Некоторые процедуры над логическими схемами и их реализация на языке ALEX.

Бозоян Ш.Е., Егиазарян В.С. (vladimir@ysu.am)

Российско-Армянский (Славянский) Государственный Университет

Введение

При проектировании больших интегральных схем (БИС) используются разные «элементарные» процедуры (соединение (в определенном смысле) двух схем, выделение некоторой «части» схемы или её удаление, замена некоторой подсхемы на другую, ей эквивалентной, схемой и т.п.), для осуществления которых создаются специальные программные инструменты. Эффективность проектирования главным образом зависит от качества комплекса этих инструментов. Созданием таких инструментов, в частности, занимаются такие крупные фирмы, как Cadens, Filips, Tetra Max и др.

В настоящее время в качестве языка описания аппаратур широко распространены Verilog HDL [1] и VHDL [2]. Главным достоинством этих языков является их простота и удобность для применения в автоматизации проектирования БИС. Однако они построены не дедуктивным принципом, поэтому практически не допускают создание алгебры преобразований схем, к тому же схему описывают неэкономно (примерно в два раза хуже оптимального). По статистическим данным через каждые 18 месяцев объем (число элементов) больших схем удваивается (закон Мура). В этих условиях ***сжатость*** описания схем и ***быстрота*** её моделирования становятся факторами первостепенной важности. Поэтому эти языки уже начинают не удовлетворять современным требованиям. В наших исследованиях в качестве языка описания схем используется другой, вполне удовлетворяющий современным требованиям язык – язык **Alex** [3,4,5].

1. Краткое неформальное описание языка Alex.

Идейным предшественником языка **Alex** является язык бесскобочной записи формул («польская запись»), обобщением которого является он сам. В отличие от «польской записи», которая описывает только формулы (на языке графов – входящее (направленное) дерево), **Alex** позволяет описать любую схему (начиная с вентильного уровня и выше), или на языке графов – любой направленный псевдограф. Описание (в дальнейшем - запись) схемы на этом языке

является конечной последовательностью символов из некоторого множества символов. В случае вентильного уровня это множество может быть множеством

$$\mathbf{A} = \{f_i(n),\ x_j(0),\ M_k(1),\ M_k(0),\ \tau_t(1)\} \quad ,\quad i,j,k,t = 1,2,...,m,...,$$

где символ $f_i(n)$ соответствует функциональному (логическому) элементу с n входами, $x_j(0)$ - входу схемы, $M_k(1)$ и $M_k(0)$ - точкам ветвления в схеме, $\tau_t(1)$- запоминающему элементу, задерживающего сигнал на один такт времени. Каждому символу s из **A** приписывается некоторое целое число $\omega(s)$, называемое ***весом*** этого символа. Для приведенного конкретного случая принимается

$$\omega(f_i(n)) = n-1,\ \ \omega(x_j(0)) = \omega(M_k(0)) = -1,\ \ \omega(M_k(1)) = \omega(\tau_t(1)) = 0.$$

Определяется также понятие ***веса последовательности символов*** из **A** следующим образом:

$$\omega(s_1 s_2 ... s_N) = \sum_{i=1}^{N} \omega(s_i).$$

Приведем наглядный пример схемы (рис.1) и её записи на языке **Alex.** Запись осуществляется по «маршруту», указанному тонкой линией со стрелками. Проходя по этому маршруту записываются все «попутные» элементы, если обход элемента выполняется по направлению «выход-вход», а при направлении «вход-выход» ничего не записывается. При первой «встрече» с точкой ветвления записывается символ типа $M(1)$, а при не первой «встрече» - символ типа $M(0)$. Входы схемы записываются символами типа $\mathbf{x}(0)$.

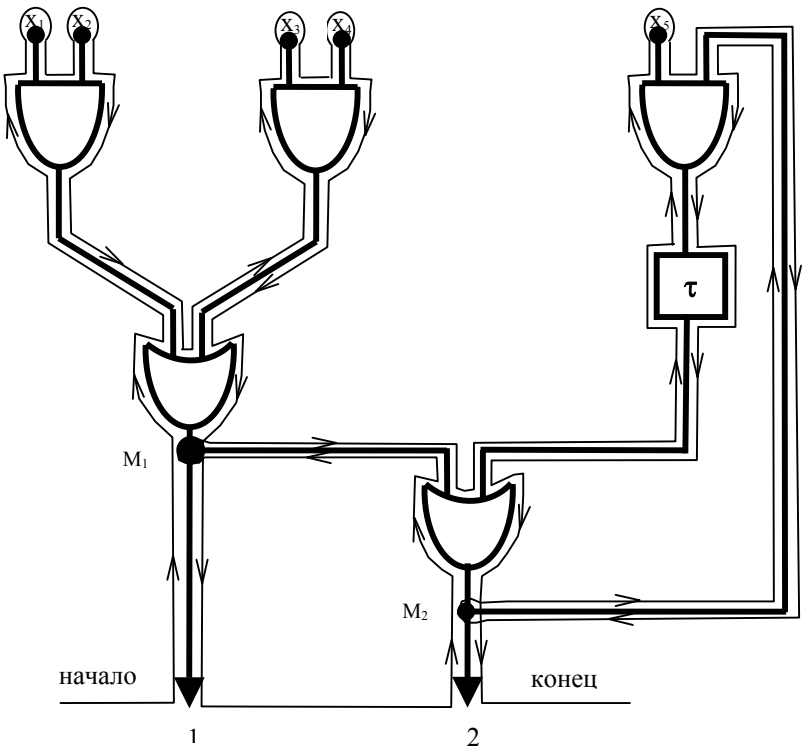


Рис. 1

Записью приведенной схемы является

$$M_1(1)V(2)\&(2)x_1(0)x_2(0)\&(2)x_3(0)x_4(0)M_2(1)V(2)M_1(0)\tau_1(1)\&(2)x_5(0)M_2(0).$$

Символы типа $M(1)$ и $M(0)$ называются ***метками.*** Очевидно, они введены с целью избежания от повторной записи той части схемы, выход которой разветвляется.

Вводится понятие ***описания*** метки $M_k(1)$ в данной последовательности $s_1s_2...s_N$ (оно обозначается через $\text{Оп}\,M_k(1)$). Это минимальный отрезок в $s_1s_2...s_N$, с весом -1, непосредственно следующий за $M_k(1)$. Определяется также понятие ***области действия*** символа $\tau(1)$ в $s_1s_2...s_N$ как минимальный отрезок с весом -1, непосредственно следующий за $\tau_t(1)$. Через эти два понятия определяется также понятие ***зависимости*** метки $M_i(1)$ от $M_j(1)$ в $s_1s_2...s_N$, содержательный смысл которого является функциональная зависимость точки ветвления M_i от M_j в схеме в данный момент времени. Доказывается следующая основная:

Теорема (о записи). Последовательность символов

$$s_1s_2...s_N \qquad (1)$$

является записью некоторой схемы с m $(m=1,2,...)$ выходами вентильного уровня тогда и только тогда, когда выполняются следующие условия:

а) $\omega(s_1s_2...s_N)=-m$;

б) $\omega(s_1 s_2 ... s_k) \geq 1 - m$ для любого $k < N$;

в) из того, что $M_i(0)$ содержится в (1), следует, что $M_i(1)$ также содержится в (1) (причем точно один раз);

г) для любого i $M_i(1)$ не зависит от $M_i(1)$.

Укажем некоторые особенности языка **Alex.**

Alex построен дедуктивным принципом, что позволяет создать алгебру эквивалентных и иных преобразований схем с помощью их записей. Он является оптимальным: длина (число символов) записи совпадает с числом рёбер графа схемы (если элементы трактовать как вершины, а связи между ними - рёбра). Далее, любая подсхема с одним выходом в общей записи схемы изображается как некоторый сплошной отрезок символов с весом -1 («подзапись» записи схемы). Это, в частности, позволяет любую подсхему легко заменить на ей эквивалентной (в определенном смысле) подсхемой, тем самым осуществить любые эквивалентные преобразования схем на языке **Alex.** Далее, функциональное моделирование осуществляется одним «просмотром» записи схемы, что исключает затрату времени на поисках, тем самым обеспечивается большая скорость моделирования. А это автоматически означает и увеличение скорости верификации, и тестирования и т.п. Наконец, теорема о записи позволяет проверкой условий теоремы (причем одним просмотром) осуществить эффективный синтаксический контроль правильности построения записи.

Настоящая работа претендует на создание начальных предпосылок для построения алгебры преобразования записей схем на языке **Alex.**

2. Основные процедуры и их осуществление на языке Alex.

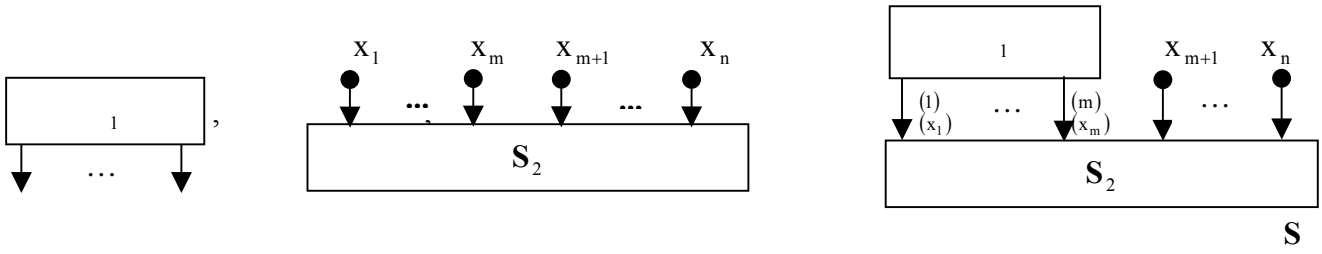
Рассмотрим схемы $\mathbf{S}_1$ и $\mathbf{S}_2$ с соответсвующими записями $\mathbf{h}(\mathbf{S}_1)$ и $\mathbf{h}(\mathbf{S}_2)$ на языке **Alex.** Пусть каждая из этих схем имеет несколько входов и несколько выходов. Пусть элементы этих схем пронумерованы натуральными числами таким образом, что для каждой схемы разным элементам приписаны разные номера. Точки ветвления выходов элементов также пронумерованы, но таким образом, что номер точки ветвления выхода данного элемента совпадает с номером этого элемента. Будем говорить, что схемы $\mathbf{S}_1$ и $\mathbf{S}_2$ ***не пересекаются*** или являются ***независимыми***, если номер любого элемента из $\mathbf{S}_1$ не совпадает с номером любого элемента из $\mathbf{S}_2$. Например, если элементы $\mathbf{S}_1$ пронумерованы числами $1,2,...,n_1$, а элементы $\mathbf{S}_2$ - числами $\mathbf{n}_1 + 1, \mathbf{n}_1 + 2, ..., \mathbf{n}_1 + \mathbf{n}_2$, то $\mathbf{S}_1$ и $\mathbf{S}_2$ не пересекаются.

Ниже рассматриваются процедуры над схемами на языке **Alex.** Рассмотренные в этих процедурах схемы $\mathbf{S}_1$ и $\mathbf{S}_2$ всюду считаются независимыми.

2.1. Последовательное соединение двух схем. Рассмотрим схемы $\mathbf{S}_1$ и $\mathbf{S}_2$, где $\mathbf{S}_1$ имеет m выходов, а $\mathbf{S}_2$ - n входов $x_1,...,x_n$ $(n \geq m)$ (рис.2.а). Пусть все выходы $\mathbf{S}_1$ требуется соединить с определенными m входами схемы $\mathbf{S}_2$. С целью не усложнения обозначений допустим, что эти m выходы соединяются с первыми $x_1,...,x_m$ входами схемы $\mathbf{S}_2$ (рис.2.б). Разумеется, такое допущение не снижает общность постановки задачи. Очевидно, полученная конструкция $\mathbf{S}$ является схемой с записью

$$\mathbf{h(S)} = \left[\mathbf{h(S}_2\mathbf{)}\right]^{\mathbf{x_1(0),...,x_m(0)}}_{\mathbf{h(S_1,1),...,h(S_1,m)}},$$

где $\mathbf{h(S}_1\mathbf{,i)}$ $(i = 1,2,...m)$ является записью схемы $\mathbf{S}_1$ относительно её i-ого выхода (т.е. слева i-ый минимальный отрезок записи $\mathbf{h(S}_1\mathbf{)}$ с весом -1), а обозначение $[\mathbf{B}]^{\mathbf{x}}_{\mathbf{y}}$ означает результат подстановки $\mathbf{y}$ вместо $\mathbf{x}$ в $\mathbf{B}$, а если $\mathbf{B}$ не содержит $\mathbf{x}$, то этим результатом считается $\mathbf{B}$

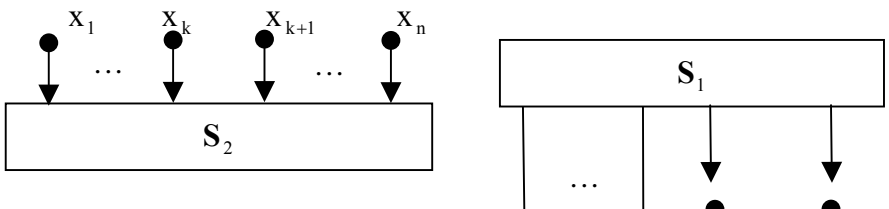


а *б*

Рис. 2

2.2. Частичное последовательное соединение двух схем. Пусть схемы $\mathbf{S}_1$ и $\mathbf{S}_2$ имеют соответственно m выходов и n входов (рис.3.а). Пусть выходы с номерами $1,2,...,\mathbf{k}$ $(\mathbf{k} \leq \mathbf{m},\ \mathbf{k} \leq \mathbf{n})$ схемы $\mathbf{S}_1$ соединяются соответственно со входами $x_1,...,x_k$ (рис.3.б). Здесь также сдедует заметить, что фиксация номеров выходов и входов схем на конкретные значения не нарушает общность постановки задачи, она сделана с целью упрощения обозначений. Очевидно, полученная конструкция $\mathbf{S}$ также является схемой, записью которой является

$$\mathbf{h(S)} = \left[\mathbf{h(S}_2\mathbf{)}\right]^{\mathbf{x_1(0)\ ,...,x_k(0)}}_{\mathbf{h(S_1,1),\ ...,h(S_1,k)}}\ \mathbf{h(S_1,k+1)h(S_1,k+2)...h(S_1,m)}$$



а *б*

Рис. 3

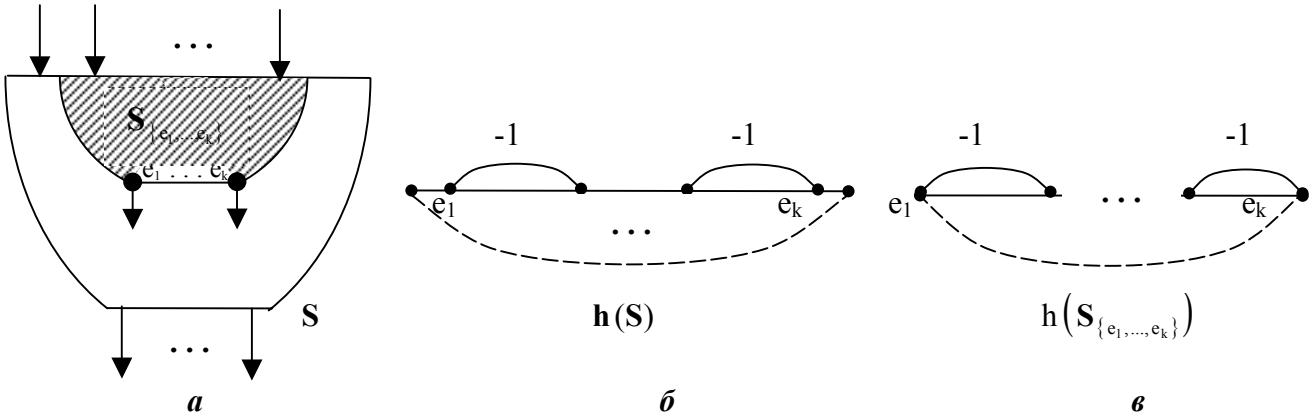
2.3. Параллельное соединение двух схем. Пусть $\mathbf{S}_1$ и $\mathbf{S}_2$ являются схемами (рис.4.а). Тогда объединение этих схем в определенном порядке назовем параллельным соединением этих схем (рис.4.б).

а *б* **S**

Рис. 4

Очевидно, $h(\mathbf{S}) = h(\mathbf{S}_1)h(\mathbf{S}_2)$.

2.4. Выделение максимальной подсхемы. Подсхема схемы $\mathbf{S}$ называется **максимальной,** если её входы являются входами $\mathbf{S}$. На рис.5.а показана максимальная подсхема $\mathbf{S}_{\{e_1,...,e_k\}}$ схемы $\mathbf{S}$, выходами которой являются выходы элементов $e_1,...,e_k$.



а *б* *в*

Рис. 5

В записи $h(\mathbf{S})$ выделим минимальные отрезки с весом -1, началами которых являются символы, соответствующие элементам $e_1,...,e_k$ (рис.5.б). Состыкуем эти отрезки в том порядке, в каком они расподожены в $h(\mathbf{S})$. Полученный отрезок будет записью подсхемы $\mathbf{S}_{\{e_1,...,e_k\}}$ в так

называемом, **неактивизированном** виде. Дело в том, что этот отрезок может содержать метку $M_i(0)$ и не содержать $M_i(1)$. В таком случае заменив одно из вхождений $M_i(0)$ на $M_i(1)\text{Оп}\,M_i(1)$, и эту процедуру производя относительно всех таких меток, очевидно, получим обычную запись подсхемы $\mathbf{S}_{\{e_1,\ldots,e_k\}}$. Эту запись будем называть записью в **активизированном** виде.

2.5. Удаление максимальной подсхемы. Эта процедура тесно связана с процедурой «выделения максимальной подсхемы» (п. 2.4). При этом в записи $\mathbf{h}(\mathbf{S})$ отрезки с весом -1, началами которых являются символы, соответствующие элементам $e_1,\ldots,e_k$, соответственно заменяются на $z_1(0),\ldots,z_k(0)$ ($z_i(0)$ $(i=1,2,\ldots,k)$ не входят в $\mathbf{h}(\mathbf{S})$), а те отрезки $M_i(1)\text{Оп}\,M_i(1)$, которыми были заменены символы $M_i(0)$ при осуществлении процедуры «выделении максимальной подсхемы», заменяются на $M_i(0)$.

2.6. Выделение подсхемы. Эта процедура предполагает выделение из схемы **S** подсхему $\Delta\mathbf{S}$ (не объязательно максимальную), если указаны элементы $e_1,\ldots,e_k$, выходы которых являются выходами подсхемы, и элементы $g_1,\ldots,g_m$, выходы которых являются входами подсхемы (рис.6). Эта процедура осуществляется двумя этапами. На первом этапе из **S** выделяется максимальная подсхема $\mathbf{S}_{\{e_1,\ldots,e_k\}}$ (рис.6.б), а затем из $\mathbf{S}_{\{e_1,\ldots,e_k\}}$ удаляется максимальная подсхема $\mathbf{S}_{\{g_1,\ldots,g_m\}}$, в результате чего получится искомая подсхема $\Delta\mathbf{S}$ (рис.6.в).

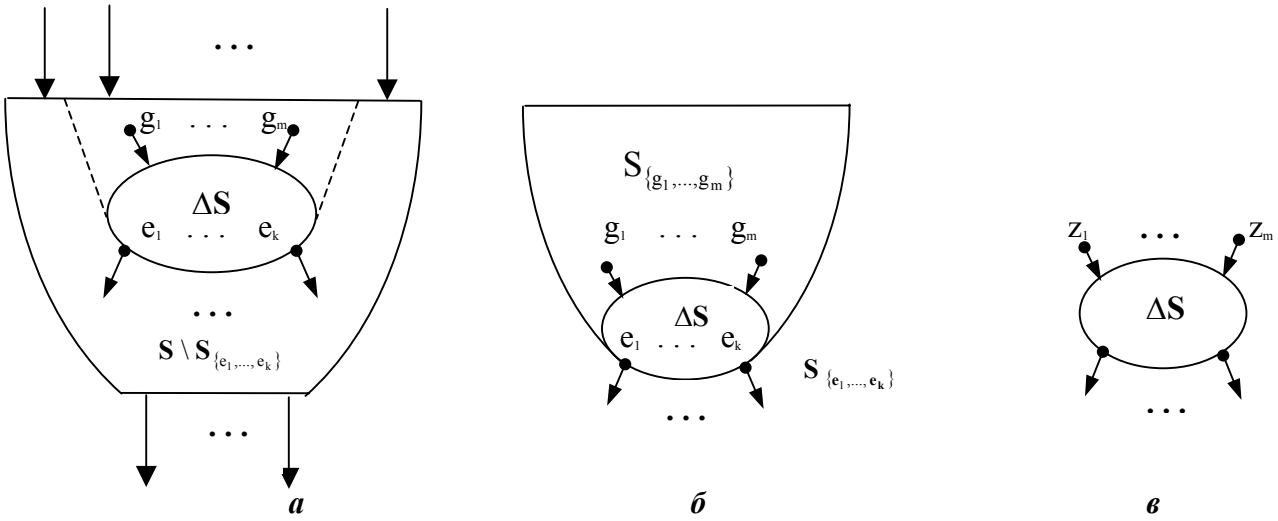


Рис. 6

Обратная процедура – «вставка» $\Delta\mathbf{S}$ в «своё предыдущее место», осуществляется применением процедуры 2.1. последовательным соединением $\mathbf{S}_{\{g_1,\ldots,g_m\}}$ и $\Delta\mathbf{S}$, а затем последовательным соединением $\mathbf{S}_{\{e_1,\ldots,e_k\}}$ и $\mathbf{S}\setminus\mathbf{S}_{\{e_1,\ldots,e_k\}}$.

Литература

1. S.Palnitkar. Verilog HDL: A Guide to Digital Design and Synthesis. SunSoftPress. Prentice Hall, 1996.
2. Маршнер Ф.Е. VHDL для моделирования, синтеза и формальной верификации аппаратуры. М., 1995, с. 1-13.
3. Бозоян Ш.Е. Язык описания функциональных схем. Изв. АН СССР, Техническая кибернетика, 4, 1978.
4. Бозоян Ш.Е. Приспособление языка Лукасевича к описанию функциональных схем. ДАН Арм. ССР, т.63, 4, 1979.
5. Бозоян Ш.Е. Алгебраическое описание направленных графов. ДАН Арм. ССР, т.76, 1, 1983.