Министерство образования и науки Украины
Харьковский национальный университет радиоэлектроники

Шкиль Александр Сергеевич

1 Основы логического моделирования цифровых устройств

Харьков
2009

Содержание

Краткая аннотация

В данном разделе изложены основные принципы логического моделирования :

Параллельность

Событиность

Асинхронность

Иерархичность.

Рассмотрены модели сигналов в системах логического мождедирования цифровых устройств (девятисимвольный алфавит стандарта IEEE_1164 типа std_logic). Перечислены способы реализации моделей цифровых в системах логического моделирования : компилятивные и интерпретативные модели.

Дано определение и показан способ построения двухуровневой структурно-функциональной модели в интерпретативных системах моделирования. Материал данного раздела используется при изучении как методов моделирования, так и методов структорного построения тестов для цифровых устройств.

Введение

При анализе цифровых устройств электрические значения сигналов заменяются их логическими эквивалентами. Задачей моделирования цифровых устройств является вычисление значений сигналов в цифровых схемах схемы при различных начальных условиях. Среди всего множества методов моделирования технических объектов можно выделить особое направление - ***simulation*** (*логическое моделирование*) - моделирование электронных (цифровых) объектов с использованием компьютеров. Особенностью этого вида моделирования является то, что в качестве моделей объектов (компонентов) используются непосредственно их законы функционирования в логическом (дискретном) виде с использованием разных форм представления.

Логическое моделирование цифровых устройств обычно выполняется с использованием специальных компьютерных программ, которые принято называть **системами логического моделирования**.

В основе анализа цифровых устройств лежит понятие сигнала - логического эквивалента реальным электрическим сигналам, который может принимать значение '0' или '1'. Цифровые схемы состоят из компонентов, таких как логические элементы, триггеры, счетчики, соединенных между собой проводниками, которые принято называть "эквипотенциальными линиями" или просто линиями. Логически объединенные жгуты проводников принято называть многоразрядными шинами или просто шинами.

Машинно-ориентированное описание цифровой схемы для целей анализа цифровых схем должно включать в себя описание поведения каждого компонента в терминах зависимости выходных значений сигналов компонентта от входных, и описание связей компонентов схемы между собой.

Теория

1.1 Основные принципы логического моделирования

Параллельность

Параллелизм является одним из базовых принципов функционирования цифровых устройств. Каждый новый двоичный набор подается на все входы схемы параллельно. При подаче на схему очередного двоичного набора вычисления новых значений сигналов на выходах элементов схемы осуществляется по мере изменения сигналов на элементах предыдущих рангов, т.е параллельно. Но программы логического моделирования реализуются как пользовательские приложения на обычных "последовательных" компьютерах, в которых обработка элементов происходит последолвательно друг за другом. Для "имитации" параллельности работы схемы применяют или ранжирование схем, т.е расстановку элементов схем по уровням срабатывания, или специальные способы анализа сигналов во времени с помощью последовательности тразакций, т.е сигналы в схемах представляются парой "значение-время".

Рисунок 1.1 - Параллельность (Анимированый)

Выше (рис 1.1) на схеме показана смена наборов 111 на 011 (сигналы в схеме изменяются параллельно).

Событийность

Под *событием* в цифровой схеме понимается любое изменение сигнала на линии схемы (с 0 на 1 или с 1 на 0 в двоичном алфавите). Принцип событийности при логическом моделированиии состоит в том, что новые значения вычисляются на выходах только тех элементов, на входах которых имеются события. Применение принципа событийности позволяет в среднем в 10-20 раз уменьшить временные затраты на реализацию алгоритма моделирования по сравнению с алгоритмом сквозного моделироввания, при реализации которого моделируются все элементы без исключения.

Рисунок 1.2 - Событийность (Анимированый)

На схеме выше (рис 1.2) показано, что событие на входе X1 не приводит к изменению на выходах элементов 2 и 3, следовательно, на элементе 4 событие не происходит и он не моделируется.

Асинхронность

Электрические схемы не могут мгновенно реагировать на изменение входных сигналов. Это обусловлено тем, что физическим устройствам, таким как транзисторы, которые используются для реализации логики вентильного уровня, требуется некоторое время для переключения с одного логического значения на другое. Следовательно, изменение значения сигнала на входе логического элемента не приведет к мгновенному изменению значения выходного сигнала, т.е. потребуется некоторое время на то, чтобы событие со входов элемента распространилось к его выходам. Такой период времени принято называть **задержкой** элемента. Электрический ток, передающий сигнал по проводам, также течет с некоторой конечной скоростью. Таким образом, в реальности передача сигналов по шинам происходит с задержкой, и ее величина зависит от длины провода. Эту задержку нельзя игнорировать, особенно в высокоскоростных, сверхплотных схемах. Важно заметить, что линии имеют значительно меньшие задержки, чем элементы и во многих системах моделирования задержки линий игнорируются. Задержки элементов принято обозначать буквой Δ (дельта). Методы моделирования, в которых задержки элементов не равны 0, принято называть асинхроными, в отличие от синхроных методов, когда Δ =0. Преимуществами синхронных методов моделирования является их высокое быстродействие (в среднем в 5-10 раз выше, чем у асинхронных методов).

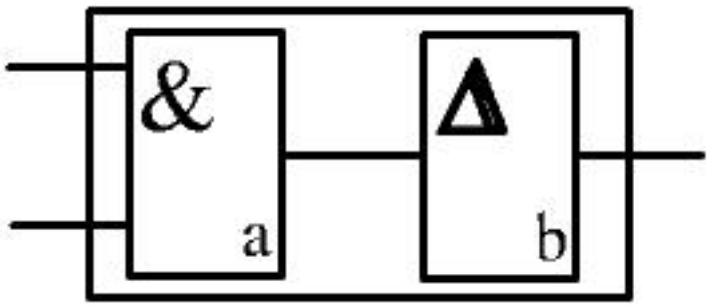


Рисунок 1.3

Любой элемент цифровой системы может быть представлен функцианалом (a) и задержкой Δ (b).

На схеме ниже (рис 1.4) показано, что при переходе от набора 111 к 011 выход остается в 1 но при Δ =1, изменение на выходе эл. 2 будет происходить позже, чем на элементе 3, поэтому на выходе Y будет кратковременный нулевой выброс длительностью D.

Рисунок 1.4 - Асинхронность (Анимированый)

Иерархичность

Иерархичность является одним из фундаментальных принципов описания любых сложных систем, в том числе и цифровых схем. Любая система разбивается на уровни иерархии, причем компоненты данного уровня иерархии могут представляться структурной суперпозицией компонентов более низкого уровня иерархии. На нижнем уровне иерархии компоненты описываются поведенческими (функциональными) моделями. Такие компоненты принято называть **примитивными элементами**, просто **примитивами** или сокращенно **ПЭ**. Двухуровневую иерархическую модель, состоящую из примитивов и связей между ними принято называть **структурно-функциональной моделью**.

На рисунке 1.5 ниже показано представление функционального элемента (ФЭ) в виде структурно-функциональной модели (слева) или чисто функциональной табличной модели (справа).

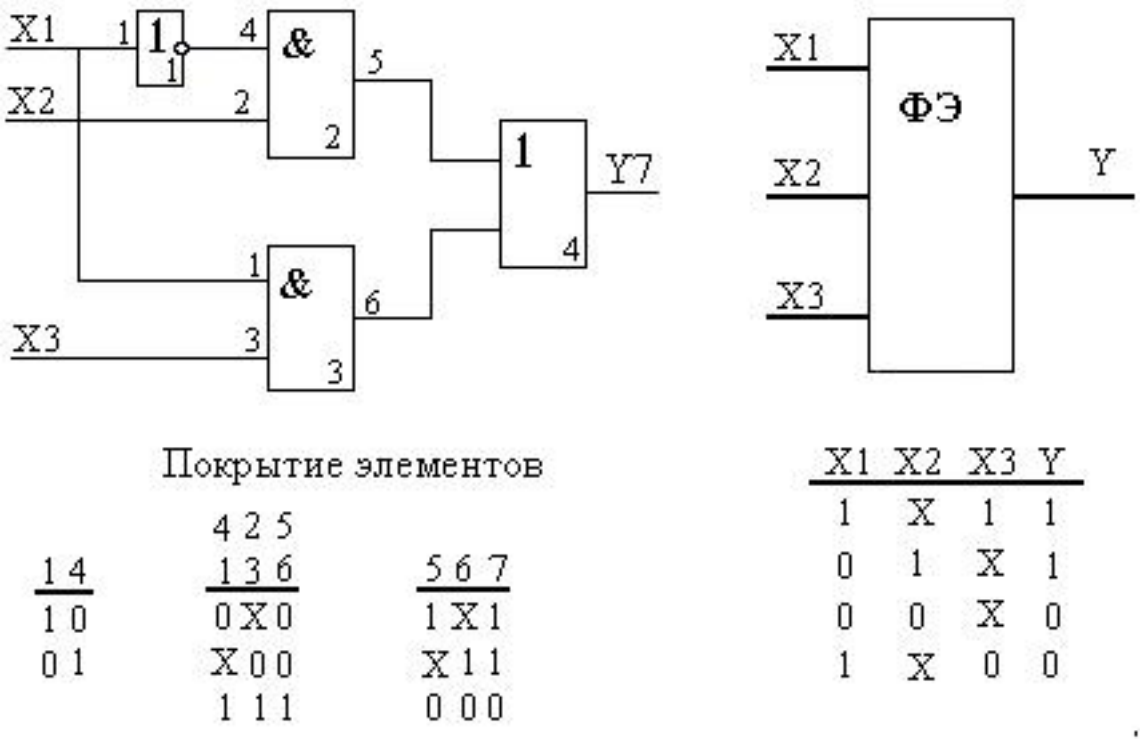


Рисунок 1.5 - Иерархичность

Классификация типов моделей цифровых устройств и подробное описание принципов построения структурно-функциональных моделей приведены в следующих подразделах.

1.2 Модели цифровых устройств

Способ и форма представления моделей

Среди всего множества методов моделирования технических объектов можно выделить особое направление - логическое моделирование, т.е. моделирование цифровых объектов с использованием компьютеров. Особенностью логического моделирования (***simulation***) является то, что в качестве моделей объектов (компонентов) используются непосредственно их законы функционирования в разных формах представления.

Все модели цифровых объектов могут классифицироваться по **способу представления** и по **форме представления**. Способ представления, как правило, определяет или моделируемый параметр или тип процедуры моделирования.

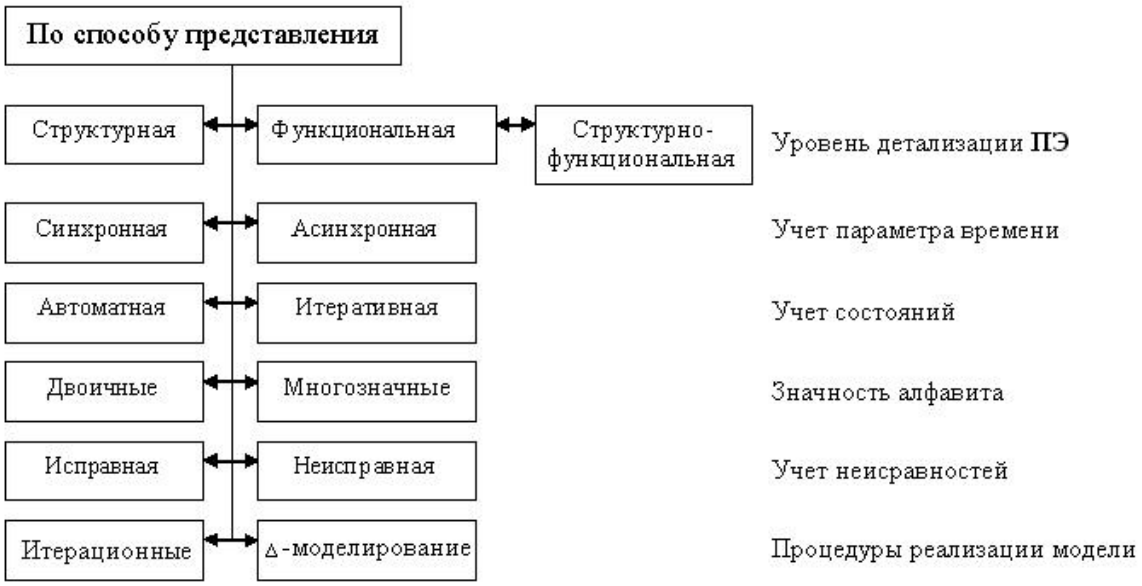


Рисунок 1.6

Форма представления модели определяет как форму визуального представления модели, так и формат представления модели в структурах данных системы моделирования.

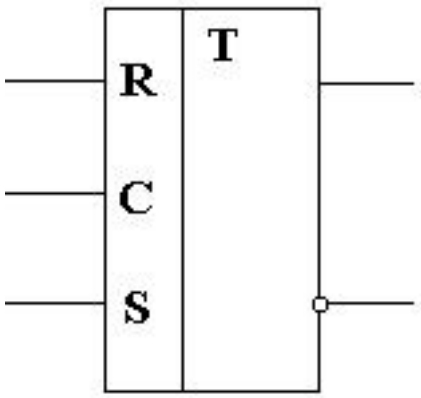
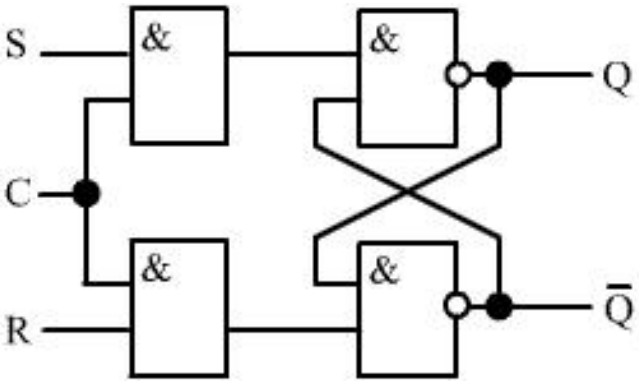
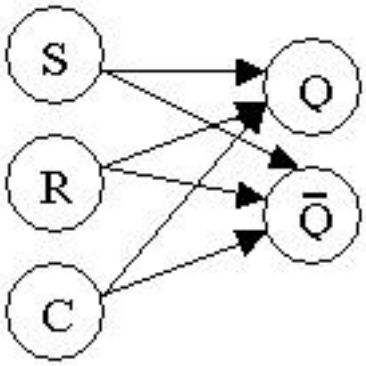
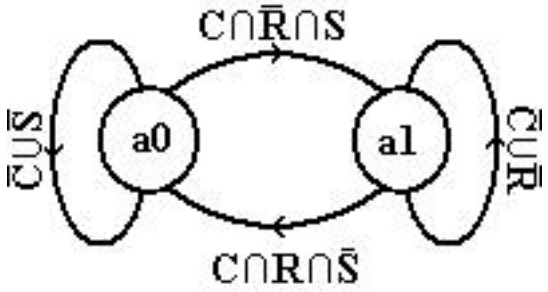
Рисунок 1.7

На практике обычно используется синхронизированный вариант RS-трриггера - триггер CRS.

Синхронизация в нем, как и во всех однотактовых триггерах, происходит по уровню, а не по фронту. В данном триггере к двум основным входам (R и S) добавлен синхровход C (Clk или Clock pulse). Если он равен "0", выходы не изменяют свое состояние, т.е. триггер не активен; при подаче на синхровход "1" устройство начинает работать как RS триггер.

Пример: CRS - триггер с прямыми входами

Таблица 1.1

Графическое представление	
Схема	Внутренняя структура
	
Структурный граф	Граф переходов
	

Табличное изображение	
Полная таблица переходов	Сокращенная таблица переходов

t				t+1
C	R	S	Q	Q
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	x
1	1	1	1	x

Матрица переходов

t			t+1
C	R	S	Q
0	x	x	Q(t)
1	0	0	Q(t)
1	0	1	1
1	1	0	0
1	1	1	x

Карта Карно

Q(t)	-	Q(t+1)	C	R	S
0	-	0	a	b	a & c
0	-	1	1	0	1
1	-	0	1	1	0
1	-	1	d	d & e	f

SQ / CR	00	01	11	10
00	0	1	1	0
01	0	1	1	0
11	0	0	x	x
10	0	1	1	1

Аналитические
Уравнение функционирования триггера, полученное путем минимизации карты Карно

$Q(t+1) = \bar{C}Q \cup \bar{R}Q \cup CS$
Алгоритмические
ЯОА(VHDL)

```
architecture JK_tr_arc of JK_tr is
begin
process(clk, reset, set) is
variable q1 :std_logic;
variable p :std_logic_vector(0 to 1);
begin
    p :=set & reset;
case p is
    when "00"=>q1 :='X';
    when "10"=>q1 :='0';
    when "01"=>q1 :='1';
    when "11"=>q1 := Q;
    when others => q1 :='X';
end case;
    Q<=q1; Not_Q<=not q1;
end process;
end JK_tr_arch;
```

Модели сигналов

В реальных цифровых схемах циркулируют электрические сигналы, но для анализа цифровых схем используются их логические эквиваленты. Международный стандарт IEEE_1164 определил 9 логических значений сигналов, которые используются при описании цифровых устройств с использованием языков описания аппаратуры (в частности VHDL) и при выполнении логического моделирования в промышленных САПР.

В языке VHDL определен тип std_logic, сигналы и переменные которого могут принимать значения, определенные в указанном стандарте. При описание моделей цифровых устройств и логическом моделировании обычно используются шесть значений:

'U' - (Uninitialized) неопределенное значение. Значение, которым инициализируются сигналы и переменные типа std_logic по умолчанию. При нормальной работе проекта, оно обычно не появляется в моменты времени после первого назначения сигнала;

'X' - (Forcing Unknown) состояние соответствующее неопределенному значению (0,4 - 2,4 В);

'0' - (Forcing 0) состояние соответствующее значению логического нуля (<0,4 В);

'1' - (Forcing 1) состояние соответствующее значению логической единицы (>2,4 В);

'Z' - (High Impedance) состояние высокого импеданса.

'-' - (Don't care) безразличное состояние (значение). Используется для задания маски, т.к. при сравнении с любым значением всегда возвращает true.

В системах компилятивного логического моделировния (например, Active-HDL фирмы ALDEC) одним из видов представления результатов моделирования являются т.н. "вектроные диаграммы" (waveform) отображающие изменения интересующих нас сигналов во времени. Обозначения на векторных диаграммах рассмотренных раннее состояний приведены ниже

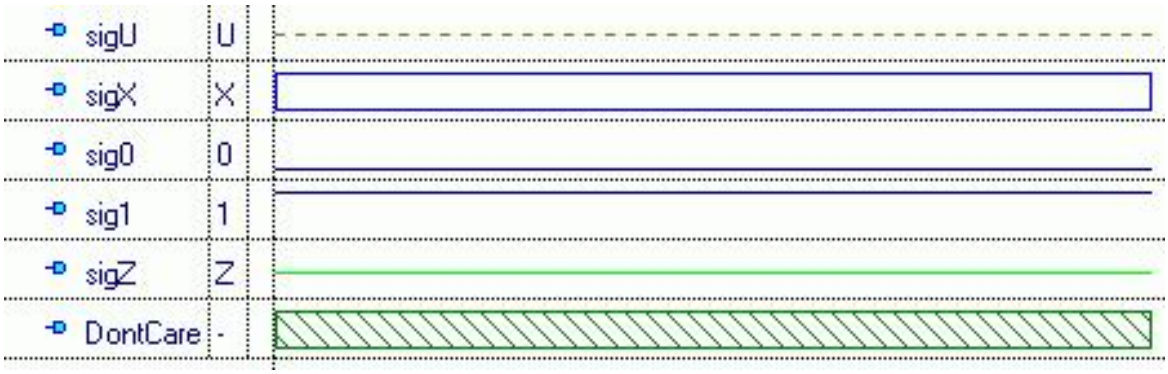


Рисунок 1.8

Необходимо отметитить, что в простых (учебных) системах моделирования безразличное состояние может объединяться с неопределенным, а "высокий импеданс" в условиях отсутствия "проводной логики" может ассоциииироваться с "логической 1". В результате минимальный алфавит моделирования сигналов в системах логического моделирования имеет вид { 0, 1, X, U }.

Ниже представлены основные свойства данных состояний:

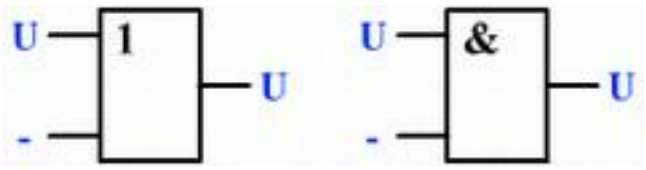


Рисунок 1.9

Как видно из рисунка, если хотя бы на одном из входов элемента И либо ИЛИ появится состояние 'U', то оно будет перенесено и на выход. Такое поведение элементов И, ИЛИ используется для выявления ошибок в проекте, поскольку, как уже было сказано выше, состояние 'U', в обшем случае, во время моделирования появлятся не должно.

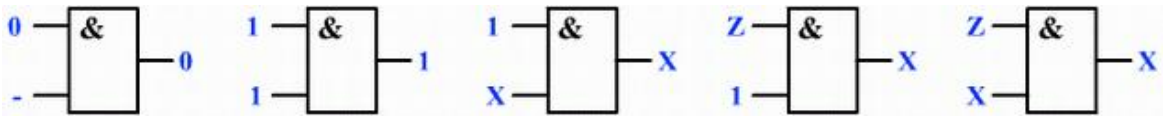


Рисунок 1.10

Элемент И устроен таким образом, что появление состояния логического нуля на одном из его входов и отсутствие состояния 'U' на остальных приводит к появлению на его выходе состояния логического нуля. В случае подачи на все входы данного элемента состояния логической единицы на выходе появится также состояние логической единицы. Подача на входы элемента любых других состояний, за исключением 'U', приведет к появлению на выходе состояния 'X', что показано ниже.

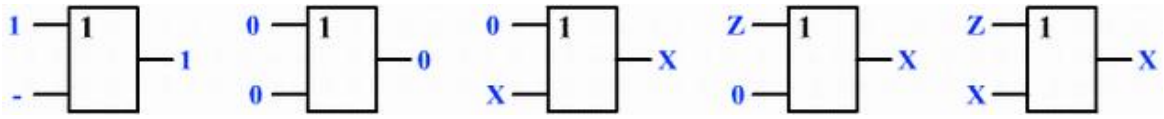


Рисунок 1.11

Элемент ИЛИ работает следующим образом образом: появление состояния логической единицы на одном из его входов и отсутствие состояния 'U' на остальных приводит к появлению на его выходе также состояния логической единицы. В случае подачи на все входы данного элемента состояния логического нуля на выходе появится также состояние логического нуля. Подача на входы элемента любых других состояний, за исключением 'U', приведет к появлению на выходе состояния 'X' (аналогично работе элемента И), что показано ниже.

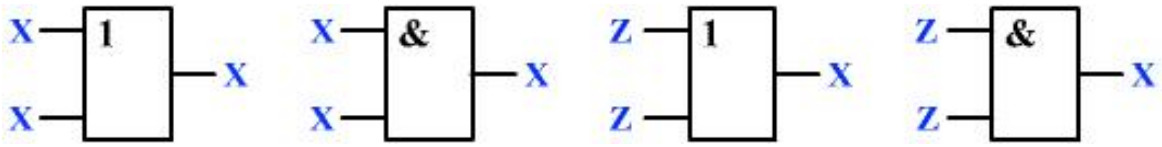


Рисунок 1.12

1.3 Структурно-функциональные модели ЦУ

Компилятивные и интерпретативные модели ЦУ

Любые цифровые схемы представимы совокупностью функциональных компонентов, которые принято называть **примитивными элементами (ПЭ)** или просто *примитивами*, и электрическими (гальваническими) связями, соединяющими указанные компоненты. Эти связи принято называть *линиями* или *переменными*. Логические эквиваленты электрических значений на линиях принято называть *сигналами*. Совокупность компонентов и соединяющих их линий принято называть **структурно-функциональной моделью (СФМ)**.

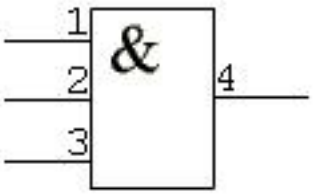
По способу описания компонентов СФМ и, соответственно, по видам процедур их обработки, модели компонентов делятся на **компилятивные** и **интерпретативные**.

Компилятивные модели представляют собой *процессы*, реализованные в виде фрагментов компьютерных программ на языках программирования (обычно, достаточно низкого уровня). Процесс представляет собой способ описания ПЭ любой сложности в виде специальных процедур на языках программирования (C или Pascal) или описания аппаратуры (VHDL), которые позволяют вычислять выходные значения сигналов компонентов на основе известных значений входных сигналов. При реализации *компилятивных* процедур моделирования в иерархических структурно-функциональных моделях происходит *уравнивание* иерархии, т.е сведение иерархической модели к двухуровневой (процессы и связи между ними), компиляция этой модели в исполняемый код компьютерной программы с последующим выполнением моделирования.

Достоинством компилятивных моделей является их высокое быстродействие, но на таких моделях нельзя выполнять "обратное" моделирование (вычисление входных значений по известным входным), и возникают большие проблемы при реализации на таких моделях моделирования неисправностей.

Примером компилятивной системы моделирования является подсистема асинхронного событийного моделирования в рамках САПР Active-HDL.

Интерпретативные модели компонентов обычно представляют собой табличные описания законов их функционирования, выполненные в соответствующих алфавитах. Такие табличные описания принято называть *кубическими покрытиями* (КП) или минимизированными таблицами истинности. Ниже приведены КП основных логических элементов [1].

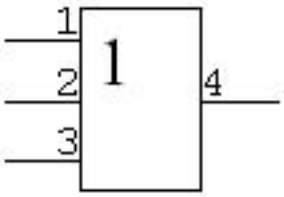


ЗИ

1	2	3	4
0	X	X	0
X	0	X	0
X	X	0	0
1	1	1	1

ЗИ-НЕ

1	2	3	4
0	X	X	1
X	0	X	1
X	X	0	1
1	1	1	0



ЗИЛИ

1	2	3	4
1	X	X	1
X	1	X	1
X	X	1	1
0	0	0	0

ЗИЛИ-НЕ

1	2	3	4
1	X	X	0
X	1	X	0
X	X	1	0
0	0	0	1

Рисунок 1.13

Процесс вычисление выходных значений сигналов компонента по известным входным при использовании интерпретативных (табличных) моделей компонента принято называть *прямой импликацией*, а вычисление входных значений по известным выходным - *обратной импликацией*.

Недостатком интерпретативных моделей является их невысокое быстродействие, но на таких моделях можно выполнять обратную импликацию (моделирование назад), которое широко используется в задачах доопределения, в частности при генерации тестов. Поэтому в

диагностических задачах применяются преимущественно интерпретативные схемные модели. Кроме того интерпретативные модели позволяют эффективно выполнять моделирования неисправностей. Многоуровневые иерархические интерпретативные модели обычно не применяются, а используются только двухуровневые СФМ.

Примером интерпретативной системы моделирования является учебная система синхронного моделирования неисправностей DCP (Deductive Circuit Processor).

Здесь и далее все изложение ведется относительно интерпретативхых систем моделирования, если не оговорено иное.

Построение интерпретативной структурно-функциональной модели схемы

Построение структурно-функциональной модели осуществляется путем ранжирования (нумерации) линий схемы и нумерации ее ПЭ.

1. Выполняется нумерация по порядку внешних входов схемы, т.е. линий не имеющих предшественников.

2. Следующими по порядку номерами нумеруются выходы примитивов, входы которых уже занумерованы, и которые не являются внешними выходами, т.е. имеют преемников.

Пункт 2 выполняется до тех пор, пока не окажутся занумерованными все внутренние линии схемы.

4. Следующими по порядку номерами нумеруются внешние выходы схем, т.е. линии не имеющие преемников.

5. Нумеруются примитивные элементы в порядке следования номеров выходных линий данных элементов.

6. Составляется список различных типов примитивов и для каждого типа ПЭ строиться табличная модель его функционирования в форме кубического покрытия (для моделирования) и D-покрытия (для построения тестов).

Пример построения СФМ для цифровой схемы (рис.1.14) взят из подраздела 3.2 "Основы структурного тестирования".

Рисунок 1.14 (Анимированый)

Практика

1.4 Практическое занятие: Кубическое исчисление. Модели элементов и основные операции.

Тема:

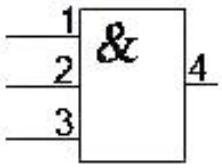
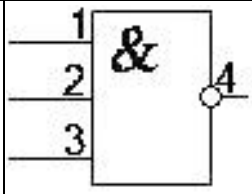
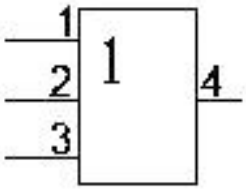
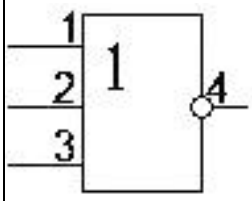
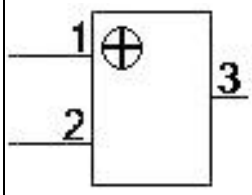
Построение кубических покрытий комбинационных схем и применение операций кубического исчисления

Задание 1

Составить кубическое покрытие для элементов: 3И, 3ИЛИ, 3И-НЕ, 3ИЛИ-НЕ, 2XOR

Ниже в таблице 1.2 приведен пример кубических покрытий для элементов 3И, 3ИЛИ, 3И-НЕ, 3ИЛИ-НЕ, 2XOR :

Таблица 1.2

3И	3И-НЕ	3ИЛИ	3ИЛИ-НЕ	2XOR
				
1 2 3 4 0 X X 0 X 0 X 0 X X 0 0 1 1 1 1	1 2 3 4 0 X X 1 X 0 X 1 X X 0 1 1 1 1 0	1 2 3 4 1 X X 1 X 1 X 1 X X 1 1 0 0 0 0	1 2 3 4 1 X X 0 X 1 X 0 X X 1 0 0 0 0 1	1 2 3 0 0 0 0 1 1 1 0 1 1 1 0

Следует отметить, что структура кубических покрытий логических элементов регулярна, т.е. с увеличением числа входов логических елементов структура кубических покрытий не изменяется.

Варианты заданий для самостоятельного выполнения:

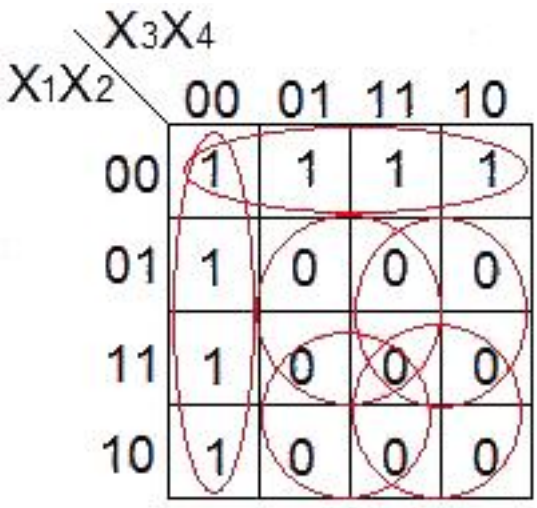
Составить кубическое покрытие для элементов: 4И, 4ИЛИ, 4И-НЕ, 4ИЛИ-НЕ, 3XOR

Задание 2

На основе заданного кубического покрытия C^1 построить C^0, составить схему без инверсий на входах и промоделировать заданный входной набор

Дано: $C^1 = \begin{Bmatrix} 00XX \\ XX00 \end{Bmatrix}$, входной набор T = { 0 0 1 1 }

Исходя из предположения, что рассматриваемая логическая функция является полностью опрнеделенной, одним из самых простых способов получения C^0 является заполнение карты Карно "1" на основании C^1, после чего пустые клетки заполняются "0". Затем выполняется минимизация по "0" (обведение групп нулей) и получение на этой основе C^0 .



$$C^0 = \begin{Bmatrix} 1XX1 \\ 1X1X \\ X1X1 \\ X11X \end{Bmatrix}$$

Рисунок 1.15

Четыре обведения по 4 нуля дают 4 куба второго ранга, которые и составляют покрытие C^0

Для построения схемы без инверсий на входах на основании C^1 строится аналитическое выражение функции в форме ДНФ (каждому кубу C^1 ставится в соответствие терм ДНФ,

координаты куба, не равные X, образуют буквы терма, и над ними ставятся инверсии, если на соответствующих координатах стоит "0"). К полученому выражению применяется правило двух инверсий (закон Де-Моргана) .

$$Y = \overline{X_1}\,\overline{X_2} \vee \overline{X_3}\,\overline{X_4} \Rightarrow Y = \overline{\overline{\overline{X_1}\,\overline{X_2}} \vee \overline{\overline{X_3}\,\overline{X_4}}} = \overline{\overline{\overline{X_1}\,\overline{X_2}} \,\&\, \overline{\overline{X_3}\,\overline{X_4}}} = \overline{(X_1 \vee X_2)\,\&\,(X_3 \vee X_4)}$$

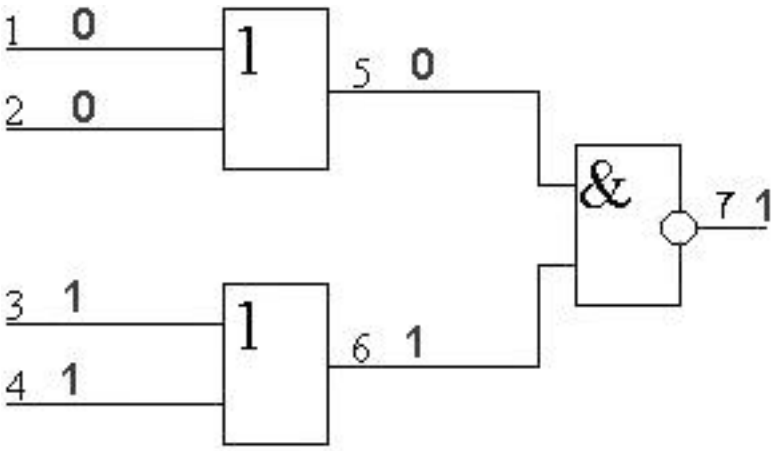


Рисунок 1.16

Синхронное моделирование заданного входного набора состоит в получении значений сигналов на всех нумерованных линиях схемы. В результате получается вектор :

1	2	3	4	5	6	7
0	0	1	1	0	1	1

Варианты заданий для самостоятельного выполнения:

Кубические покрытие C^1 , входной набор T :

Таблица 1.3

Вариант 1	Вариант 2	Вариант 3	Вариант 4
{ 0XX0, X00X }	{ 1XX1, X11X }	{ X0X0, 0X0X }	{ X1X1, 1X1X }
T = { 0 1 1 0 }	T = { 1 0 0 1 }	T = { 0 1 0 1 }	T = { 1 0 1 0 }

Задание 3

Для заданной в аналитическом виде логической функции получить C^0 и C^1 путем обратной импликации

Дана схема, заданная в виде логического уравнения :

$$Y = \overline{\overline{(X_1 \vee X_2)} \,\&\, \overline{(X_2 \vee X_3)}}$$

Для построения кубического покрытия схемы на основе заданных выходных значений "0" и "1" используется π -алгоритм [1]. Для выполнения π -алгоритма строится структурно-функциональная модель схемы путем нумерации ее линий, получения кубических покрытий отдельных элементов схемы (ПЭ) и представления каждого ПЭ в терминах номеров его вхожных и выходных линий (строка 1 в таблице ниже) .

Выполнение π -алгоритма состоит из следующих пунктов, выполнение которых представлено ниже (строки 2 и 3 в таблице ниже).

1. Задается полноразрядный вектор с "0" ("1") на выходной координате (T_1).

2. Выполнение алгоритма начинается с максимального номера линии и для него выполняется пересечение формируемого вектора с кубами соответствующего покрытия. Результаты непустых пересечений записываются в стек (T_i). Номер текущей обрабатываемой линии уменьшается на 1.

3. Из стека выбирается очередной вектор (T_i) и для его обрабатываемой линии (обведена прямоугольником) для него выполняется пункт 2 .

4. Пункты 2 и 3 выполняются до тех пор, пока не будут пребраны все невходные линии (текущий номер достигнет внешних входов) .

5. Среди полученных векторов (T_i) выполняется поглощение одинаковых, а оставшиеся кубы образуют C^0 и C^1 соответственно .

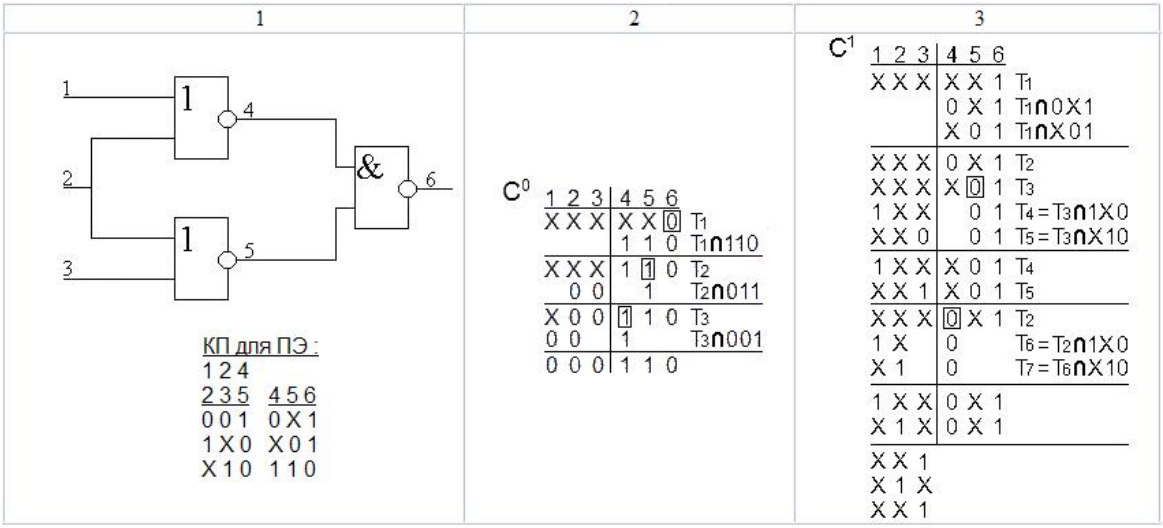


Рисунок 1.17

Таким образом в C^0 записываем T_4 из второй строки, а в C^1 { T_6 , T_7, T_5 } из третьей строки.

$$C^1 = \begin{Bmatrix} 00XX \\ XX00 \end{Bmatrix}$$

Варианты заданий для самостоятельного выполнения:

Аналитическа форма представления логической функции :

Таблица 1.4

Вариант 1	Вариант 2	Вариант 3	Вариант 4
$Y = \overline{(X_1 \& X_2)} \& \overline{(X_2 \& X_3)}$	$Y = \overline{(X_1 \vee X_2)} \vee \overline{(X_2 \vee X_3)}$	$Y = \overline{(X_1 \& X_2)} \vee \overline{(X_2 \& X_3)}$	$Y = \overline{(X_1 \vee X_2)} \vee \overline{(X_2 \& X_3)}$

Задание 4

Для заданного словесным описанием функционального элемента построить кубическое покрытие (КП)

Функционалиный элемент : Коммутатор 4 х 1 с прямым выходом.

Коммутатор 4 х 1 представляет собой функциональный элемент, имеющий четыре информационных входа D0, D1, D2, D3 и два управляющих A, B. Информация передаетс яна выход Y с того информационного входа, чей номер в двоичном эквиваленте задан значениями 0 и 1 на управляющих входах A и B .

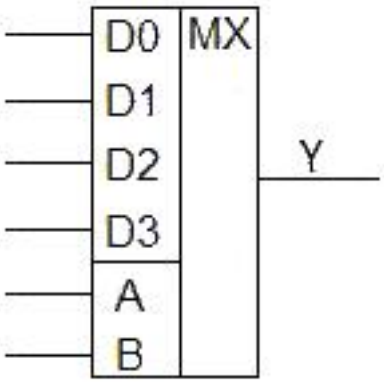


Рисунок 1.18

D0	D1	D2	D3	A	B	Y
0	X	X	X	0	0	0
1	X	X	X	0	0	1
X	0	X	X	0	1	0
X	1	X	X	0	1	1
X	X	0	X	1	0	0
X	X	1	X	1	0	1
X	X	X	0	1	1	0
X	X	X	1	1	1	1
1	1	1	1	X	X	1
0	0	0	0	X	X	0
0	0	X	X	0	X	0
1	1	X	X	0	X	1
X	X	0	0	X	1	0
X	X	1	1	X	1	1

Рисунок 1.19

Варианты заданий для самостоятельного выполнения:

Тип функционального элемента :

Таблица 1.5

Вариант 1	Коммутатор 4 в 1 с инверсными входамии прямым выходом
Вариант 2	Коммутатор 4 в 1 с инверсными входамии инверсным выходом
Вариант 3	Дешифратор 2 в 4 с прямыми выходами
Вариант 4	Дешифратор 2 в 4 с инверсными выходами

Текущий контроль знаний

Самоконтроль: Основы логического моделирования цифровых устройств

Какому трехвходовому элементу принадлежит КП {1XX0 X1X0 XX10 0001} :

- ❑ 3ИЛИ-НЕ
- ❑ 3XOR
- ❑ 3ИЛИ
- ❑ 3И-НЕ

Компонентом какой модели является примитивный элемент (ПЭ):

- ❑ структурной
- ❑ структурно-функциональный
- ❑ графовой
- ❑ аналитической

Сколько значений сигналов определено в стандарте IEEE_1164 :

- ❑ 9
- ❑ 8

- ❑ 7

Как принято называть метод моделирования, в котором задержки эл-тов не равны 0 :

- ❑ асинхронный
- ❑ синхронный

Сколько логических уровней сигналов существует в цифровых схемах :

- ❑ один
- ❑ два
- ❑ три
- ❑ бесконечное множество