КОМП'ЮТЕРНІ ЗАСОБИ, МЕРЕЖІ ТА СИСТЕМИ

O.V. Petrishina

TECHNOLOGY OF REALIZATION OF UNIVERSAL PLURAL ACCESS

Technology of realization of universal plural access is offered to the general environment of data communication which is based on the stack mechanism of anio. Thus on the basis of the conducted analysis of existent access methods, an universal adaptive stack algorithm allowing to attain the maximum productivity in networks with plural access to the general environment of data communication is offered.

Предлагается технология реализации универсального множественного доступа к общей среде передачи данных, которая основана на стековом механизме anio. При этом на основе проведенного анализа существующих методов доступа предложен универсальный адаптивный стековый алгоритм, позволяющий достичь предельной производительности в сетях с множественным доступом к общей среде передачи данных.

УДК 681.324

О.В. ПЕТРИШИНА

ТЕХНОЛОГИЯ РЕАЛИЗАЦИИ УНИВЕРСАЛЬНОГО МНОЖЕСТВЕННОГО ДОСТУПА

Введение. В Институте кибернетики им. В.М. Глушкова НАН Украины была разработана технология универсального множественного доступа с целью повышения эффективной производительности на канальном уровне, а также для динамической оптимизации взаимодействия узлов локальной сети. Основу данной технологии составляет адаптивный стековый алгоритм anio, который является надстройкой над классическим методом доступа CSMA/CD. Этот алгоритм используется для улучшения характеристик дисциплины обслуживания стохастических запросов [1]. Он построен таким образом, что не требует разработки новой элементной базы и позволяет реализовать его в уже существующих сетях Ethernet [2]. Практическая реализация данной технологии позволит достичь граничной производительности.

Постановка задачи. Для упрощения задачи стандартизации в процессе интеграции алгоритма, необходимо чтобы программная модель пересылки сообщений между компьютерами на базе операционной системы семейства Windows NT соответствовала семиуровневой модели соединения открытых систем, международной организации по стандартизации (Open System Interconnection reference model, OSI ISO) [3, 4]. В официальной терминологии модели OSI группа бит, посылаемая канальным уровнем, называется Physical Layer Service Data Unit (единица данных, обслуживаемая физическим уровнем). На практике эту группу бит называют

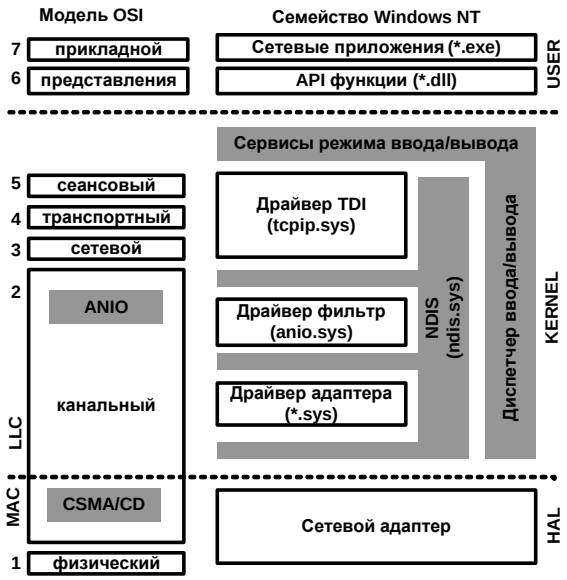

РИС. 1. Расположение алгоритма ANIO относительно Windows NT и OSI

фреймом. Внедрение алгоритма anio должно быть выполнено на канальном уровне (рис. 1). Для практической реализации и внедрения алгоритма anio в стек протоколов сетевой архитектуры семейства Windows NT [5] на канальном уровне необходимо использовать NDIS (Network Driver Interface Specification) спецификацию интерфейса сетевых драйверов, разработанную совместно Microsoft и 3Com для локальных вычислительных сетей [6]. NDIS скрывает от разработчика особенности реализации сетевого адаптера и сетевой операционной системы. Алгоритм anio является надстройкой над CSMA/CD, расположенным на подуровне MAC канального уровня, где выполняется пересылка фреймов, ожидание подтверждения их получения и повторение передачи потерянных фреймов. Следовательно, алгоритм anio должен работать на подуровне LLC над подуровнем MAC, канального уровня и выполнять функции управления доступом к среде передачи данных.

Программное внедрение данной технологии требует затраты аппаратных и программных ресурсов. Для записи статистической информации создается поток LOG, а для организации стекового механизма – ANIO. Для обслуживания этих потоков требуются затраты процессорного времени. Программная реализация стека ANIO обусловливает получения MAC адреса узла, осуществившего передачу в реальном масштабе времени. Для этого сетевой адаптер должен быть переведен в режим PROMISCUOS. Этот режим предполагает программную обработку большого количества пакетов, которые сетевой адаптер получает из сети, что требует дополнительных ресурсов системы. В результате работы стек должен выдавать информацию о том, что наступила очередь передачи пакета (для реализации доступа к среде передачи данных по данному алгоритму). До наступления очереди передачи пакеты, готовые к отправке нужно хранить в буфере, для которого необходимо выделить память. Все это может привести к снижению производительности. Поэтому в данной работе рассматриваются технология и алгоритмы реализации универсального множественного доступа, обеспечивающие рациональное использование системных ресурсов.

Состояние проблемы. Для изучения характеристик программной реализации алгоритма и их влияния на производительность всей системы, эффективности и целесообразности применения в процессе обмена по сети необходимо про-

вести исследования. Для этого была разработана инструментальная среда [7], позволяющая осуществлять разработку и отладку сетевого программного обеспечения, исследовать характеристики, влияющие на производительность операционной системы и сетевого адаптера, выполнять работы по оптимизации программной реализации алгоритма, а также выявлять недочеты самого алгоритма.

Методы решения задачи. Исходя из описания алгоритма anio [8], предполагается, что каждый узел сети включает унифицированный стековый механизм. Общее количество ячеек N в стеке равно максимальному количеству узлов в конкретной сети. $N = \{n_1, n_2, ..., n_j, ..., n_{\langle N \rangle}\}$ – множество станций, допускаемых методом доступа в канал CSMA/CD для работы в локальной сети. Пусть $M^0 = \overline{2, \langle N \rangle}$ – количество станций в локальной сети, где $\langle N \rangle$ – количество элементов множества *N*. Положим, что $M = \overline{2, \langle M^0 \rangle}$ – множество станций, подключенных в локальную сеть. Тогда $M_1 = \{m_1^1, m_1^2, ..., m_1^{i_1}, ... m_1^{\langle M_1 \rangle}\}$ – множество активных станций, имеющих данные для передачи; $M_1 = \overline{2, M}$ – количество элементов множества M_1; $M_2 = \{m_2^1, m_2^2, ..., m_2^{i_2}, ... m_2^{\langle M_2 \rangle}\}$ – множество неактивных станций, включенных в сеть, работающих в автономном режиме, где $M_2 = \overline{0, M - \langle M_1 \rangle}$ – количество элементов множества M_2; $M_3 = \{m_3^1, m_3^2, ..., m_3^{i_3}, ... m_3^{\langle M_3 \rangle}\}$ – множество пассивных станций, не включенных в сеть; $M_3 = \overline{0, M - \langle M_1 \rangle + \langle M_2 \rangle}$ – количество элементов множества M_3; $M_4 = \{m_4^1, m_4^2, ..., m_4^{i_4}, ... m_4^{\langle M_4 \rangle}\}$ – множество станций, которые в процессе информационного обмена завершают работу в сети и переходят из режима M_1 в режим станций множества M_2 или M_3; $M_5 = \{m_5^1, m_5^2, ..., m_5^{i_5}, ... m_5^{\langle M_5 \rangle}\}$ – множество станций, которые начинают принимать участие в процессе информационного обмена и переходят из режимов множества станций M_2 или M_3 в режим множества станций M_1. Функционирование стека реализуется в трех режимах: начальный режим формирования стека, стационарный режим, рабочий режим.

Исходя из общего принципа множественного доступа к общей среде передачи данных по алгоритму anio, разработан принцип организации работы алгоритма относительно каждой отдельно взятой станции, которая осуществляет обмен данными по данному алгоритму. Это позволит изучить процессы, происходящие в каждой станции во время передачи данных, и разработать программную реализацию протокола доступа к каналу по алгоритму anio. Так как алгоритм anio работает на канальном уровне, то в качестве адреса станции необходимо брать MAC адрес, размер которого 6 байт. Исходя из этого, для программной реализации алгоритма, разработана структура элемента стека и набор функций для работы с ним.

Входными данными для стека является MAC адрес. Первый полученный MAC адрес интерпретируется как MAC адрес локального компьютера, помещается в стек и никогда не удаляется. Все последующие MAC адреса – MAC адреса узлов, получивших доступ к среде передачи данных в данный момент времени. Если MAC = NULL то это означает, что произошел таймаут. В таблице представлены функции работы со стеком, которые должен поддерживать алгоритм. Необходимо также отметить наличие двух режимов работы: активный (А) и неактивный (Н). Активный режим предполагает, что MAC адрес локального узла находится в стеке в любой позиции, и как только он оказывается на дне стека, разрешается передача пакета в сеть. В неактивном режиме MAC адрес локального узла все время находится на дне стека и в любой момент времени разрешена передача пакета в сеть. Как только пакет будет передан в сеть, станция переходит в активный режим. При переходе станции из множества M_3 пассивного режима в активный у станции постепенно сформируется полный стек. При этом станция принадлежит множеству M_5. Станции, принадлежащие множеству M_5, постоянно инициируют внеочередную передачу данных. Как только у такой станции будет полностью сформирован стек, она перейдет из множества станций M_5 в множество станций M_1.

ТАБЛИЦА. Функции работы со стеком anio

Режим	Событие	Функции				Пример относительно станции «9»			Прирост элементов	Смена режима
		Ожидать	Добавить	Сравнить	Удалить	Ожидать	Добавить и сравнить	Удалить		
активный	Таймаут (А→Н)	+	–	–	–	5 1 3 9	5 1 3 9	5 1 3 [9]	0	А → Н
	Таймаут (А→А)	+	–	–	+	1 3 9 5	1 3 9 5	1 3 9	–1	А → А
	Очередной	+	+	+	+	5 1 3 9	9 5 1 3 9	9 5 1 3	0	А → А
	Внеочередной	+	+	+	+	5 1 3 9	3 5 1 3 9	3 5 1 9	0	А → А
	Новый	+	+	+	–	5 1 3 9	7 5 1 3 9	7 5 1 3 9	+1	А → А
неактивный	Таймаут	+	–	–	+	5 1 3 [9]	5 1 3 [9]	5 1 [9]	–1	Н → Н
	Очередной	+	+	+	+	5 1 3 [9]	3 5 1 3 [9]	3 5 1 [9]	0	Н → Н
	Внеочередной (Н→А)	+	+	+	+	5 1 3 [9]	9 5 1 3 [9]	9 5 1 3	0	Н → А
	Внеочередной (Н→Н)	+	+	+	+	5 1 3 [9]	1 5 1 3 [9]	1 5 3 [9]	0	Н → Н
	Новый	+	+	+	–	5 1 3 [9]	7 5 1 3 [9]	7 5 1 3 [9]	+1	Н → Н

Если станция принадлежит множеству M_1, т.е. находится в активном режиме, она выполняет вращение стека и участвует в обмене в соответствии с очередностью в стеке. В случае, если у станции нет данных для передачи в сеть, она пропускает собственную передачу, но вместо того чтобы удалить свой адрес из стека, она переходит к множеству станций M_2 в неактивный режим. В неактивном режиме во время вращения стека станция удерживает собственный адрес на дне стека и поддерживает в актуальном состоянии содержимое стека.

Как только у станции появляются данные для передачи, она может перейти из множества M_2 к множеству M_1 и начать передачу данных. Такая станция не будет инициировать внеочередную передачу.

В пассивном режиме станция не имеет возможности поддерживать в актуальном режиме содержимое стека, и в случае необходимости передачи данных, имеет право внеочередной передачи. Стек anio постоянно находиться в режиме ожидания захвата канала (рис. 2, а).

Процесс добавления нового элемента в стек сводится к добавлению на вершину стека адреса станции, получившей доступ к среде передачи данных (рис. 2, г). Далее выполняется сравнение с остальными элементами стека, что дает возможность убедиться – появилась ли новая станция. В случае, если в стеке отсутствует такой адрес, удаление из стека не осуществляется (рис. 2, е). Это позволяет увеличить количество элементов в стеке на единицу.

Процесс удаления элемента из стека возможен только в случае наступления таймаута и сводится к сравнению собственного адреса станции с элементом на дне стека, чтобы исключить возможность удаления из стека собственного адреса. Если на дне стека обнаружен собственный адрес, удаление из стека не осуществляется, и станция переходит из активного режима в неактивный. В остальных случаях элемент, расположенный на дне стека, удаляется (рис. 2, б). Это позволяет уменьшить количество элементов в стеке на единицу.

Процесс вращения стека (движение очереди) сводится к добавлению на вершину стека адреса станции, получившей доступ к среде передачи данных.

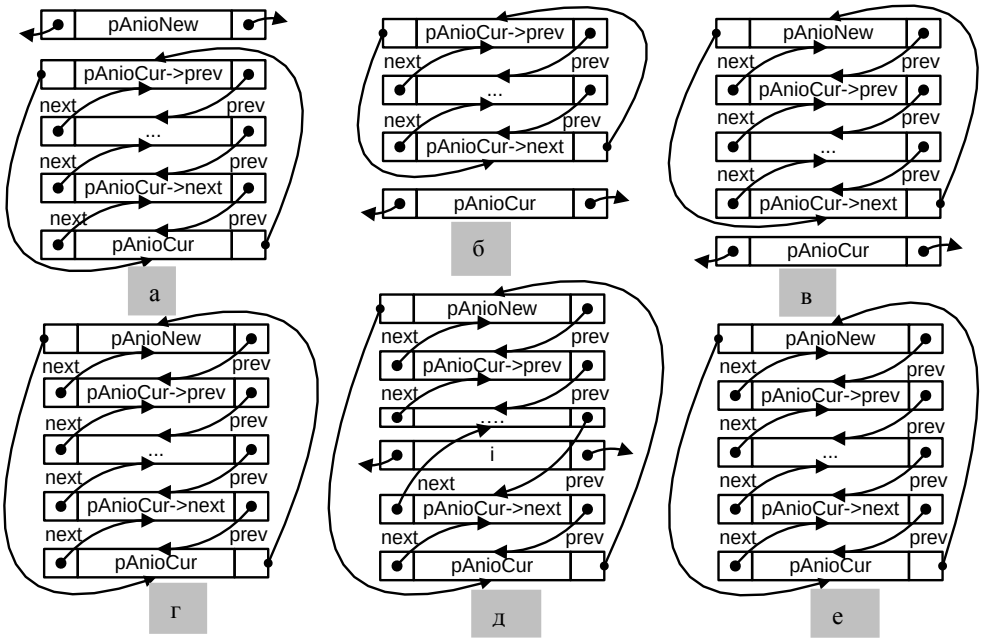

РИС. 2. Режимы работы стека anio: а – ожидание; б – таймаут; в – очередной; г – передача; д – внеочередной; е – новый

Далее выполняется сравнение с остальными элементами стека. Это дает возможность убедиться в том, что очередная (рис. 2, в) или внеочередная (рис. 2, д) станция получила доступ к каналу и начала передачу. Затем найденный элемент со дна стека или с i-й позиции удаляется из стека, чтобы не допустить повторение адресов. Это позволяет сделать один шаг очереди. Тогда количество адресов в стеке не изменяется.

Следует отметить, что базовые операции работы со стеком избыточны. Для организации стекового механизма anio [8] достаточно всего две операции сдвига АСДЧ и АСДП. Поскольку АСДП является частным случаем АСДЧ (где i=1), то можно ограничиться единственной операцией сдвига АСДЧ, которая необходима для удаления i-го элемента из стека. Событие разрешения передачи наступает в момент, когда на дне стека обнаружен собственный адрес.

Заключение. Изучен и описан стековый механизм с точки зрения программной реализации на станции, которая будет работать по данному алгоритму. Проведён анализ и определено, что стековый алгоритм anio работает на канальном уровне. Поэтому его следует внедрить в операционную среду на базе операционной системы семейства Windows NT, на уровне ядра в виде драйвера-фильтра, который будет взаимодействовать с NDIS (ndis.sys). Разработан программно-ориентированный стек, который может быть программно реализован в виде отдельного класса и внедрен в драйвер-фильтр. Драйвер фильтр должен обеспечить стек информацией о MAC адресах станций, получивших в данный момент доступ к среде передачи данных или сигнал наступления таймаута. Стек должен информировать драйвер-фильтр о наступлении очереди передачи. Драйвер-фильтр должен обеспечить возможность управления отправкой пакетов в сеть (задержка передачи и разрешение передачи). Для минимизации задержек в работе и оптимизации использования ресурсов в процессе внедрения стекового механизма anio в операционную среду была разработана и апробирована оригинальная реализация технологии универсального множественного доступа.

1. *Алишов Н.И.* Универсальный метод CSMA/CD // Тр. Междунар. конф. «Локальные вычислительные сети». – Рига, 1990. – С. 322 – 325.
2. *Шварц М.* Сети связи // Протоколы, моделирование и анализ. Часть II. – М.: Наука, 1992. – С. 53 – 63.
3. *Олифер В.Г., Олифер Н.А.* Компьютерные сети // Принципы, технологии, протоколы. – СПб.: Питер, 2002. – С. 67 – 79.
4. *Шайберг С.* Недокументированные возможности Windows 2000 // Библиотека программиста. – СПб.: Питер, 2002. – С. 146 –181.
5. *Сорокина С.И., Тихонов А.Ю., Щербаков А.Ю.* Программирование драйверов и систем безопасности: Учеб. пособие. – СПб.: БХВ-Петербург; М.: Издатель Молгачева С.В., 2002. – С. 161–167.
6. Текущая версия NDIS 3.0 ftp://ftp.microsoft.com/misc/ndis.doc
7. *Петрішина О.В.* Інструментальне середовище дослідження множинного доступу в локальних мережах // Вісті академії інженерних наук України.– 2005.– № 4 (27).– С. 38 – 40.
8. *Алишов Н.И.* Адаптивный стековый алгоритм универсального множественного доступа в распределенных системах и сетях компьютеров // УСиМ. – 2004. – № 2. – С. 59 – 72.

Получено 23.02.2006