

РЕАЛИЗАЦИЯ БУЛЕВОЙ МОДЕЛИ ЛОГИКИ ПЕРЕСТРАИВАЕМЫХ СТРУКТУР С ПРИМЕНЕНИЕМ ИЗОТРОПНЫХ СРЕД

3.1. ВВОДНЫЕ ПОНЯТИЯ

Создание эффективно и надежно функционирующих систем логического управления для автоматизации различного рода производственных объектов является весьма важной и актуальной задачей.

При разработке современных систем логического управления особое внимание должно уделяться структурному и функционально-логическому уровням проектирования. Значение этих уровней обусловлено формированием на них важных решений, отвечающих за качество и надежность системы, а также за содержание и направленность последующих работ на остальных уровнях проектирования [11].

В настоящее время существует множество методов проектирования систем логического управления и их классификаций по ряду признаков, основными из которых являются автоматизированные и неавтоматизированные. Последний метод весьма трудоемкий и не отвечает современным информационным технологиям.

При реализации первого метода в качестве главных рабочих принципов выходят модульность, многофункциональность и перестраиваемость будущих систем логического управления.

Использование модульности позволяет создать функционально-блочные системы, результатом которых является высокая работоспособность, сокращение времени наладки и восстановление, т.е. увеличение надежности системы.

Многофункциональность и перестраиваемость решают разнородные задачи управления. К примеру, сокращенный объем аппаратуры позволяет управлять ресурсами с различных уровней иерархии систем, что оказывает влияние на пропускную способность, живучесть и надежность, а в целом повышает тактико-технические характеристики системы.

В практике проектирования и моделирования используется множество моделей и языков, описывающих поведение дискретных систем. Применение модели конечного автомата, являющейся научной основой логического управления, ранее было затруднено при описании и оптимизации функционирования систем логического управления из-за огромных комбинационных вычислений, которые превышали возможности вычислительных средств. В настоящее время современные персональные компьютеры позволяют приблизиться к решению этой задачи.

3.2. ИМИТАЦИОННАЯ СИСТЕМА CELL SYSTEM

Для синтеза последовательных и комбинационных логических схем разработано большое количество методов минимизации функций алгебры логики [2]. Многие из этих методов принципиально предназначены для «ручной» минимизации, некоторые допускают формализацию процесса решения этой задачи и использование компьютерных технологий. При большом числе входных и выходных переменных синтезируемой схемы «ручная» минимизация описывающих ее формул становится неэффективной и во многих случаях просто невозможной, поэтому автоматизация этого процесса является актуальной задачей.

Среди признанных методов исследования сложных систем особое место занимает имитационное моделирование. Повышенное внимание к нему определяется не только возможностью анализа систем при условиях большой размерности и неполной информации о структуре системы, но и доступностью методологии для широкого круга специалистов.

В нашем случае мы рассматриваем некоторые аспекты имитационного моделирования однородных сред. Из существующих ОС выделим класс сред, которые настраиваются на реализацию того или иного дискретного автомата путем подачи на управляющие входы соответствующих настроечных кодов. Главное достоинство структур этого класса состоит в том, что синтез дискретного автомата сводится по существу лишь к нахождению настроечных кодов на основе системы булевых формул, описывающих работу автомата.

ОС могут быть реализованы аппаратными и программными способами. Относительно возможностей аппаратной реализации в [3–5] отмечено, что ОС представляют собой структуры, идеально приспособленные к особенностям техники массового производства. Аппаратная реализация необходима в тех случаях, когда требуется обеспечить мак-

симальное быстродействие автомата. Если же требования к быстродействию не доминируют, то ОС целесообразнее реализовывать многотактным способом, например с использованием компьютерных технологий.

Интерес к программному моделированию ОС неслучаен. Дело в том, что программная реализация булевых функций в «чистом» виде сопряжена со значительными трудностями даже в таких случаях, когда число аргументов не велико (10–15). Эти трудности вызваны многими причинами.

Во-первых, для хранения сложных булевых функций необходим значительный объем памяти компьютера. Например, если функцию представить в виде изображающего числа [9], то для ее машинного представления потребуется N бит памяти:

$$N = 2^s,$$

где s – число логических аргументов.

Такая реализация функций (которая является по сути СДНФ) ограничивается 10–15 аргументами, что с практической точки зрения во многих случаях совершенно недостаточно. Например, логика работы информационно-поисковых автоматов [7] даже в минимальном варианте их использования моделируется булевыми формулами 20 аргументов.

Во-вторых, время вычисления формулы быстро растет с увеличением числа аргументов.

Иное дело алгоритмы, моделирующие работу однородной среды. Если ОС состоит из s МЛМ, где каждый МЛМ имеет m входов, то для машинного представления любой неповторной булевой формулы достаточно $s \cdot m$ бит памяти. Если программная реализация МЛМ ОС осуществляется в течение времени t , то вычисление всякой неповторной булевой формулы будет выполнено за время $s \cdot t$ [15].

Отсюда следует, что имитационный метод моделирования ОС можно рассматривать как решение проблемы программного определения значений булевых формул многих аргументов.

При синтезе автомат представляется в виде системы булевых формул, каждая из которых записывается в СДНФ:

$$F_k = \bigvee_{i \in C_k} \left(\bigwedge_{j=1}^n Y_j^{a_{ij}} \right)_i \quad (k = \overline{1, p}), \quad (3.1)$$

где p – число выходов схемы; C_k – множество наборов переменных, на которых k -я формула определена и принимает истинное значение; $C_k \subseteq G$ (G – множество наборов переменных, на которых хотя бы одна из формул F_k определена и принимает истинное значение); n – число переменных формулы F_k ; a_{ij} – показатель инверсирования; $Y_j^{a_{ij}}$ – j -я переменная формулы F_k , определяемая по выражению

$$Y_j^{a_{ij}} = \begin{cases} Y_j & \text{при } a_{ij} = 1, \\ \bar{Y}_j & \text{при } a_{ij} = 0, \end{cases}$$

$$a_{ij}, Y_j^{a_{ij}}, F_k \in E, \quad E = \{0, 1\}.$$

В процессе минимизации осуществляется переход от СДНФ к сокращенной дизъюнктивной нормальной форме, а затем к их минимальной дизъюнктивно нормальной форме.

Поскольку существует некоторое множество L , образуемое набором переменных, на которых формула F_k не определена, причем эти наборы переменных не могут появиться на входах синтезируемой схемы, то они могут использоваться для образования элементарных конъюнкций минимального ранга, поглощающих максимальное количество исходных наборов переменных из множества G . Таким образом, в общем случае в выражении (3.1) $i \in H_k = C_k \cup L$ [1].

На основе полученной минимизированной булевой формулы синтезируется комбинационная схема автомата. Данная концепция заложена в основу созданной автором логической системы имитационного моделирования Cell System, представляющей собой расширение динамической библиотеки Simulink интегрированной системы автоматизации математических и научно-технических расчетов Matlab (рис. 3.1).

Cell System включает в себя МЛМ – (S -, T -, H -, L -, V -ячеек).

Каждый из включенных в библиотеку МЛМ ориентирован на вычисление определенного класса булевых формул. Вышеупомянутые модули разрабатывались в перечисленной очередности, и для каждого модуля предыдущий является прототипом, для первой указанной ячейки (S -) прототипом является ячейка, описанная в работе [10]. Разработка каждого нового модуля связана с необходимостью вычисления последующих классов булевых формул, не поддающихся вычислению с по-

мощью уже разработанных. Причем каждая вновь разработанная ячейка наделяется свойством преемственности, т.е. последняя ячейка (V -) охватывает самую обширную часть классификации, приведенной в работе [14], и является наиболее универсальной по сравнению со всеми другими указанными модулями.

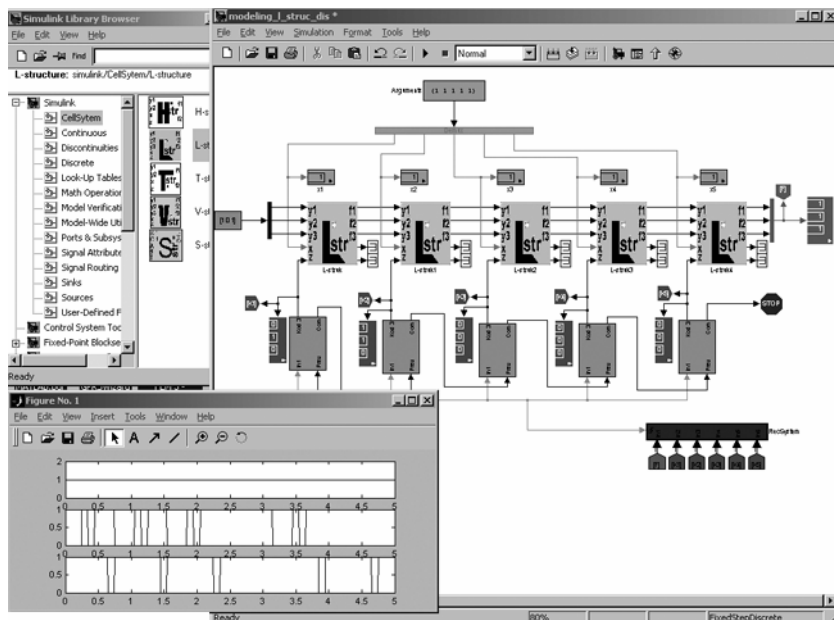


Рис. 3.1 . Система имитационного моделирования

Полученная динамическая библиотека МЛМ позволяет существенно облегчить работу при исследовании и построении на их основе одно-родных сред и самих МЛМ.

Представление моделей в такой форме очень хорошо согласуется с теоретическими выкладками и является отличной основой для более детальной проработки исследовательского материала. Однако когда возникает задача реализации того или иного устройства в промышленности с использованием микропроцессорных средств, появляются трудности с переводом модели, представленной в структурной форме. Для наиболее быстрого перевода модели в микропроцессорные средства

хотелось бы работать с некоей функцией, отражающей поведение устройства. Аргументы функции являются входом устройства, а принимаемые ею значения – его выходом [12].

Кроме структурной реализации, все рассмотренные выше МЛМ реализованы в среде MatLab в качестве функций. Для этого использовались системы булевых формул, описывающие каждую из МЛМ. Данный метод хорошо себя показал и при исследовании, требующем обрабатывать большое количество информации. Например, при определении функциональной полноты V -структуры необходимо было рассмотреть 256 различных комбинаций; с помощью Cell System на компьютере Pentium Celeron 700 MHz/RAM 256 MB это занимает 0,485 с, включая генерацию восьмиразрядного кода. В результате получается матрица данных размером 256×11 , отражающая все возможные состояния МЛМ V -структуры. Для дальнейшей обработки этой информации создан блок анализа, выполняющий проверку МЛМ на полноту.

Одним из важных достоинств Cell System является ее интеграция со средой MatLab, в результате которой появляется возможность использовать в исследованиях наработки ученых всего мира. После исследования свойств вновь разработанного модуля и проверки его работоспособности следует рассмотреть вопрос о его практическом применении. Подобные модули могут широко использоваться в области скоростной обработки информации, а также в перспективном направлении – в системах автоматического регулирования. Построенные на основе МЛМ устройства управления наделяют всю систему автоматического регулирования новыми динамическими свойствами, такими как инвариантность к параметрическим возмущениям и повышение быстродействия, не присущие ей при использовании традиционных линейных законов регулирования. Исследование полученной системы аналитическими методами становится неэффективным, а иногда и невозможным. Так, единственным способом исследования на устойчивость систем автоматического регулирования более чем с одной нелинейностью (например, реальные физические системы), пожалуй, является имитационное моделирование.

3.3. ИМИТАЦИОННОЕ МОДЕЛИРОВАНИЕ ИЗОТРОПНЫХ СРЕД

Рассмотрим однородную среду, представляющую собой линейную структуру, состоящую из пяти МЛМ (рис. 3.2). Каждая МЛМ описывается системой булевых формул вида (2.17).

Используя интерпретирующий метод [6, 8] для структуры, где каждая МЛМ имеет $m = 4$ входа, можно подсчитать объем оперативной памяти, необходимой для запоминания любой бесповторной булевой формулы: он будет равен $4s$ бит, где s – число аргументов. Это значит, что имеется реальная возможность реализации булевых формул, зависящих от сотен аргументов.

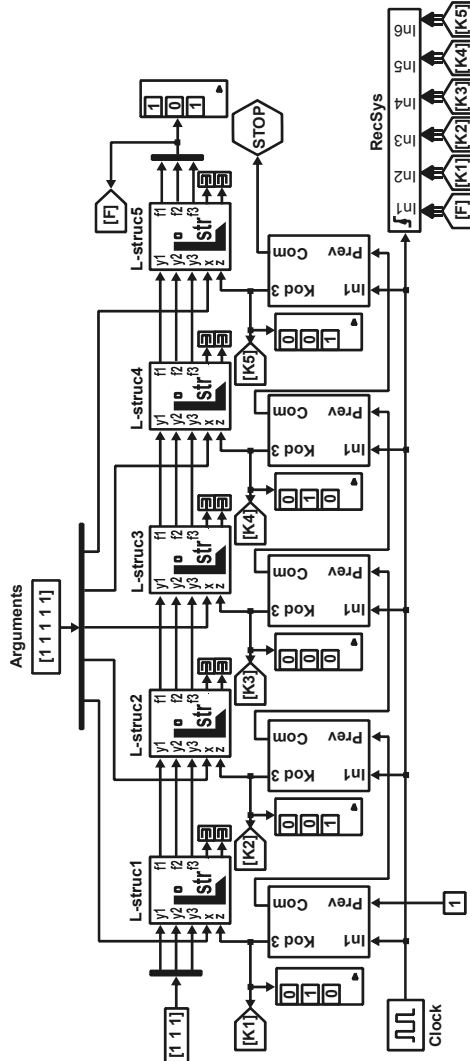


Рис. 3.2. Реализация однородной линейной среды из МЛМ L-структуры

Рассматриваемая ОС состоит из пяти МЛМ, описываемых системой (2.17), имеет три режимных входа (y_1 , y_2 и y_3). Следовательно, возможны восемь режимов его работы. Все они были экспериментально проверены в процессе имитационного моделирования. В левой части табл. 3.1 приведены все возможные

настроечные коды, правая часть разделена на восемь областей, каждой области соответствует определенный режим работы ОС. Режимы закодированы в тех же обозначениях, что и настроечные коды. Первый слева символ кода режима соответствует входу y_1 , второй – y_2 и третий – y_3 .

Т а б л и ц а 3.1

**Таблица истинности функций f_1, f_2, f_3 ОС
при подаче на все МЛМ настроечного кода z_1, z_2, z_3**

Настроеч- ные коды z_1, z_2, z_3	000	001	010	011	100	101	110	111
	f_1, f_2, f_3	f_1, f_2, f_3	f_1, f_2, f_3	f_1, f_2, f_3	f_1, f_2, f_3	f_1, f_2, f_3	f_1, f_2, f_3	f_1, f_2, f_3
000	000	001	010	011	100	101	110	111
001	000	001	010	011	100	101	110	111
010	100	101	110	111	100	101	110	111
011	011	011	011	011	111	111	111	111
100	100	101	100	101	100	101	110	111
101	010	011	010	011	110	111	110	111
110	110	111	110	111	110	111	110	111
111	010	010	010	011	110	110	110	111

После проверки работоспособности имитационной модели МЛМ была исследована работа всей ОС на множестве контрольных примеров.

Дальнейшее более тщательное исследование работы имитационной модели ОС выявило некоторые особенности ее функционирования, которые на этапе логического анализа никак не проявлялись. Суть их в следующем.

Первый случай. Если на режимные входы первого МЛМ подать 0, т.е. принять $y_1 = y_2 = y_3 = 0$, и каждый из пяти МЛМ настроить на код 011, а на информационные входы подать единичные значения, то на выходе последнего МЛМ получим (см. табл. 3.1):

$$f_1 = 0, f_2 = f_3 = 1.$$

Хотя, как следует из формулы (2.17), надо было ожидать $f_1 = f_3 = 0$; $f_2 = 1$, поскольку все МЛМ построены совершенно одинаково. Это объясняется тем, что режимный вход второго МЛМ при указанных условиях изменяется и устанавливается равным 010.

Второй случай обнаружен при этом же настроечном коде (011), поданном на все МЛМ, единичных значениях всех логических аргументов и режиме 100 первого МЛМ. Установлено, что $f_1 = f_2 = f_3 = 1$, хотя по формуле (2.17) следовало ожидать $f_1 = f_2 = 1$; $f_3 = 0$, поскольку, как и в первом случае, все МЛМ настроены совершенно одинаково.

Третий случай обнаружен при настроечном коде 110, поданном на все МЛМ, единичных значениях всех логических аргументов и режиме 001 первого МЛМ. Было установлено, что $f_1 = f_2 = f_3 = 1$, хотя по формуле (2.17) следовало ожидать $f_1 = f_3 = 1$; $f_2 = 0$.

На следующем этапе имитационного моделирования ОС различные ячейки настраивались на различные настроечные коды. Поскольку всего существует 2^{15} различных вариантов настройки ОС, составленной из пяти МЛМ, то методом случайной выборки было определено несколько вариантов настройки всей ОС. Результаты вычисления десяти функций для единичных значений логических аргументов представлены в табл. 3.2.

Т а б л и ц а 3.2

Таблица истинности функций f_1, f_2, f_3 при случайной выборке для каждой МЛМ линейной ОС

Настроечные коды					000	001	010	011	100	101	110	111
z_1, z_2, z_3					f_1, f_2, f_3	f_1, f_2, f_3	f_1, f_2, f_3	f_1, f_2, f_3	f_1, f_2, f_3	f_1, f_2, f_3	f_1, f_2, f_3	f_1, f_2, f_3
1	2	3	4	5								
000	111	001	010	110	110	110	110	111	110	110	110	111
000	000	110	110	001	110	111	110	111	110	111	110	111
101	110	001	010	001	110	111	110	111	110	111	110	111
000	000	001	100	011	110	111	110	111	110	111	111	111
011	101	101	001	000	010	011	011	011	110	111	111	111
001	001	110	110	000	110	111	110	111	110	111	110	111
111	011	111	111	111	011	011	011	011	111	111	111	111
001	010	110	110	111	110	111	110	111	110	111	110	111
101	110	000	000	101	110	111	110	111	110	111	110	111
011	100	011	101	110	110	111	111	111	111	111	111	111

Таким образом, если режимные и настроечные коды каждого МЛМ рассматривать как обобщенный шестиразрядный код, то:

а) первый МЛМ может быть настроен на любой из 64 возможных обобщенных кодов;

б) для всех остальных МЛМ существуют обобщенные коды, которые не могут быть поданы на входы этих МЛМ.

Такой интересный результат оказалось возможным получить лишь в результате имитационного моделирования ОС [14].

Т е о р е м а 3.1. Если при настройке МЛМ ее выходы f_i тождественно равны информационным входам y , то построенная линейная ОС, состоящая из n таких модулей, соответствует одному при такой же настройке.

Д о к а з а т е л ь с т в о. Пусть линейная ОС состоит из n ячеек, настроенных таким образом, что выполняется равенство

$$f_{i,n} = y_{i,n}, \quad (3.2)$$

где $i = 1, 2, 3$ – соответствующий номер входа и выхода ячейки.

Исходя из того, что структура является линейной, т.е. каждый вход последующей ячейки равен выходу предыдущей

$$y_{i,n} = f_{i,n-1},$$

учитывая (3.2), можно записать

$$f_{i,n} = y_{i,n} = f_{i,n-1}, \text{ или } f_{i,n} = f_{i,n-1} = \dots = f_{i,1} = y_{i,1}.$$

Что и требовалось доказать.

Т е о р е м а 3.2. Для каждого единичного множества $A_{i+\bar{x}}$ ($i = 1, 3, \dots, 2^n - 1$), состоящего из значения кода F на выходе МЛМ, при условии, что

$$F = Y, \quad (3.3)$$

где Y – код на входе; n – число выходов ячейки; x – значение аргумента и связанного с ним множества настроечных кодов $B_{i+\bar{x}}$, существует такое множество $A_{i+1-\bar{x}}$, при котором $B_{i+\bar{x}} = B_{i+1-\bar{x}}$, кроме случая, когда множеству $A_{i+1-\bar{x}}$ принадлежит элемент, к старшему разряду α которого два раза справа примыкает значение аргумента x ($\alpha x x \in A_{i+1-\bar{x}}$), тогда $B_{i+1-\bar{x}} = B_{i+\bar{x}} \cup \{\bar{x}11\}$.

Доказательство. Обозначим каждое значение из 3-разрядного кода соответствующей буквой латинского алфавита в порядке возрастания, всего получится 2^3 букв.

По формуле, описывающей L -структуру (2.17), найдем коды настройки, при которых выполняется условие (3.3).

Пусть значение аргумента $x = 1$:

$$A_1 = \{a\}, B_1 = \{a, b\}; A_2 = \{b\}, B_2 = \{a, b\};$$

$$A_3 = \{c\}, B_3 = \{a, b, f, h\}; A_4 = \{d\}, B_4 = \{a, b, d, f, h\};$$

$$A_5 = \{e\}, B_5 = \{a, b, c, e\}; A_6 = \{f\}, B_6 = \{a, b, c, e\};$$

$$A_7 = \{g\}, B_7 = \{a, b, c, e, f, g, h\}; A_8 = \{h\}, B_8 = \{a, b, c, d, e, f, g, h\}.$$

Отметим, что $B_1 = B_2$, $B_5 = B_6$, а симметрическая разность множеств

$$B_3 \oplus B_4 = B_3 \cap \bar{B}_4 \cup \bar{B}_3 \cap B_4 = \{d\} \text{ и } B_7 \oplus B_8 = \{d\}$$

дает нам один результат – элемент $d = 011$ или $d = \bar{x}11$.

Аналогично при $x = 0$:

$$A_1 = \{a\}, B_1 = \{a, b, c, d, e, f, g, h\}; A_2 = \{b\}, B_2 = \{a, b, c, d, e, f, g\};$$

$$A_3 = \{c\}, B_3 = \{a, c, f, g\}; A_4 = \{d\}, B_4 = \{a, c, f, g\};$$

$$A_5 = \{e\}, B_5 = \{b, c, d, f, h\}; A_6 = \{f\}, B_6 = \{b, c, d, f\};$$

$$A_7 = \{g\}, B_7 = \{c, f\}; A_8 = \{h\}, B_8 = \{c, f\}.$$

Следовательно, $B_3 = B_4$; $B_7 = B_8$.

Симметрическая разность даст нам $B_1 \oplus B_2 = B_5 \oplus B_6 = \{h\}$.

Таким образом, при $x = 0$ существует равенство между множествами B_3 и B_4 , а также между B_7 и B_8 , множества же B_1 и B_2 , как, в свою очередь, B_5 и B_6 , отличаются друг от друга на элемент $h = 111$ или $h = \bar{x}11$ при принадлежности единичному множеству A_1 элемента $a = 000 = \alpha x x$, или множеству A_5 элемента $e = 100 = \alpha x x$, где α – дво-

ичное число, соответствующее старшему разряду кода на выходе МЛМ. Что и требовалось доказать.

С л е д с т в и е из теорем 3.1 и 3.2.

Если в линейной ОС есть ячейки с настроечными кодами из набора B_i и кодом на входе A_i , то считается, что они удалены из структуры (т.е. осуществляется операция пропуска аргумента).

Таким образом, имитационное моделирование изотропных структур позволило провести проверку работоспособности этих структур, выявить ряд интересных результатов настройки структур обобщенными кодами, а также исключить все ошибки, которые могли возникнуть при объединении отдельных ячеек изотропной структуры.

При моделировании изотропных сред интерпретирующий метод можно использовать как для решения проблемы программного нахождения значений булевых формул практически неограниченного числа аргументов, так и для анализа этих сред.

3.4. СИНТЕЗ ЛИНЕЙНЫХ ИЗОТРОПНЫХ СРЕД

Рассмотрим четыре случая построения однородных сред.

1. Боковые входы ОС задействованы для подачи значений аргументов и в процессе функционирования могут принимать значения из множества $\{0, 1\}$.

2. Осуществляется поворот с одного из выходов ОС на один из ее входов, при этом на боковые входы также подаются значения аргументов.

3. Фиксированный код 111 подается на боковые входы ОС и в процессе функционирования не изменяется (вариант режимных входов).

4. Осуществляется поворот с одного из выходов ОС на один из ее входов, при этом на боковые входы подаются константы (режимные входы).

С л у ч а й 1 для L -с т р у к т у р ы. Зададимся системой булевых формул вида

$$\begin{cases} f_1 = x_6 \vee x_7, \\ f_2 = x_4 x_5, \\ f_3 = x_1 \vee x_2 x_3. \end{cases} \quad (3.4)$$

Соответствующая ей однородная среда, построенная на базе L -структуры с настройочными кодами, представлена на рис. 3.3.

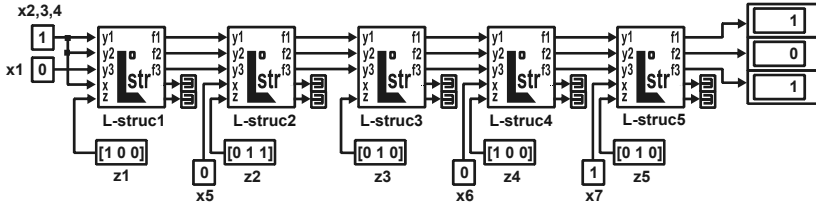


Рис. 3.3. Однородная среда на МЛМ L -структуры, реализующая систему (3.4)

В данном случае на функциональный вход x третьей ячейки подается нуль, чтобы в дальнейшем подавать в качестве аргумента выходные значения формул с пятой ячейки ОС, т.е. осуществлять поворот.

С л у ч а й 2 д л я L -с т р у к т у р ы. Будем поочередно производить поворот с каждого выхода ОС (f_1, f_2, f_3) на вход третьей ячейки. На рис. 3.4 отражена структура ОС при повороте первого выхода f_1 .

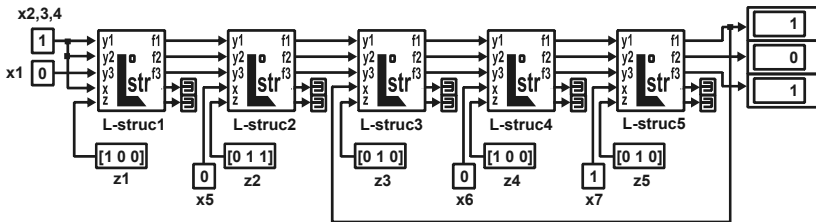


Рис. 3.4. Однородная среда в случае 2 для L -структуры

Система булевых формул в этом случае принимает вид

$$\begin{cases} f_1 = x_6 \vee x_7, \\ f_2 = (x_4 \vee x_6 \vee x_7)x_5, \\ f_3 = x_1 \vee x_2x_3. \end{cases}$$

В случае введения поворота со второго выхода пятой ячейки f_2 ОС система булевых формул, описывающая поведение всей ОС, преобразуется к форме

$$\begin{cases} f_1 = x_6 \vee x_7, \\ f_2^{t+1} = [(x_4 \vee f_2)x_5]^t, \\ f_3 = x_1 \vee x_2x_3. \end{cases}$$

Здесь и в дальнейшем индекс t , расположенный над скобкой, указывает, что значения переменных, находящихся в скобках, относятся к моменту времени t .

И, наконец, в случае поворота третьего выхода f_3 система булевых формул, описывающая ОС, примет вид

$$\begin{cases} f_1 = x_6 \vee x_7, \\ f_2 = (x_1 \vee x_2x_3 \vee x_4)x_5, \\ f_3 = x_1 \vee x_2x_3. \end{cases}$$

Реализация системы булевых формул (3.4) в однородной линейной среде из V -структур оказалась затруднительной, однако применение принципов квазиоднородных сред позволило решить эту задачу (рис. 3.5).

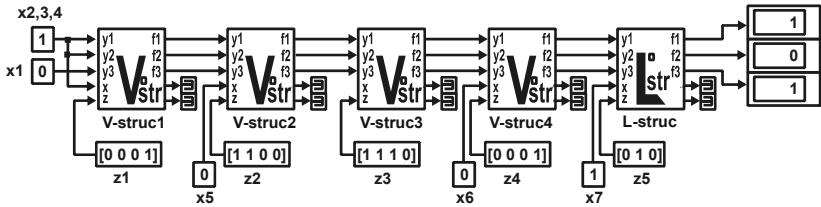


Рис. 3.5. Квазиоднородная среда, реализующая систему (3.4)

Случай 1 для V -структуры. Рассмотрим реализацию систем булевых формул вида

$$\begin{cases} f_1 = 0, \\ f_2 = (x_3 \vee x_4x_5)x_7, \\ f_3 = x_1x_2 \vee x_5 \vee x_6. \end{cases} \quad (3.5)$$

Соответствующая ей однородная среда, построенная на базе V -структуры с настроенными кодами, представлена на рис. 3.6.

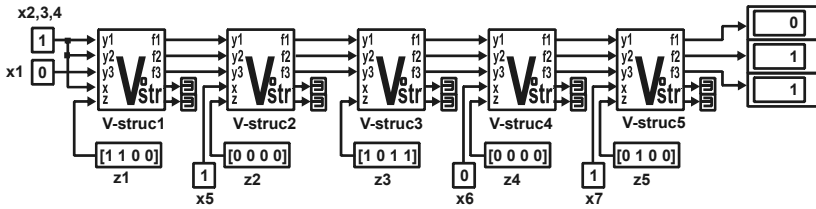


Рис. 3.6. Однородная среда на МЛМ V -структуры, реализующая систему (3.5)

Случай 2 для V -структуры. Аналогично рассмотренному выше примеру будем последовательно заводить выходы ОС (f_1 , f_2 , f_3) на функциональный вход x третьей ячейки. Система формул при повороте с первого выхода f_1 примет вид

$$\begin{cases} f_1^{t+1} = 0, \\ f_2 = [(x_3 \vee x_4 x_5 \vee f_1) x_7]^t, \\ f_3 = x_1 x_2 \vee x_5 \vee x_6. \end{cases}$$

В случае введения поворота со второго выхода f_2 система формул преобразуется к виду

$$\begin{cases} f_1 = f_2 x_6, \\ f_2^{t+1} = [(x_3 \vee x_4 x_5 \vee f_2) x_7]^t, \\ f_3 = x_1 x_2 \vee x_5 \vee x_6. \end{cases}$$

И, наконец, в случае поворота третьего выхода f_3 система булевых формул, описывающая ОС, примет вид

$$\begin{cases} f_1 = (x_1 x_2 \vee x_5 \vee x_6) x_6, \\ f_2 = (x_1 x_2 \vee x_3 \vee x_4 x_5 \vee x_5 \vee x_6) x_7, \\ f_3 = x_1 x_2 \vee x_5 \vee x_6. \end{cases}$$

Случай 3 для V -структуры. Рассмотрим реализацию булевой формулы вида

$$f = (x_1 \vee x_2) x_2 x_3 \vee x_4 \tag{3.6}$$

с помощью ОС, построенной на V -структурах (рис. 3.7). В данном случае осуществлять поворот будем только с первого выхода f_1 , а реализацию формулы (3.6) будем производить на втором выходе f_2 .

При настроечном коде 0110 на третьей ячейке происходит операция пропуска аргумента (ячейка принимает структуру, приведенную на рис. 2.20, e), поэтому как такового поворота в ОС нет, т.е. если проводить аналогию с теорией автоматического управления, то цепь обратной связи в этом состоянии разомкнута.

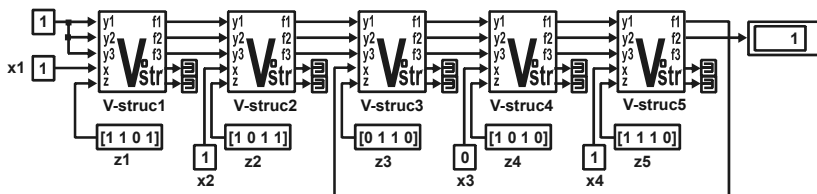


Рис. 3.7. Однородная среда к случаю 3 для V -структуры

С л у ч а й 4 для V -с т р у к т у р ы. Выясним, как влияет введение поворота на реализацию булевой формулы. Для этого изменим настроечный код третьей ячейки (рис. 3.7) на 1110. В итоге реализуемая средой булева формула примет вид

$$f = (x_1 \vee x_2 \vee x_3)x_2x_3 \vee x_4.$$

С л у ч а й 3 для L -с т р у к т у р ы. Рассмотрим аналогичную ситуацию для ОС, построенной на L -структурах. Реализуем формулу вида

$$f = x_1 \vee x_2 \vee x_4.$$

Соответствующая ей однородная среда с настроечными кодами представлена на рис. 3.8.

Аналогично вышерассмотренному примеру на третьей ячейке осуществляется операция пропуска аргумента; в этом случае потребовались боковые, режимные входы ОС. Данный аспект связан с тем, что L -ячейка не имеет явной структуры, выполняющей пропуск аргумента, поэтому при необходимости эту операцию нужно вводить искусственно. Таким образом, при настроечном коде 101 третья ячейка ОС не осуществляет поворот.

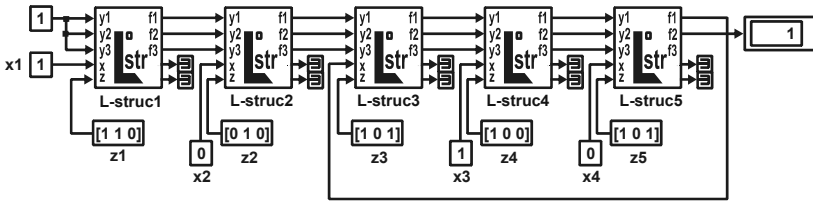


Рис. 3.8. Однородная среда к случаю 3 для L -структуры

С л у ч а й 4 д л я L -с т р у к т у р ы. При изменении настроечного кода третьей ячейки на 001, т.е. при осуществлении операции «разрешения» поворота, реализуемая формула примет вид

$$f = (x_1 \vee x_2)x_3 \vee x_4.$$

Таким образом, в ОС, реализуемых на L - или V -структурах, можно осуществлять операции «разрешение» или «запрет» поворота, не изменяя комбинационных связей между ячейками, а изменяя только настроечный код той, на которую заведен поворот. Следует также отметить, что операция «поворот» требует ввода дополнительной ячейки в структуру ОС. Введение же поворота в канал, с которого он осуществляется (замкнутый цикл), позволяет строить многотактные автоматы на основе МЛМ L - и V -структур.

3.5. ДЕКОМПОЗИЦИЯ ЛИНЕЙНЫХ ИЗОТРОПНЫХ СРЕД

Задачей декомпозиции линейных изотропных сред является исследование и изучение их реализаций при помощи МЛМ, соединенных между собой.

Пусть M_1, M_2, \dots, M_n – совокупность МЛМ, образуемых при декомпозиции ОС. Обозначим через F_i, G_i соответственно множества выходных и входных переменных ячейки M_i . Для каждой ячейки M_i множество G_i определяется векторным произведением подмножеств Y_i, X_i и Z_i .

Y_i обозначает множество входных переменных ячейки M_i , являющихся выходными переменными других ячеек, X_i – множество внешних входных аргументов, приложенных непосредственно к ячейке

M_i , а Z_i – множество внешних входных переменных, отражающих настройку внутренней структуры ячейки M_i .

Приведем несколько определений.

О п р е д е л е н и е 1. Будем считать, что взаимное соединение ячеек M_1, M_2, \dots, M_n действует одновременно, если выходная переменная F_i каждой M_i -й ячейки ($i=1, 2, \dots, n$) зависит от состояний входов Y_i , определяемых текущим структурным состоянием других ячеек, к которым подсоединена M_i , и от состояния внешних входов X_i и определяющих внутреннюю структуру входов Z_i .

Таким образом, упорядоченное множество из n элементов, образованных n состояниями ячеек, является состоянием всей ОС.

О п р е д е л е н и е 2. Подмножество взаимно соединенных МЛМ $M_{i1}, M_{i2}, \dots, M_{ip}$ образуют петлю, если выход модуля M_{ik} является входом модуля $M_{i(k+1)}$ ($k=1, 2, \dots, p-1$) и если выход модуля M_{ip} является входом модуля M_{i1} .

О п р е д е л е н и е 3. Ячейка M_i называется предшествующей ячейке M_j , если одним из входов ячейки M_j является выход ячейки M_i . Подмножество ячеек $M_{i1}, M_{i2}, \dots, M_{ip}$, соединенных между собой, называют замкнутым, если все ячейки этого подмножества являются предшествующими. Для заданной ОС наименьшее замкнутое подмножество, содержащее M_i , будем обозначать $C(M_i)$.

О п р е д е л е н и е 4. Будем считать, что линейная ОС реализуется при помощи множества ячеек M_1, M_2, \dots, M_n , действующих одновременно и соединенных без обратных связей в том случае, если выполняются следующие условия:

1) ячейки M_1, M_2, \dots, M_n действуют одновременно и соединены без обратных связей (определения 1 и 2);

2) множество входов M является подмножеством множества возможных входов ОС, реализованного при помощи M_1, M_2, \dots, M_n ;

3) существует взаимно однозначное соответствие между множеством структурных состояний M и подмножеством декартова произведения множеств структурных состояний ячеек M_1, M_2, \dots, M_n .

Таким образом, если ОС и ячейки M_1, M_2, \dots, M_n находятся в одном из исходных структурных состояний, то после подачи некоторой входной последовательности X они окажутся в соответствующих конечных структурных состояниях.

Основная теорема декомпозиции

Теорема 3.3. Пусть M – линейная ОС, а M_1, M_2, \dots, M_n – множество ячеек, работающих одновременно, соединенных без обратных связей и выполняющих те же переходы между состояниями, что и M . При таких условиях каждому замкнутому подмножеству $C(M_i)$ этих ячеек соответствует подстановочное разбиение π_i . Множество этих разбиений обладает свойством

$$\prod_{i=1}^n \pi_i = 0.$$

Кроме того, из условия $C(M_i) \leq C(M_j)$ следует, что $\pi_i \geq \pi_j$, и наоборот, каждому множеству подстановочных разбиений $T = (\pi_1, \pi_2, \dots, \pi_n)$ ОС M при $\prod_{i=1}^n \pi_i = 0$ соответствует реализация этой ОС при помощи n соединенных между собой без обратных связей ячеек, действующих одновременно. При такой реализации выход ячейки M_i может быть входом ячейки M_j только в том случае, если $\pi_i > \pi_j$. В разбиении π_i каждый из блоков содержит все те состояния ОС M , которым соответствует один и тот же набор состояний ячеек, образующих подмножество $C(M_i)$.

Доказательство:

Первая часть. Пусть ОС M состоит из ячеек M_1, M_2, M_3 , каждая из которых принимает по две структуры, определяемые настроечными кодами Z_1, Z_2 и Z_3 соответственно.

Следовательно, каждый настроечный код может принимать значения из двух наборов, например $Z_i = \{z_{i1}^1 z_{i2}^1 z_{i3}^1 z_{i4}^1, z_{i1}^2 z_{i2}^2 z_{i3}^2 z_{i4}^2\}$, тогда для

$i = 1, 2, 3$ существует множество $S = \{s_1, s_2, \dots, s_8\}$, состоящее из $2^3 = 8$ элементов, отражающих состояние ОС M . Элементы множества S получают путем обобщения настроечных кодов каждой ячейки. Замкнутое множество $C(M_2)$ образуется ячейками M_1 и M_2 . Ячейка M_1 определяет переменную F_1 и имеет структуры v_1 и v_2 ; v_1 соответствует настроечным кодам s_1, s_2, s_7 и s_8 ОС M , v_2 – настроечным кодам s_3, s_4, s_5 и s_6 . Ячейка M_2 определяет переменную F_2 , обладает структурами v_3 и v_4 с настроечными кодами s_2, s_4, s_5, s_7 и s_1, s_3, s_6, s_8 соответственно.

Таким образом, в множестве $C(M_2)$ состояния ОС M представляются следующим образом:

$$\begin{array}{llll} s_1 \rightarrow v_1, v_4 & s_3 \rightarrow v_2, v_4 & s_5 \rightarrow v_2, v_3 & s_7 \rightarrow v_1, v_3 \\ s_2 \rightarrow v_1, v_3 & s_4 \rightarrow v_2, v_3 & s_6 \rightarrow v_2, v_4 & s_8 \rightarrow v_1, v_4. \end{array}$$

Разбиение, соответствующее $C(M_2)$ [$\pi = s_1 s_8, s_2 s_7, s_3 s_6, s_4 s_5$] = π_1], содержит в одном блоке структурные состояния M , которые имеют одно и то же представление в этом замкнутом множестве.

Покажем теперь, что в общем случае разбиение π_i , соответствующее $C(M_i)$, является подстановочным. Для этого заметим, что по определению замкнутого подмножества ячеек (определение 3) структурное состояние ячеек, входящих в $C(M_i)$, может быть определено по внешнему (функциональному) входному и настроечному входу ячеек из $C(M_i)$. Таким образом, если два настроечных кода из множества S ОС M , отражающих структуру v_1 и v_2 , содержатся в одном блоке разбиения π_i , то для каждой возможной входной переменной X_k информация о выходных переменных будет также содержаться в том же блоке π_i , т.е. этот блок определяется входной переменной X_k и настроечными кодами тех ячеек, которые содержатся в $C(M_i)$. Следовательно, разбиение π_i является подстановочным.

Кроме того, если $C(M_j)$ содержит по меньшей мере все ячейки из $C(M_i)$ [$C(M_j) \geq C(M_i)$], то каждый блок из π_j будет содержаться в одном из блоков разбиения π_i и, следовательно, $\pi_i \geq \pi_j$. В рассматриваемом случае подмножеству $C(M_1)$, т.е. просто M_1 , соответствует разбиение $\pi_3 = (s_1s_2s_7s_8, s_3s_4s_5s_6)$, а замкнутому подмножеству $C(M_2)$, содержащему M_1 и M_2 , соответствует разбиение $\pi_1 = (s_1s_8, s_2s_7, s_3s_6, s_4s_5)$. Таким образом, $C(M_2) \geq C(M_1)$ при $(s_1s_8, s_2s_7, s_3s_6, s_4s_5) \leq (s_1s_2s_7s_8, s_3s_4s_5s_6)$. Что и требовалось доказать.

В т о р а я ч а с т ь. Пусть число разбиений n равно числу ячеек реализуемой декомпозиции ОС и ячейка M_j может получать информацию от M_i , т.е. быть соединенной с M_i только тогда, когда $\pi_j \leq \pi_i$.

Доказательство осуществляется в пять этапов.

1. Прежде рассмотрим все те разбиения множества T , которые не меньше любого другого разбиения этого множества. Пусть T_1 – множество таких разбиений. Каждое разбиение из T_1 определяет ячейку, и каждая из этих ячеек не будет получать информацию от других ячеек, т.е. будет иметь только внешние входы. Структура ячеек определяется блоками соответствующих разбиений. Следовательно, каждая ячейка будет иметь столько структур, сколько блоков содержится в соответствующем разбиении. К этому типу относятся ячейка M_1 и разбиение π_3 из доказательства первой части теоремы.

2. Если $T_1 = T$, то ОС M может быть реализована при помощи n ячеек, работающих параллельно. В этом случае число структур M_i равно числу блоков в разбиении π_i .

3. Если $T_1 \neq T$, рассмотрим разбиения множества $T - T_1$. Выберем в этом новом множестве те разбиения, которые не меньше любого другого разбиения из этого множества. Пусть такие разбиения образуют множество T_2 .

4. Для каждого разбиения π_j из множества T_2 рассмотрим все те разбиения π_i из множества T_1 , которые больше π_j . Определим пересечение $\pi_j^* = \prod \pi_i$ при $\pi_i \in T_1 > \pi_j \in T_2$.

Обозначим через $l_j = [\pi_j / \pi_j^*]$ наибольшее число блоков π_j , содержащихся в одном блоке π_j^* . Для выбора МЛМ M_j , соответствующей разбиению π_j , будем учитывать, что существует разбиение τ_j такое, что $\tau_j \geq \pi_j$ (τ_j содержит l_j блоков) и $\tau_j \cdot \pi_j^* = \pi_j$. Наконец, поскольку π_j является подстановочным разбиением и $\tau_j \geq \pi_j$, (π_j, τ_j) являются парой разбиений.

Таким образом, можно выбрать МЛМ M_j , имеющую l_j структур, соответствующих l_j блокам разбиения τ_j . Эта ячейка получает информацию только от ячеек, соответствующих тем разбиениям из множества T_1 , которые больше π_j .

5. Если $T \neq T_1 + T_2$, то способом, аналогичным предыдущему, строят следующий каскад декомпозиции до тех пор, пока не будут рассмотрены все разбиения T . Так, будут рассмотрены разбиения из множества $T - (T_1 + T_2)$, которые не меньше любого другого разбиения из этого нового множества. Они образуют множество T_3 , исходя из которого можно определить соответствующие ячейки, получающие информацию от ячеек множества T_2 и т.д. Что и требовалось доказать.

Таким образом, предложено решение задачи декомпозиции линейных изотропных сред.

3.6. ВЫВОДЫ

Отметим основные результаты, полученные в данной главе:

1. Впервые создана логическая система имитационного моделирования Cell System для рассматриваемого класса многофункциональных логических модулей. Таким образом, у разработчика появляется мощ-

ный инструментарий, позволяющий синтезировать и исследовать МЛИМ, построенные на их основе изотропные и квазиизотропные среды и их взаимодействие с системами, имеющими практическую направленность в той или иной области науки и техники.

2. Установлено, что при моделировании изотропных сред интерпретирующий метод можно использовать как для решения проблемы программного нахождения значений булевых функций практически неограниченного числа аргументов, так и для анализа этих сред.

3. Впервые решены задачи реализации операций «разрешение» и «запрет» поворота без изменения комбинационной связи между ячейками.

4. Предложено решение задачи декомпозиции линейных изотропных сред.

ЛИТЕРАТУРА

1. Баранов С.И., Скляр В.А. Цифровые устройства на программируемых БИС с матричной структурой. – М.: Радио и связь, 1986. – 272 с.
2. Гаврилов М.А., Девятков В.В., Пупырев Е.И. Логическое проектирование дискретных автоматов. – М.: Наука, 1997. – 352 с.
3. Евреинов Э.В. Однородные вычислительные системы, структуры и среды. – М.: Радио и связь, 1981. – 208 с.
4. Каляев А.В. Многопроцессорные системы с программируемой архитектурой. – М.: Радио и связь, 1984. – 240 с.
5. Кудрявцев В.Б., Алешин С.В., Подколзин А.С. Введение в теорию автоматов. – М.: Наука, 1985. – 320 с.
6. Лазарев В.Г., Пийль Е.И., Турута Е.Н. Построение программируемых управляющих устройств. – М.: Энергоатомиздат, 1984. – 192 с.
7. Однородные управляющие структуры адаптивных роботов / А.В. Каляев и др. – М.: Наука, 1990. – 152 с.
8. Пупырев Е.И. Перестраиваемые автоматы и микропроцессорные системы. – М.: Наука, 1984. – 192 с.
9. Фет Я.И. Параллельные процессоры для управляющих систем. – М.: Энергоатомиздат, 1981. – 160 с.
10. Шальто А.А. Многофункциональный логический модуль. А.с. № 798802 (СССР) // Б.И. – 1981. – № 3.
11. Шальто А.А. Логическое управление. Методы аппаратной и программной реализации алгоритмов. – СПб.: Наука, 2000. – 780 с.

12. *Шидловский С.В.* Автоматизация синтеза перестраиваемых структур // Известия Томского политехнического университета. – 2006. – № 1. – С. 190–192.

13. *Шидловский С.В.* Имитационное моделирование однородной среды // Докл. Томского государственного университета систем управления и радиоэлектроники. Т. 8: Автоматизированные системы обработки информации, управления и проектирования: Сб. научных трудов. – Томск: ТУСУР, 2003. – С. 99–103.

14. *Шидловский С.В.* Ячейка однородной среды. Патент РФ на изобретение № 2251141 // Б.И. – 2005. – № 12. – 13 с.

15. *Шидловский С.В., Светлаков А.А.* Применение информационных технологий при структурном моделировании элементов и устройств систем автоматического управления // Материалы региональной науч.-методич. конф. «Современное образование: Системы и практика обеспечения качества». – Томск: ТУСУР, 2002. – С. 79–80.