Список литературы: 1.*Бережная М.А., Лобода В.Г., Цуканов В.Ю.* К вопросу проектирования структуры процессора // Радиоэлектроника и информатика. 1998. №2(3) .С.120-124. 2. *Комплект БИС К1804 в процессорах и контроллерах* / В.М. Мещеряков, И.Е. Лобов, С.С. Глебов и др. М.: Радио и связь, 1990. 256 с. 3. *Сизов К.А.* Микропроцессор будущего: RISC, CISC или MISC?// Библиотека информационных технологий: Сборник статей. Вып.1 / Под ред. Г.Р. Громова. М.: Наука, 1990. С.118-124. 4. *Ельчанинов Д.Б., Лобода В.Г., Цуканов В.Ю.* Модели архитектуры MISC-процессора// Радиоэлектроника и информатика. 1999. №1(06). С. 85-89. 5. *Язык программирования Си* / Бриан В. Керниган, Деннис М. Ритчи. М.: Финансы и статистика, 1992. 270 с.6. Р. Хантер. Проектирование и конструирование компиляторов. М.: Финансы и статистика, 1984. 232 с. 7. *Цуканов В.Ю.* Взаимная адаптация аппаратно-программных средств в процессоре спецназначения // Радиоэлектроника и информатика. 2000. №1. С.59-63.

Поступила в редколлегию 02.09.2000

Петросов Давид Арегович, студент ХТУРЭ. Научные интересы: алгоритмическое и программное обеспечение функционально-ориентированных процессоров. Адрес: Украина, 61137, Харьков, ул. Межлаука, 11/7, кв.67.

Цуканов Виталий Юрьевич, аспирант ХТУРЭ. Научные интересы: алгоритмическое и программное обеспечение функционально-ориентированных процессоров. Адрес: Украина, 61111, Харьков, ул. Познанская, 2, кв.67, тел. 10-42-63.

УДК 681.325:519.713

В.В. ХАНЬКО

СТРУКТУРНО-ЛОГИЧЕСКИЕ МОДЕЛИ ДЛЯ АНАЛИЗА И ТЕСТИРОВАНИЯ СЕТЕВЫХ СЕГМЕНТОВ

Предлагаются структурно-логические модели компонентов компьютерной сети, используемые для реализации системы анализа и тестирования исправного поведения и неисправностей, деструктивно влияющих на техническое состояние и возникновение коллизий в сетевых сегментах.

1. Введение

При разработке и развертывании корпоративных компьютерных сетей со сложной архитектурой, а также при изменении архитектуры уже существующих сетей возникает необходимость в предварительном тестировании правильности принятых технических решений. Такая проверка позволяет выявить и устранить ошибки в проекте ещё до его реализации, что впоследствии дает возможность значительно сократить материальные и временные затраты, необходимые для устранения тех же ошибок и просчетов в уже развернутой сети. Подход к решению задач создания и модернизации сетей особенно актуален для корпоративных сетей крупных предприятий, которые, как правило, неоднородны, имеют сложную архитектуру, в связи с чем их модернизация сопряжена с большими трудностями.

Задача предварительной проверки правильности принятых технических решений может быть решена путём моделирования спроектированной сети либо отдельных элементов уже существующей с помощью специализированных пакетов, например, NetMaker XA или COMNET Predictor [1]. Однако они весьма дороги, поэтому приобретение такого рода программ компанией, деятельность которой не связана с компьютерными технологиями, не является приоритетным. Кроме того, воспользоваться услугами компаний – системных интеграторов по ряду причин (например, отсутствие таковых в данном регионе) также бывает проблематично. В этих условиях администратору сети приходится обходиться своими силами, создавая инструментарий для диагностики своей сети самостоятельно, используя такие языки программирования, как Perl или C++, которые часто входят в дистрибутивный комплект большинства операционных систем. Несмотря на то, что данные языки являются мощным средством создания приложений, их применение для построения моделей *параллельных* процессов, происходящих в аппаратуре, связано с большими трудностями ввиду отсутствия в них

специализированных средств, ориентированных на описание такого рода объектов и процессов. Хорошей средой для создания таких моделей являются HDL-языки [2]. Изначально ориентированные на описание и моделирование параллельных процессов, происходящих в электронной аппаратуре, они хорошо подходят для моделирования процессов, происходящих в компьютерных сетях. В последнее время появилось множество компиляторов VHDL, как свободно распространяемых (например, проект FreeHDL для среды Unix), так и коммерческих (Active-HDL от Aldec Inc., VHDL Studio от Green Mountain Computing Systems, Inc.), поэтому вопрос выбора конкретного продукта является делом вкуса пользователя.

Актуальной представляется разработка системы моделирования сегментов компьютерной сети на основе использования структурно-логических моделей, учитывающих топологию и функциональные особенности компонентов компьютерной сети, которые представлены на языке описания аппаратуры высокого уровня, в целях уменьшения временных и материальных затрат анализа и тестирования корпоративной сети и диагностирования неисправностей, приводящих к отказам или возникновению коллизий.

Для достижения поставленных целей решаются основные задачи:

– разработка общей концепции модели компьютерной системы и сети в виде триады <H,S,N>, позволяющей распространить методы анализа и тестирования вычислительных систем [3,4] на компьютерные сети с учетом особенностей последних, связанных с рассредоточением в пространстве аппаратного, программного, информационного обеспечений;

– создание структурно-логических моделей различных топологий и технологий исполнения локальных вычислительных сетей (ЛВС) на основе теории графов, кубического исчисления [2], ориентированных на существующие системы моделирования и позволяющих уменьшить время определения коллизий и неисправностей;

– создание асинхронных моделей компонентов ЛВС в целях реализации методов синхронного и асинхронного моделирования [2] влияния неисправностей на функционирование сети и выявления деструктивных процессов, влияющих на утилизацию каналов;

– разработка функциональных и физических моделей неисправностей канального и физического уровней в форме <дефект – диагностические признаки>, позволяющих проектировать таблицы неисправностей в целях диагностирования технического состояния сети, установления факта наличия коллизий и места возникновения неисправности [5-9];

– разработка модели сегмента вычислительной сети с использованием аппарата сетей Петри, которая позволяет анализировать техническое состояние сети и определять деструктивные процессы, связанные с наличием коллизий или физических дефектов [10];

– создание модели описания неисправностей в виде таблицы, где заданы функциональные и физические дефекты, возникающие в сети на канальном и физическом уровнях, в целях реализации безусловного

поиска дефектов на основе анализа физических признаков их проявления в сегменте [11];

– разработка и практическое применение программного комплекса моделирования исправного поведения и неисправностей, позволяющих осуществлять асинхронный анализ сегментов сети в целях определения коллизий и локализации заданных неисправностей компонентов сети.

2. Табличная форма описания графов

Достоинство автоматного представления графовой структуры заключается в технологичности ее дальнейшей реализации на VHDL-языке высокого уровня описания аппаратуры в целях верификации проектов или их реализации в аппаратуре FPGA, CPLD. Поэтому табличные модели представления графовых структур сетевых сегментов следует рассматривать как отвечающие современным технологиям проектирования.

Граф есть совокупность вершин, связанных дугами, иначе, граф – это система бинарных отношений или переходов, которые можно разбить на элементарные связи смежных вершин, одни из которых истоки (исходные состояния), другие – стоки (конечные состояния). В автоматном представлении определение графа есть частный случай, задаваемый выражением

$$M_{FSM} = (X, Y, Z, Z_0, f, g);$$
$$Z(t) = f[X(t), Z(t-1)];$$
$$Y(t) = f[X(t), Z(t-1)],$$

где X, Y, Z, Z_0, f, g – параметры задания конечного автомата: входной алфавит; множества: выходных, отмеченных состояний; начальное состояние; функция переходов; функция выходов соответственно.

С учетом условий, характерных для задания графа, характеристические уравнения конечного автомата приводятся к виду:

$$(X; Y = \varnothing; Z = H, Z_0 = \varnothing, f = E, g = \varnothing) \Rightarrow$$
$$\{H(t) = E[X, H(t-1)]; Y(t) = \varnothing\} \Rightarrow$$
$$G = (H, E, X), H = E(X, H),$$

здесь H, E, X – множества вершин, дуг, идентификаторов дуг. Функция E задает маркированный символом X_S переход из вершины-истока H_I графовой структуры в вершину-сток H_S: $H_S = E(X_S, H_I)$.

Выполнив маркирование самой функции E, можно исключить, как избыточную, переменную X. В этом случае получается следующая автоматная форма описания модели графовой структуры:

$$G = (H, E), H = E(H),$$

где каждый переход обозначен выражением $H_S = E_S(H_I)$.

Следуя такому определению, каждый символ двухтактного алфавита [3]:

A^2={Q=00,E=01,H=10,J=11,O={Q,H},I={E,J},A={Q,E},
B={H,J},S={Q,J},P={E,H},C={E,H,J},F={Q,H,J},L={Q,E,J},
V={Q,E,H},Y={Q,E,H,J}, A^1={0,1,X={0,1}}, $\varnothing$ (U)}

есть граф, содержащий две вершины, соединенные дугами, число которых в зависимости от символа определяется интервалом значений от 1 до 4. Однако интерес представляет не изоморфизм двухтактных символов элементарным графам, но возможность преобразования любой структуры в кубическое покрытие с последующим анализом ее с помощью методов, ориентированных на обработку модели автоматного уровня.

Процедура проектирования кубической формы представления графа по известной структуре объекта состоит из нескольких этапов.

1. Определение минимального множества разрядов (автоматных переменных) в целях присвоения каждой из N вершин двоичного кода по формуле нахождения целой части числа, округленной в большую сторону: $n=]\log_2 N[$.

2. Дуге ставится в соответствие куб кубической формы представления графа (КФПГ) C_i, получаемый на основе операции конкатенации

*	0	1	X	Z
0	Q	E	A	G
1	H	J	B	T
X	O	I	Y	K
Z	0	1	X	Z

над соответствующими разрядами кодов вершин истока и стока:

$$C_i = C_{pq} = (H_p * H_q)\Big|_{\forall p,q(H_p \to H_q)}, i = \overline{1,k},$$

где k – число дуг в графе или кубов.

3. Итеративная минимизация (I – шаг итерации):

$$C_i^I = C_i^{I-1} \cup C_t^{I-1} \leftarrow \exists! j(C_{ij}^{I-1} \neq C_{tj}^{I-1}), I = \overline{1,s}$$

с последующим применением операции векторного поглощения

$$C_i^I \subseteq C_t^I \leftarrow \forall j(C_{ij}^I \cap C_{tj}^I = C_{ij}^I)(i,t = \overline{1,k})$$

к исходным кубам покрытия, в которых каждый вектор может участвовать более одного раза.

Для произвольной структуры построение КФПГ начинается с определения количества разрядов n для кодирования N вершин, удовлетворяющего неравенству $N \le 2^n$. Затем выполняется итеративная процедура определения вершины H_i с максимальной степенью полуисхода $S_I(H_i)$ (полузахода $S_O(H_i)$) еще не отмеченных стоков (истоков). Последние кодируются в порядке возрастания двоичных чисел подмножествами кодов, кратных степени двойки. Такими являются: (0,1); (00,01,10,11); (000, 001, 010 ,..., 111). При равенстве полустепеней просчитываются

варианты кодирования вершин вычислением оценки качества Q с последующим выбором решения, имеющего минимальный критерий. Процедура не гарантирует построения оптимального покрытия, которое может быть получено с помощью эвристических приемов.

Иллюстрацией процедуры проектирования КФПГ служит построение покрытия для графа топологии "звезда", в которой восемь рабочих станций соединены между собой посредством коммутатора (рис.1).

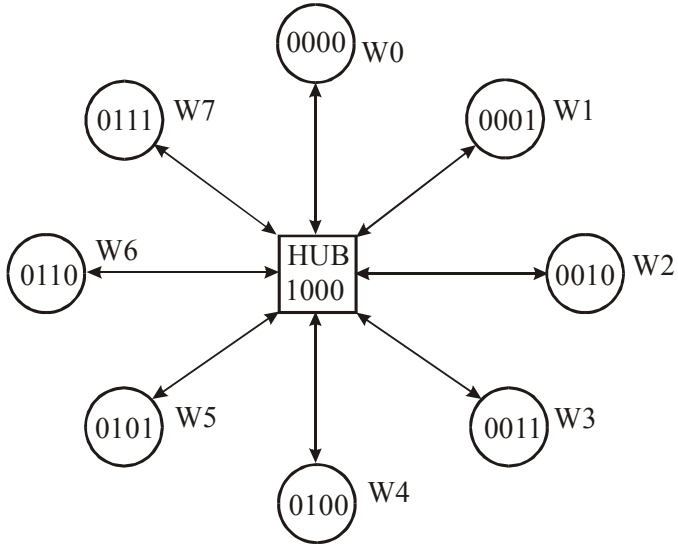

Рис. 1. Граф топологии ArcNet

Для такой структуры выполняется произвольное кодирование всех вершин, соответствующих рабочим станциям, при условии, что код коммутатора должен соответствовать старшему в десятичном эквиваленте числу. Полученные пары двоичных векторов, соответствующие всем дугам графа, определенным подмножествами $\{W_i \to HUB, HUB \to W_i\}$, а также сконкатенированные двухтактные векторы имеют следующий вид:

$$W_i \to HUB = \left|\begin{array}{cc|c} 0000 & 1000 & EQQQ \\ 0001 & 1000 & EQQH \\ 0010 & 1000 & EQHQ \\ 0011 & 1000 & EQHH \\ 0100 & 1000 & EHQQ \\ 0101 & 1000 & EHQH \\ 0110 & 1000 & EHHQ \\ 0111 & 1000 & EHHH \end{array}\right|; \quad W_i \to HUB = \left|\begin{array}{cc|c} 1000 & 0000 & HQQQ \\ 1000 & 0001 & HQQE \\ 1000 & 0010 & HQEQ \\ 1000 & 0011 & HQEE \\ 1000 & 0100 & HEQQ \\ 1000 & 0101 & HEQE \\ 1000 & 0110 & HEEQ \\ 1000 & 0111 & HEEE \end{array}\right|.$$

Минимизация полученных двухтактных кубов, соответствующих дугам подмножеств $\{W_i \to HUB, HUB \to W_i\}$, дает результат:

$$\{W_i \to HUB = EOOO, HUB \to W_i = HBBB\},$$

формирующий кубическое двухтактное покрытие графа в виде двух дуг

$$G_{Arcnet} = \left|\begin{matrix} EOOO \\ HBBB \end{matrix}\right|.$$

3. Временные модели компонентов ЛВС

Будем рассматривать модель ЛВС как совокупность рабочих станций, соединенных между собой структурированной кабельной системой. Также будем считать, что каждая станция по отношению к другим в конкретный момент времени может находиться в режиме ожидания, приема или передачи информации. Каждый компонент сети имеет номинальную задержку распространения сигналов или пакетов.

Коллизия есть состязания двух или более станций, приводящие к искажению при приеме информационных пакетов. Следовательно, анализировать сеть на предмет наличия коллизий можно методами синхронного и асинхронного моделирования цифровых систем. Суть этих методов заключается в квантовании временного континуума модельными задержками переходных процессов и прохождения сигналов по компонентам сети. Модель сегмента, учитывающая синхронно-асинхронный характер состязаний, представлена компонентами

$$\mu = < \varphi, \tau, \lambda, \alpha >,$$

где $\varphi = [t_i, t_{i+1}]$– интервал времени между подачей очередного набора (пакета) является стандартным атрибутом протокола сети; $\tau = [\tau_{min}, \tau_{max}]$ – интервал номинальных задержек компонентов цифровой системы (сети); $\lambda = [\lambda_{min}, \lambda_{max}]$ – интервал (неопределенности) разброса начала передачи сигналов, единый для всех станций, который является функцией различных путей прохождения сигналов от передатчиков до приемника по кабелю; $\alpha = \{0,1,X,\varnothing\}$ – алфавит описания сигналов в сетевом сегменте.

Для создания программных приложений моделирования состязаний или коллизий трудно оперировать временем, не имея наперед заданного для каждой системы или сети масштаба. Поэтому при создании модели реальной корпоративной компьютерной сети и ее компонентов используется процедура нахождения наибольшего общего делителя для задержек.

Пусть элемент сети g_i реализует некоторую логическую функцию, а τ_i определяет задержку появления сигнала на станции-приемнике W_i= = $g_i(t-\tau_i)$. Для упрощения процедуры анализа сети с асинхронными моделями примитивов выбирается единый масштаб времени Δt, как наибольший общий делитель номинальных задержек элементов τ_i, благодаря чему значения модельных задержек r_i являются целыми

числами. При этом реальная задержка компонента определяется в виде $\tau_i = r_i \Delta t$, где r_i – целое положительное число; Δt – единица масштаба.

Пусть три элемента имеют задержки (60, 90, 150 нс). Масштаб, определяемый как наибольший общий делитель, имеет значение Δt=30 нс. Тогда модельные задержки находятся по выражению $r_i = \tau_i / \Delta t$ и равны соответственно 2, 3, 5. Смена значений сигналов на линиях схемы осуществляется только в моменты 0, Δt, $2\Delta t$, ... реального времени. Значение сигнала на приемнике W_i в такте t определяется состоянием входов примитива в такте t - r_i.

Модель общего анализа коллизий сети. Многообразие методов анализа цифровых систем основывается на вариациях компонентов μ:

$$\mu = < \varphi = [t_i, t_{i+1}]; \tau = [\tau_{min}, \tau_{max}]; \lambda = [\lambda_{min}, \lambda_{max}], \alpha = \{0,1,X,\varnothing\} >.$$

Последнее уравнение порождает многообразие методов логико-временного анализа:

1. Синхронный анализ коллизий сети:

$$t_{i+1} - t_i = const;\ \tau_{min} = \tau_{max} = 0;\ \lambda_{min} = \lambda_{max} = const; \alpha = \{0,1,X,\varnothing\}.$$

2. Асинхронный анализ коллизий сети:

$$t_{i+1} - t_i \neq const;\ \tau_{min} = \tau_{max} \neq 0;\ \lambda_{min} = \lambda_{max} = 0.$$

3. Δ-троичный анализ коллизий сети:

$$t_{i+1} - t_i = const;\ \tau_{min} = \tau_{max} \neq 0;\ \lambda_{min} = \lambda_{max} = q; \alpha = \{0,1,X,\varnothing\}.$$

4. Анализ с нарастающей неопределенностью:

$$t_{i+1} - t_i = const;\ \tau_{min} \neq \tau_{max};\ \lambda_{min} = \lambda_{max} = 0; \alpha = \{0,1,X,\varnothing.$$

5. Δ-троичное моделирование коллизий сети с нарастающей неопределенностью:

$$t_{i+1} - t_i = const;\ \tau_{min} \neq \tau_{max};\ \lambda_{min} \neq \lambda_{max}; \alpha = \{0,1,X,\varnothing.$$

Соединение двух последних методов в единый дает возможность изменением параметра разброса интервала переключения входных сигналов $q = \{\Delta t, 2\Delta t, ...\} \in [r_{min} = \frac{1}{\Delta t}\lambda_{min}, r_{max} = \frac{1}{\Delta t}\lambda_{max}]$ и интервала неопределенности номинальных задержек

$0 \le \frac{1}{\Delta t}\tau_{min} \le \frac{1}{\Delta t}\tau_i \le \frac{1}{\Delta t}\tau_{max}$ получать любой из ранее рассмотренных методов анализа состязаний или коллизий. Но универсальность такого подхода связана с достаточной сложностью его практической реализации.

4. Модели неисправностей корпоративных сетей

Класс дефектов корпоративной компьютерной сети представлен двумя подмножествами $F = \{F^f, F^m\}$, где F^f – физические дефекты компонентов сети; F^m – модельные неисправности элементов сети, наличие которых приводит к неправильному функционированию модели или к ее несоответствию описанному объекту.

Физические дефекты двух типов $F^f = \{F^f_s, F^f_o\}$ возникают вследствие неправильных действий, связанных с процессами проектирования, изменения, дополнения, модернизации, эксплуатации, ремонта компьютерной сети, что формализует подмножество субъективно возникающих дефектов, описываемое в виде элементов

$$F^f_s = \{F^f_d, F^f_c, F^f_a, F^f_m, F^f_e, F^f_r\},$$

а также в результате отказа аппаратуры по причине ее износа, морального старения, наличия внутреннего дефекта, формирующих подмножество

$$F^f_o = \{F^f_w, F^f_f, F^f_s\}.$$

Учитывая высокую надежность комплектующих изделий компьютерной системы или сети, определяемую тем фактом, что 90 % физических неисправностей приходится на места соединений (пайки) структурированной кабельной системы с активным и пассивным оборудованием, на практике монтаж и наладка кабельной системы занимают большую часть времени проектирования локальной сети.

Все дефекты, за исключением тех, которые связаны с исчезновением информации, выявляются путем перехвата и анализа кадров канального уровня. Неисправности сетевой платы связаны с конкретными MAC-адресами [1]. Исчезновение информации в активном оборудовании на канальном уровне определить практически невозможно. Косвенным признаком этой ошибки является большое число повторных передач, выполняемых по конкретному MAC-адресу. В данном случае следует проанализировать информацию, инкапсулированную протоколами верхних уровней в кадр канального уровня.

Далее рассмотрим класс функциональных дефектов для метода доступа к среде CSMA/CD [8], принятой в Ethernet, где существует значительный процент ошибок, называемых коллизиями. Множество различимых коллизий, как проявление для наблюдателя существенных для идентификации физических дефектов на канальном уровне, представлено в виде

$$C^k \subset F^f_f, C^k = \{C_{loc}, C_{rem}, C_{sho}, C_{lon}, C_{jab}, C_{crc},\}.$$

Каждому функциональному дефекту из списка ставится в соответствие определение и модель диагноза – как он проявляется и что ему соответствует на физическом уровне реализации компьютерной сети. Множество диагностических признаков функциональных дефектов определено как

$$R(C) = (R_{pre}, R_{cad}, R_{crc}, R_{cab}, R_{rec}),$$

$$R_i \in R(C) = \begin{cases} R^0_i \leftarrow R(C) \oplus R^*(C) = 0; \\ R^1_i \leftarrow R(C) \oplus R^*(C) = 1; \\ R^X_i \leftarrow R(C) - \text{unnown state}, \end{cases}$$

где R^0_i, R^1_i – идентификаторы совпадения, искажения фактической реакции $R(C)$ на линии с эталонными сигналами $R^*(C)$; R^X_i – несущественность параметра для формирования модели диагноза; $(R_{pre}, R_{cad}, R_{crc}, R_{cab}, R_{rec})$ – вектор диагностических параметров: преамбула; кадр в пределах 64-1518 байтов; контрольная сумма CRC [1]; уровень напряжения в кабеле; сигнал на линиях приема передающей станции.

Локальная коллизия – это процесс одновременной передачи информации двумя и более станциями, проявляющийся для наблюдателя в домене в виде уменьшения 64 байтов кадра преамбулы и искажения контрольной суммы CRC. Сопровождается удвоением уровня напряжения в кабеле (10BASE-2) или появлением сигнала на линиях приема передающей станции (10BASE-T) [6]. Модель диагноза:

$$C_{loc} = (R^0_{pre}, R^1_{cad}, R^1_{crc}, R^1_{cab}, R^1_{rec}).$$

Коллизия, возникающая в другом сегменте сети, за повторителем, когда наблюдатель и передатчик находятся в разных доменах, называется удаленной. Характеризуется правильно переданным кадром с неверной контрольной суммой и нормальным уровнем напряжения в сетях 10BASE-2, либо отсутствием сигнала на линиях приема передающей станции в сетях 10BASE-T. Модель диагноза:

$$C_{rem} = (R^0_{pre}, R^0_{cad}, R^1_{crc}, R^0_{cab}, R^0_{rec}).$$

Коллизия, которая фиксируется после передачи станцией первых 64 байтов кадра, называется поздней. Указывает на наличие проблем в сети. В 10Base-T поздние коллизии фиксируются анализаторами как ошибки CRC. Модель диагноза: $C_{lat} = (R^0_{pre}, R^0_{cad}, R^1_{crc}, R^0_{cab}, R^X_{rec})$.

Коллизия короткий кадр определяется длиной меньше 64 байтов, следующей после 8-байтной преамбулы, с правильной контрольной последовательностью (CRC). Модель диагноза:

$$C_{sho} = (R^0_{pre}, R^1_{cad}, R^0_{crc}, R^X_{cab}, R^X_{rec}).$$

Коллизия, определяемая кадром длиной более 1518 байтов, с правильной либо неправильной CRC, называется длинным кадром. Модель диагноза: $C_{lon} = (R^0_{pre}, R^1_{cad}, R^X_{crc}, R^0_{cab}, R^X_{rec})$.

Jabber – коллизия, имеющая кадр более 1518 байтов с неправильной CRC, вследствие нарушения паузы 9,6 мкс между кадрами. Модель диагноза: $C_{jab} = (R^0_{pre}, R^1_{cad}, R^1_{crc}, R^0_{cab}, R^X_{rec})$.

Коллизия типа ошибки CRC есть нормально оформленный кадр допустимой длины (64-1518 байтов) с неправильной CRC. Модель диагноза: $C_{jab} = (R^0_{pre}, R^0_{cad}, R^1_{crc}, R^0_{cab}, R^X_{rec})$.

Другие функциональные дефекты, не вызываемые ошибками канального уровня, идентифицируются путем дополнительного анализа результатов тестирования сети или сегмента. К ним следует отнести:

$$C^{\bar{k}} \subset F^f_f, C^{\bar{k}} = \{C_{loo}, C_{gho}, C_{equ}, C_{err}, C_{blo}, C_{sto}, C_{mar}\}.$$

Множество дефектов в соответствии с заданной последовательностью определено ниже:

1. Потеря информации, определяемая в виде частых повторных передач, не вызванных ошибками канального уровня (пакеты IP, IPX), либо с отключением порта концентратора.

2. Блики – последовательность сигналов, отличных по формату от кадров Ethernet, не содержащая разделителя (SFD) и имеющая длину более 72 байтов. Выявляются на стадии стрессового тестирования.

3. Ошибки выравнивания – число битов в кадре не кратно числу байтов, вследствие неисправного активного оборудования.

4. Искажение информации в активном оборудовании на транспортном уровне (протоколы IP, IPX), не обнаруживаемое на канальном уровне, по причине несогласованности работы буферов в активном оборудовании.

5. Блокировка канала – захват активным оборудованием передающей среды по причине неисправности активного оборудования или несоблюдения им стандарта CSMA/CD, связанного с нарушением паузы 9,6 мкс.

6. Широковещательные "штормы" – высокий уровень широковещательных передач в сети, занимающий полосу пропускания среды, вызванный дефектами мостов, некорректно работающим программным обеспечением, инсталлированным в сети.

7. Ошибки маршрутизации, связанные с дубликатными адресами протоколов верхних уровней и приводящие к прямым конфликтам в сети, вызванные ошибками в конфигурации ПО рабочих станций, маршрутизаторов, брандмауэров, серверов.

Приведенные функциональные дефекты являются следствием не только возникающих неисправностей, но и ошибок проектирования, связанных с:

– построением сегментов чрезмерной длины;

– прокладкой кабеля без учета влияния на него внешних источников различных помех;

– большим числом перегибов кабеля в сегменте;

– отсутствием или неверной топологией заземления;

– выбором некачественного активного оборудования;

– некорректной установкой, конфигурированием и использованием программного обеспечения серверов, маршрутизаторов, рабочих станций.

5. Моделирование сегментов на сетях Петри

В качестве исходных данных для построения модели ВС используется проект локальной сети Ethernet 10Base2, состоящей из 7 узлов (персональных компьютеров) (рис. 2).

Поскольку сети Петри используются для асинхронного моделирования параллельных событий в системе, где каждому из них соответствует переход t_j, а условию – позиция p_i, то необходимо представить процесс функционирования ВС со стороны событийно-условного аспекта, причем функционирование ВС включает в себя: передачу пакета, прием пакета, обнаружение коллизий.

Таким образом, исходя из принципов действия протокола CSMA/CD, принятого в Ethernet 10Base2, технология Ethernet 10Base2 в сетях Петри будет иметь следующие состояния (условия): p_1 – пакет готов к передаче; p_2 – наличие несущей (линия занята); p_3 – отсутствие несущей; p_4 – период отсутствия несущей: 9,6 мкс, 2 мкс; p_5 – отсутствие коллизии; p_6 – выявление коллизии во время передачи пакета; p_7 – число коллизий $N_{col} \le 10$; p_8 – число коллизий $10 < N_{col} < 16$; p_9 – число коллизий $N_{col} \ge 16$; p_{10} – сброс передаваемого пакета; p_{11} – получен диапазон $[0,2^{N_{col}} - 1]$; p_{12} – выбрана случайная величина из диапазона $[0,2^{N_{col}} - 1]$; p_{13} – получена случайная задержка отката; p_{14} – узел готов к очередной попытке передать пакет; p_{15} – p_{21} – станция 1-7 получила пакет; p_{22} – значение адреса в поле приемника пакета и адрес получившей станции совпали; p_{23} – значение адреса в поле приемника пакета и адрес получившей станции не совпали; p_{24} – получена jam-последовательность; p_{25} – выявление коллизии во время прослушивания среды; p_{26} – среда очищена от коллизий.

При этом происходят следующие события: t_1 – подготовка пакета к передаче; t_2 – прослушивание среды на наличие или отсутствие несущей (carrier); t_3 – передача пакета и прослушивание среды на предмет коллизий; t_4 – занесение коллизии в счетчик; t_5 – удаление пакета из очереди на передачу; t_6 – посылка запроса к программному приложе-

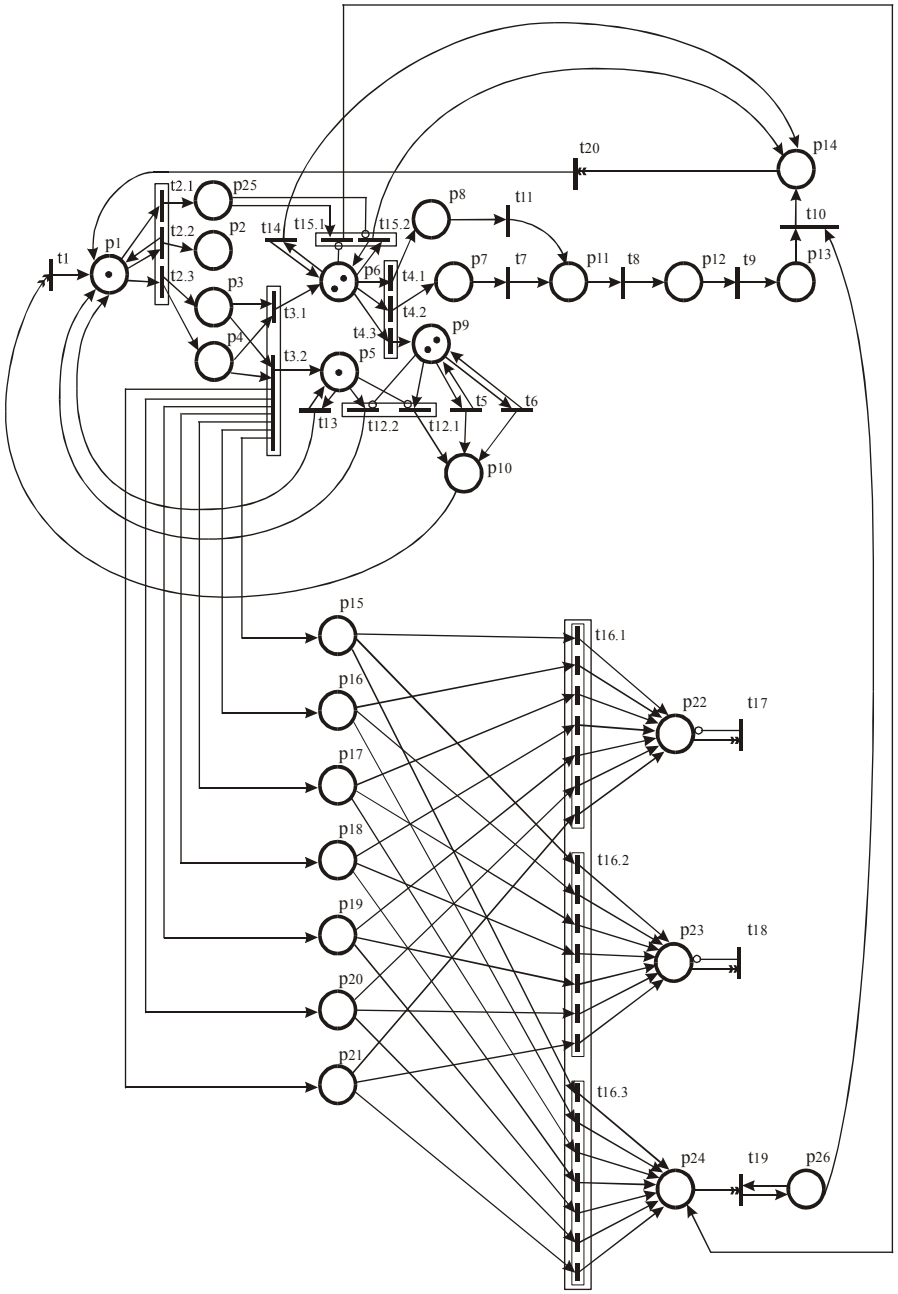

Рис. 2. Модель сегмента корпоративной сети

нию о новом формировании пакета; t_7 – вычисление значения $2^{N_{col}}$; t_8 – выбор случайной величины из интервала $[0,2^{N_{col}}-1]$; t_9 – умножение выбранной случайной величины на 51,2 мкс в целях получения случайной задержки отката; t_{10} – ожидание в течение случайной задержки отката; t_{11} – выбор случайной величины из интервала $[0,2^{N_{col}=10}-1]$; t_{12} – очищение счетчика коллизий; t_{13} – ожидание в течение dt, где dt=9,6 мкс, dt=9,6 мкс+x, dt=2 мкс; t_{14} – прекращение пересылки пакета; t_{15} – посылка jam-последовательности; t_{16} – сверка поля адреса приемника кадра с адресом узла получателя; t_{17} – узел не принимает пакет; t_{18} – узел принимает пакет; t_{19} – режим бездействия (silent mode) всех узлов сети; t_{20} – очередная попытка передать пакет.

Здесь наличие буквы d означает, что имеет место макропереход – действие, после которого система может перейти в одно из ряда состояний (постусловий), причем эти состояния не пересекаются. Например, макропереход t_2 может привести ВС к состоянию либо наличия несущей (среда занята), либо ее отсутствия, либо к состоянию обнаружения jam-последовательности, после чего для каждого из состояний происходят собственные непересекающиеся события.

С помощью использования макропереходов, состоящих из простых переходов, а также при соблюдении правил построения и срабатывания макропереходов можно достичь выполнения условия непересекаемости.

В модели (см. рис.2) можно выделить последовательную часть передачи пакета любой станцией и параллельную часть его приема и обработки всеми остальными станциями в сети, что полностью соответствует принципу приема-передачи информации в сети Ethernet 10Base2.

Выполнение сети Петри начинается с запуска одного из простых переходов макроперехода t_2, для чего в позиции p_1 должна присутствовать метка, свидетельствующая о готовности пакета к передаче, имея в виду одно из приведенных ниже состояний:

– формирование пакета для передачи (первичная стадия);

– готовность сформированного пакета к передаче при освобождении среды,

– готовность сформированного ранее пакета к передаче после обнаружения и обработки коллизии;

– готовность пакета к передаче после его повторного формирования в случае, когда количество коллизий $N_{col} \geq 16$.

По мере продвижения метки (меток) по сети происходит запуск соответствующих переходов, что является сущностью протокола CSMA\CD.

Ряд переходов в данной модели в соответствии с функционированием протокола CSMA\CD требует параллельного запуска. При выявлении коллизии во время передачи пакета (позиция p_6) должны одновременно выполниться следующие действия:

– прекращение пересылки пакета,

– занесение коллизии в счетчик коллизий,

– посылка jam-последовательности.

Таким образом, в позиции p_6 для параллельного запуска переходов, выполняющих указанные выше действия, необходимо наличие трех меток, причем две из них хранятся в позиции p_6 до запуска сети, что изначально сигнализирует о необходимости параллельного выполнения переходов по приходу в p_6 третьей метки.

В модели такие метки, сигнализирующие о необходимости параллельного выполнения переходов, требуются также в позициях p_5 и p_9.

Однако при запуске переходов все метки уходят из упомянутой позиции и при следующем цикле выполнения сети Петри параллельный запуск требуемых переходов не может осуществиться из-за недостаточного количества меток в позиции.

В модели, представленной сетью Петри, имеет место такой недостаток как накапливание меток в некоторых позициях. И если процесс накапливания не влияет на корректность выполнения сети (позиция p_2), то большинство фактов накапливания меток говорит об угрозе некорректной реализации протокола CSMA\CD (позиции p_1, p_{10}, p_{14}, p_{23}, p_{24}).

Предложен следующий способ избежания данной проблемы: во-первых, можно использовать уничтожающие дуги, если процесс, запускаемый накопленными метками, работает независимо во времени от других процессов (отбрасывание пакета, пришедшего не по адресу – позиция p_{23}); во-вторых, если для позиции p_i, где имеет место накапливание меток из-за количества входящих в позицию дуг > 1, существует количество переходов $t_j \in T^p$=1, то можно от позиции p_i вывести конвейерные дуги к упомянутому переходу. Так было сделано с позициями p_{10}, p_{14}, p_{22}, p_{23}, p_{24}.

Однако при этом остается проблема удаления лишних меток из позиции p_1. Перед каждым запуском в p_1 должна находиться одна и только одна метка, а по завершению цикла выполнения сети в p_1 в случае успешной передачи пакета оказываются 2 метки (от t_{13} и $t_{12.2}$).

Таким образом, следует установить условие по умолчанию: если в p_1 оказывается количество меток >1, лишние метки выбрасываются либо просто не проходят (можно говорить о жестком ограничении "объема" позиции p_1).

Модель (см. рис. 2) полно представляет функционирование вычислительной сети архитектуры Ethernet 10Base2, а также позволяет анализировать структуру и динамическое поведение модели с использованием традиционных методов анализа сетей Петри. Имеется возможность представления ВС в виде совокупности взаимодействующих компонентов, которым присущи свойства совмещенности и параллелизма, а также состояний компонентов.

Однако совмещенная природа действий создает некоторые трудности при моделировании, так как при взаимодействии компонентов должно выполняться условие синхронизации, т.е. согласование по времени, что очень сложно реализовать в сетях Петри. Отсутствие возможности представления в сетях Петри ряда параметров, связанных с временными характеристиками (задержки прохождения сигнала в физических компонентах сети, задержки при обработке ситуации с возникновением коллизии), приводит к неполному представлению модели ВС, что является недостатком использования данного теоретического аппарата для целей моделирования.

6. Система логического анализа сетей

Программный комплекс логического анализа сетей, основанный на предложенных моделях, предназначен для предварительной проверки принятых при разработке и модернизации сети технических решений, а также для моделирования работы фрагментов действующих сетей под воздействием неисправностей. Он позволяет моделировать работу сегмента сети Ethernet 10BASE-2 для любого допустимого количества узлов с учетом особенностей режима работы каждого из них. В зависимости от поставленной задачи диагностирования и исходных данных, программа анализа позволяет контролировать такие параметры, как уровень утилизации канала для данного участка сети, уровень повторных передач, количество успешно переданных за время моделирования кадров и количество произошедших коллизий для каждого узла сети.

Среда Active-HDL, в которой работает данная программа, предоставляет пользователю возможность наблюдать за изменениями любого из используемых в программе сигналов и их комбинаций, что позволяет наглядно оценить ту или иную ситуацию, имеющую место в исследуемом сегменте сети. В программе имеются средства для идентификации типа коллизии с указанием номеров конфликтующих узлов сети и времени, что облегчает поиск нужного фрагмента временных диаграмм.

Программа позволяет моделировать следующие неисправности: обрыв участка кабеля; обрыв в терминаторе; чрезмерная длина сегмента; неисправности сетевой платы узла сети (неисправности блоков формирования, кодирования, передачи, приема и декодирования кадра, блоков обработки и обнаружения коллизий) и программного обеспечения, которые проявляются в виде таких ошибок, как высокий уровень местных коллизий при низком уровне утилизации канала, поздние коллизии, короткий кадр, длинный кадр, jabber, потеря информации, блокировка канала.

Модульная структура программы даёт возможность пользователю, знакомому с VHDL, самостоятельно вносить необходимые изменения в программу, а также дополнять её новыми модулями.

В зависимости от характера решаемой диагностической задачи перед началом работы с программой пользователю может потребоваться следующая информация об архитектуре исследуемого сегмента сети: количество узлов, подключенных к исследуемому сегменту; длина исследуемого сегмента (м); длина кабеля между каждой парой узлов (м); характер работы узлов сети: сервер, рабочая станция; основные маршруты потоков информации в исследуемом сегменте сети. Например, для случая, когда в нем находится сервер интернет и несколько рабочих станций, то поток информации будет направлен от каждой рабочей станции до сервера и от сервера к каждой рабочей станции; средняя частота запросов доступа к среде передачи данных для каждого узла сети. Если предполагается моделировать работу сети под воздействием тех или иных ошибок, то пользователь должен располагать информацией об их типе, о том, как они влияют на техническое состояние сети, и что важнее всего, о причинах возникновения данных ошибок (например, неисправности или несоответствие требованиям стандарта кабельной системы, оборудования, программного обеспечения).

Благодаря тому, что при создании системы моделирования сети использовался подход, обеспечивающий построение VHDL-модели сети из функционально законченных модулей, удалось добиться достаточной компактности VHDL-кода, его доступности для понимания, упрощения процедуры построения модели сети в целом. Кроме того, данный подход позволяет легко вносить изменения в VHDL-описания компонентов сети, строить на основе этих описаний библиотеки VHDL-моделей сетевого оборудования, перестраивать модель под нужную спецификацию физической среды Ethernet.

При создании VHDL-модели сети использовались только стандартные средства и библиотеки языка VHDL, благодаря чему возможно использование данного VHDL-кода с VHDL-системами других производителей.

При тестировании данной программы было установлено, что скорость её выполнения недостаточна для моделирования больших сетей в течение длительного интервала времени. Это связано, главным образом, с использованием в модели битового представления кадра Ethernet. В настоящее время проводятся исследования в целях создания VHDL-модели сети 100BASE-T, в которой предполагается использовать модифицированную концепцию представления кадров, благодаря чему возможно увеличение скорости работы программы на один-два порядка.

Применение данной программы показало её пригодность для моделирования поведения небольших сетей Ethernet 10BASE-2 в различных режимах. Полученные при тестировании программы данные достаточно хорошо коррелировались с результатами, полученными с помощью промышленных анализаторов протоколов.

Для решения задачи определения технического состояния ВС и поиска дефектов используются таблицы неисправностей, являющиеся результатом работы программы логического анализа сети. С их помощью локализуются физические и функциональные дефекты, возникающие в сети на канальном и более высоких уровнях описания протокола OSI.

7. Заключение

В целях уменьшения временных затрат логического анализа при восстановлении работоспособности корпоративной компьютерной сети и ее компонентов на стадиях проектирования и эксплуатации путем использования логических и топологических моделей сети, ориентированных на тестирование исправного поведения и диагностирование неисправностей, приводящих к отказам или возникновению коллизий, предложены:

– структурно-логические модели различных топологий и технологий исполнения ЛВС на основе теории графов, кубического исчисления, сетей Петри и VHDL-языка описания аппаратуры высокого уровня, ориентированных на существующие системы моделирования и позволяющих уменьшить время определения коллизий и неисправностей;

– асинхронные модели компонентов ЛВС в целях реализации методов синхронного и асинхронного моделирования влияния неисправностей на функционирование сети и выявления деструктивных процессов, влияющих на утилизацию каналов;

– функциональные и физические модели неисправностей канального и физического уровней в форме <дефект – диагностические признаки>, позволяющие проектировать таблицы неисправностей в целях безусловного диагностирования технического состояния сети, установления факта наличия коллизий и места возникновения неисправности;

– модель сегмента вычислительной сети с использованием аппарата сетей Петри, которая позволяет анализировать техническое состояние сети и определять деструктивные процессы, связанные с наличием коллизий или физических дефектов;

– программный комплекс моделирования исправного поведения и неисправностей, позволяющий осуществлять асинхронный анализ сегментов сети в целях определения коллизий и локализации заданных неисправностей компонентов сети.

Список литературы: 1. *Стернс Том.* Учимся моделировать // Сети,1998.№5.С.35-39. 2. Active-VHDL Series. Book #1 - #4. Reference Guide. ALDEC Inc. 1998. 206 p. 3. *Хаханов В.И.* Техническая диагностика элементов и узлов персональных компьютеров.К.: ІЗМН. 1997.308 с. 4. *Бондаренко М.Ф., Кривуля Г.Ф., Рябцев В.Г., Фрадков С.А., Хаханов В.И.* Проектирование и диагностика компьютерных систем и сетей. К.: НМЦ ВО. 2000. 306 с. 5. *Спортак М., Паппас Ф.,*

Рензинг. Высокопроизводительные сети /DiaSoft, К. 1998. 421 с. 6. *Новиков Ю.В., Кондратенко С.В.* Аппаратура локальных сетей: функции, выбор, разработка. М.: ЭКОМ, 1998. 288 с. 7. *Новиков Ю.В., Кондратенко С.В.* Локальные сети: архитектура, алгоритмы, проектирование. М.: Издательство ЭКОМ, 2000. 312 с. 8. TechFest Ethernet Technical Summary. http://www.techfest.com/networking/lan/ethernet3.htm. 9. http://www.synapse.de/ban/HTML/P_LAYER2/Eng/P_lay215.html. 10. *Питерсон Д.* Теория сетей Петри и моделирование систем. М.: Мир, 1984. 264с. 11. 66. Автоматизация диагностирования электронных устройств/ Ю.В.Малышенко и др./ Под ред.В.П.Чипулиса. М.: Энергоатомиздат, 1986. 216с.

Поступила в редколлегию 12.07.2000

Ханько Вадим Викторович, аспирант кафедры автоматизации проектирования вычислительной техники ХТУРЭ. Научные интересы: техническая диагностика компьютерных систем и сетей. Адрес: Украина, 61166, Харьков, пр. Ленина, 14, тел. 40-93-26.

УДК 681.326: 519.713

В.И. ХАХАНОВ, А.С. ШКИЛЬ, Е.В. КОВАЛЕВ

МОДЕЛЬ ПРОЦЕССА ПЕРЕХОДА ОТ СОДЕРЖАТЕЛЬНОГО ГРАФА МИКРОПРОГРАММЫ К ГРАФУ АВТОМАТА

Рассматривается задача перехода от традиционной формы представления закона функционирования микропрограммного автомата в виде граф-схемы алгоритма (ГСА) к содержательному графу автомата. Полученное описание конечного автомата в виде содержательного графа используется в системе визуального ввода САПР Active-HDL. Рассматриваются особенности использования языковых конструкций VHDL при описании функций переходов и выходов конечных автоматов.

При описании проектов микропрограммно-управляемых цифровых операционных устройств существует два способа описания алгоритма их функционирования: в виде содержательной ГСА микропрограммы и в виде содержательного графа автомата [1]. В англоязычной технической литературе способ описания алгоритма функционирования автомата в виде содержательной ГСА (flow chart) часто называют ASM (Algorithmic State Mashine), а в виде содержательного графа автомата (Moore/Mealy state) – FSM (Finite State Mashine).

Указанные формы описания алгоритмов функционирования конечных автоматов (КА) используются современными САПР программируемой логики в системах визуального ввода проектов, носят конкурирующий характер и применение одного из них зависит от опыта и привычек разработчика. Большинство современных промышленных

САПР устройств программируемой логики в качестве основной системы визуального ввода закона функционирования КА используют содержательный граф автомата (FSM). В свою очередь большинство отечественных разработчиков цифровой аппаратуры привыкли оперировать описанием микропрограммных автоматов (МПА) в виде ГСА. Исходя из этого, актуальной является разработка формальной процедуры перехода от описания МПА в виде ГСА к описанию в виде содержательного графа автомата, с учетом особенностей конкретной системы визуального ввода САПР программируемых логических интегральных схем (ПЛИС), в частности Active-HDL фирмы Aldec [2].

Содержательная ГСА микропрограммы часто используется при проектировании МПА, так как алгоритмы выполнения операций в устройстве удобно описывать в графической форме. ГСА строится с использованием вершин четырех типов и дуг, связывающих вершины. Начальная вершина отмечает начало алгоритма и имеет единственный выход, из которого исходит дуга к первой выполняемой вершине графа. Для обозначения операторов и переходов используются вершины двух типов: функциональные и условные. Функциональная (основная) вершина определяет действие – совокупность функционально совместимых микроопераций, выполняемых параллельно. Микрооперации в вершине представляются в виде операторов присваивания. В функциональную вершину может входить любое, но не меньшее 1, число дуг, а выходит только одна дуга. Условная вершина используется для разветвления вычислительного процесса в одном из двух возможных направлений, выбор которого осуществляется текущим значением логического условия, указанного в вершине. Если условие имеет значение 0, вычислительный процесс развивается по дуге, отмеченной символом 0, в противном случае – по дуге, отмеченной символом 1. В условную вершину может входить любое число дуг, но выходят всегда две дуги. Конечная вершина отмечает конец микропрограммы. В конечную вершину может входить любое число дуг.

Кроме содержательной ГСА описание микропрограммы включает описание слов и массивов. Основным элементом информации, с которым оперирует функциональная микропрограмма, является слово. Наименование и формат слова задаются в следующем виде: $C(n_1{:}n_2)$, где C – идентификатор слова; n_1 и n_2 – номера старшего и младшего двоичных разрядов слова. Примем, что разряды слова нумеруются слева направо неотрицательными целыми числами, т.е. $n_1 \geq 0$, $n_2 \geq 0$, $n_1 < n_2$. В соответствии с описанием разряды слова получают номера n_1, n_1+1, ..., n_2. Так, описание $A(0{:}31)$ определяет 32-разрядное слово A, а описание $B(1{:}8)$ – 8-разрядное слово B. Описание одноразрядного слова состоит только из идентификатора. Так, описания *Z*, *ПП*, *ЗНАК* определяют три одноразрядных слова.

Совокупность слов, имеющих одинаковую длину, может объединяться в массив. Массив описывается в следующем виде:

$$M[m_1{:}m_2](n_1{:}n_2),$$