)**

```
Library IEEE;
use IEEE.STD_LOGIC_1164.all;
entity macro is
port(x0, x1, x2, x3, x4: in std_logic;
      y0, y1, y2, y3, y4: out std_logic);
end macro;
architecture arch of macro is
begin
y0 <= '0';

y1 <= not x0 and not x1 and     x2 and not x3 and     x4
 or   not x0 and not x1 and     x2 and     x3 and not x4
 or   not x0 and not x1 and     x2 and     x3 and     x4
 or   not x0 and     x1 and not x2 and not x3 and not x4
 or   not x0 and     x1 and not x2 and not x3 and     x4
 or   not x0 and     x1 and not x2 and     x3 and not x4
 or   not x0 and     x1 and not x2 and     x3 and     x4
 or   not x0 and     x1 and     x2 and not x3 and not x4;

y2 <= not x0 and not x1 and not x2 and     x3 and     x4
 or   not x0 and not x1 and     x2 and not x3 and not x4
 or   not x0 and not x1 and     x2 and     x3 and     x4
 or   not x0 and     x1 and not x2 and not x3 and not x4
 or   not x0 and     x1 and not x2 and not x3 and     x4
 or   not x0 and     x1 and not x2 and     x3 and not x4
 or   not x0 and     x1 and not x2 and     x3 and     x4
 or   not x0 and     x1 and     x2 and not x3 and not x4;

y3 <= not x0 and not x1 and not x2 and     x3 and not x4
 or   not x0 and not x1 and     x2 and not x3 and not x4
 or   not x0 and not x1 and     x2 and     x3 and not x4
 or   not x0 and     x1 and not x2 and not x3 and     x4
 or   not x0 and     x1 and not x2 and     x3 and not x4
 or   not x0 and     x1 and not x2 and     x3 and     x4
 or   not x0 and     x1 and     x2 and not x3 and not x4;

y4 <= not x0 and not x1 and not x2 and not x3 and     x4
 or   not x0 and not x1 and not x2 and     x3 and not x4
 or   not x0 and not x1 and not x2 and     x3 and     x4
 or   not x0 and not x1 and     x2 and not x3 and not x4
 or   not x0 and not x1 and     x2 and not x3 and     x4
 or   not x0 and     x1 and not x2 and not x3 and not x4
 or   not x0 and     x1 and not x2 and     x3 and not x4
 or   not x0 and     x1 and not x2 and     x3 and     x4
 or   not x0 and     x1 and     x2 and not x3 and not x4;
end arch;
```

Представление на языке VHDL схемы *macro* осуществлено в стиле *data flow* (потока данных). Запись VHDL-кода основывается на общеизвестном способе записи системы совершенных дизъюнктивных нормальных форм (ДНФ) булевых функций по таблице истинности, при этом используются следующие операторы логических операций языка VHDL: AND – логическое И, OR – логическое ИЛИ, NOT – логическое НЕ. Например, набор 00101 значений буле-

вых переменных интерпретируется как полная элементарная конъюнкция $\overline{x0} \& \overline{x1} \& x2 \& \overline{x3} \& x4$, входящая в совершенную ДНФ функции *y1*. Данная конъюнкция заменяется логическим выражением

```
not x0 and not x1 and x2 and not x3 and x4
```

входящим в представление функции y1 на языке VHDL (листинг 6).

В книге [7] показано, что для представления комбинационной логики могут быть использованы различные стили. Однако синтезаторы логических схем наиболее эффективно работают с VHDL-представлениями в стиле *data flow*. По данному VHDL-коду можно строить комбинационную схему. Это может быть ПЗУ, программируемая логическая матрица (ПЛМ) и другие макроэлементы заказных СБИС.

Например, реализация схемы macro на ПЛМ в системе SCAS [9] осуществляется после совместной минимизации системы функций (см. табл. 1) в классе ДНФ. Для того чтобы по VHDL-коду построить комбинационную схему в заданной библиотеке проектирования полузаказных СБИС на основе БМК либо реализовать VHDL-код на ПЛИС, могут использоваться синтезаторы: для синтеза схем в библиотеке проектирования БМК – LeonardoSpectrum [7], для синтеза ПЛИС – САПР XILINX ISE [10].

Синтезаторы логических схем по функциональным и алгоритмическим VHDL-описаниям проектируемых систем строят логические схемы. Некоторые простейшие функции записываются на языке VHDL. Это арифметические функции над целочисленными (*integer*) типами данных и суперпозиции таких функций. Однако синтезаторы логических схем по VHDL-описаниям не поддерживают вещественную арифметику и функции над вещественными числами. Возможности математических пакетов для аналитического задания функций неизмеримо выше.

Пользователь может использовать листинги 1, 2 и 3 для схемной реализации любых функций, которые допускают представление в *аналитической* форме. При этом пользователь должен изменить задание функции, точность представления и начальную и конечную точки интервала (первые четыре строки в листингах 1 – 3) и указать путь и имя файла (последняя строка), в который необходимо поместить табличное представление требуемой функции в десятичной системе исчисления. Затем необходимо в командной строке выполнить следующую последовательность команд:

```
dtob.exe result.dat bin_result.dat x:n y:m
tbl_txt.exe bin_result.dat tab.vhd
```

после чего файл *tab.vhd* будет представлять собой VHDL-описание требуемой функции.

Реализуем функцию $y = \sin x$ при n=9, m=9. Получим систему 10 булевых функций от 11 аргументов. Двоичная таблица истинности будет иметь 806 строк. Текстовый файл VHDL-модели системы логических функций имеет объем примерно 480 Кб. Данная модель моделируется в системе ModelSim [10] (рис. 3).

Реализация данной системы (без предварительной минимизации в классе ДНФ) в системе LeonardoSpectrum на FPGA типа SPARTAN2 требует 173 конфигурируемых логических ячеек CLB (Configurable Logic Block). Системой синтеза была автоматически выбрана микросхема 2s15cs144. Совместная минимизация системы булевых функций в классе ДНФ позволила получить эквивалентную систему ДНФ, заданную на 446 элементарных конъюнкциях (строках матричного представления). Минимизация осуществлялась с помощью программ кремниевого компилятора SCAS [9]. Заметим, что с помощью SCAS схема может быть реализована в виде макроэлемента заказной СБИС. Минимизация исходного представления позволила системе LeonardoSpectrum среди семейства SPARTAN2 выбрать более простую микросхему 2s30cs144, так что для реализации минимизированного представления понадобилось 166 CLB. Поэтому получаемые в результате работы алгоритма VHDL-модели лучше использовать для моделирования логических схем, синтезированных другими средствами, а не для синтеза от VHDL-модели, построенной по таблице истинности.

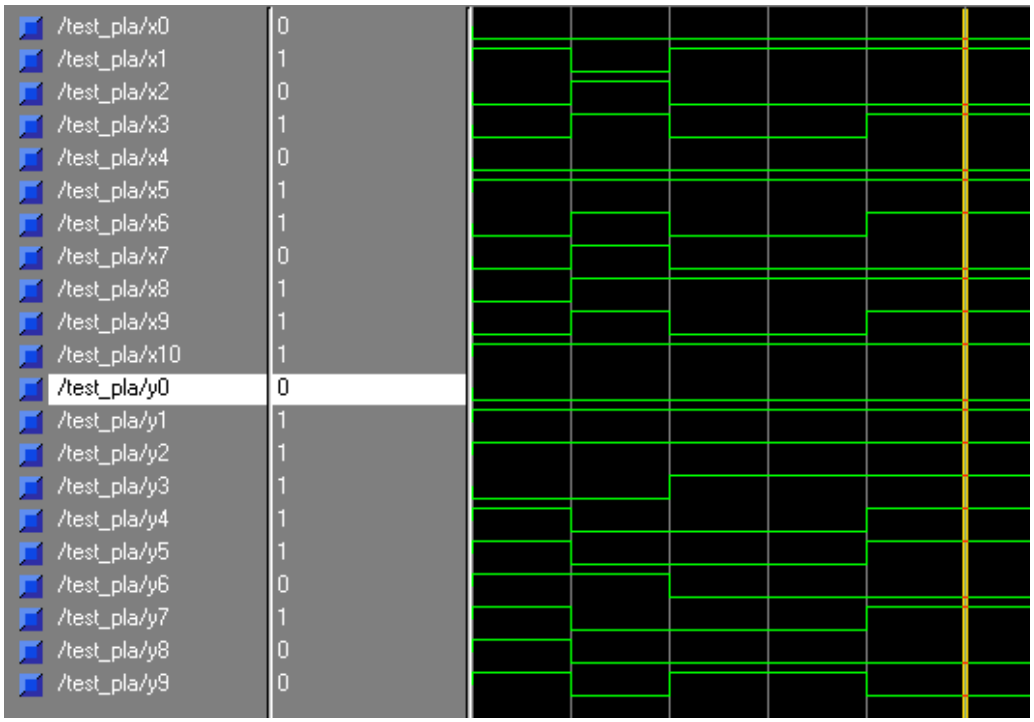

Рис. 3. Временные диаграммы моделирования функции $y = \sin x$ при n=9, m=9

П р и м е р 2. Пусть задана функция

$$f(x) = \frac{\ln(x)}{\prod_{i=0}^{4} \frac{(x - \frac{\pi * i}{8})}{\sin(x)}},$$

при этом $x > 0$, а $f(x) = 0$.

Требуется реализовать на СБИС макроблок, вычисляющий значения функции f на интервале $[0, \frac{\pi}{2}]$. Необходимая точность значений аргумента - три двоичных разряда после запятой; точность задания значений функции – пять двоичных разрядов (рис. 4). В листингах 7, 8 и табл. 2 представлены промежуточные результаты построения VHDL-модели макроблока.

Листинг 7. **Задание функции f в десятичной форме записи (файл result.dat)**

```
0.          0
0.125       -0.0018616610465720857
0.25        -0.05488090746245369
0.375       -2.47173771822073
0.5         1.5790933890986074
0.625       2.6455265698621426
0.75        12.70077506966297
0.875       -4.460578650579939
1.          0.
1.125       10.635035279982697
1.25        -14.954815776384153
1.375       -9.418727577588193
1.5   -14.80303034679264
```

Листинг 8. **Задание функции f в двоичном коде (файл bin_result.dat)**

```
.000  .00000
.001  -.00000
.010  -.00001
.011  -10.01111
.100  1.10010
.101  10.10100
.110  1100.10110
.111  -100.01110
1.000 .00000
1.001 1010.10100
1.010 -1110.11110
1.011 -1001.01101
1.100 -1110.11001
```

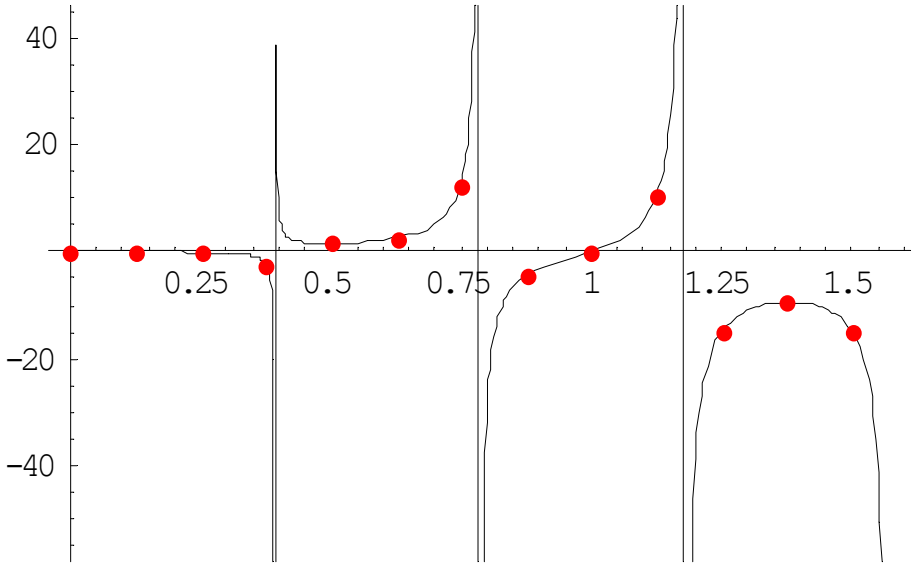

Рис. 4. График функции *f* и множество точек, по которым производится приближение

Таблица 2

Табличное задание функции *f* в двоичном коде вместе со знаковым разрядом

x0	x1	x2	x3	x4	y0	y1	y2	y3	y4	y5	y6	y7	y8	y9
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	0	0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	0	0	0	0	0	0	0	1
0	0	0	1	1	1	0	0	1	0	0	1	1	1	1
0	0	1	0	0	0	0	0	0	1	1	0	0	1	0
0	0	1	0	1	0	0	0	1	0	1	0	1	0	0
0	0	1	1	0	0	1	1	0	0	1	0	1	1	0
0	0	1	1	1	1	0	1	0	0	0	1	1	1	0
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	1	0	1	0	1	0	1	0	1	0	0
0	1	0	1	0	1	1	1	1	0	1	1	1	1	0
0	1	0	1	1	1	1	0	0	1	0	1	1	0	1
0	1	1	0	0	1	1	1	1	0	1	1	0	0	1

VHDL-модель функции *f* легко строится по табл. 2.

Заключение

Использование математических пакетов для создания моделей макроэлементов СБИС повышает надежность проектирования, так как математическая форма описания объектов проектирования является компактной и строгой. Кроме того, использование математических пакетов для приближения функций и предлагаемых программ преобразования выходных данных математических пакетов в стандартные входные данные САПР позволяет сокращать сроки проектирования цифровых СБИС.

Список литературы

1. Байков В.Д., Смолов В.Г. Специализированные процессоры: Итерационные алгоритмы и структуры. – М.: Радио и связь, 1985. – 288 с.

2. Балашов Е.П., Пузанков Д.В. Проектирование информационно-управляющих систем. – М.: Радио и связь, 1987. – 256 с.

3. Прокопеня А.Н., Чичурин А.В. Применение системы *Mathematica* к решению обыкновенных дифференциальных уравнений: Учеб. пособие. – Мн.: Изд-во БГУ, 1999. – 265 с.

4. Дьяконов В. Maple 7: Учеб. курс. – СПб.: Питер, 2002. – 627 с.

5. Кирьянов Д.В. Самоучитель MathCad 11. – СПб.: БХВ-Петербург, 2003. – 560 с.

6. IEEE Standard VHDL Language Reference Manual, IEEE Std 1076-1993.

7. Бибило П.Н. Синтез логических схем с использованием языка VHDL. – М.: Солон-Р, 2002. – 384 с.

8. Бибило П.Н., Каркоцкая И.П. О реализации на ПЛМ элементарных функций // Формализация и автоматизация логического проектирования. – Мн.: Ин-т техн. кибернетики АН Беларуси, 1993. – С. 99-109.

9. Бибило П.Н. Кремниевая компиляция заказных СБИС. – Мн.: Ин-т техн. кибернетики АН Беларуси, 1996. – 268 с.

10. Зотов Ю.В. Проектирование цифровых устройств на основе ПЛИС фирмы XILINX в САПР WebPack ISE. – М.: Горячая линия – Телеком, 2003. – 624 с.

Поступила 26.05. 04

Объединенный институт проблем
информатики НАН Беларуси,
Минск, Сурганова, 6
email:bibilo@newman.bas-net.by

P.N. Bibilo, D.A. Kochanov

CREATION OF VHDL-MODELS OF SCHEMAS DESIGNED FOR TABLE FORM FUNCTION CALCULATION

This article proposes a solution of a problem of creating VHDL-models of macroelements, which are designed to calculate the values of any analytical functions. The initial function's definition is given in analytical form in mathematical packages. The resulting function's description is provided in table form following its conversion into VHDL form.