АКАДЕМИЯ НАУК СССР
СИБИРСКОЕ ОТДЕЛЕНИЕ
АВТОМЕТРИЯ

№ 5 1978

МАШИННАЯ ГРАФИКА

УДК 681.3.06 : 389.6

Ю. М. БАЯКОВСКИЙ, В. А. ГАЛАКТИОНОВ
(Москва)

ГРАФИЧЕСКИЕ ПРОТОКОЛЫ
(Обзор)

1. **Введение.** Если проследить историю развития машинной графики, то можно выделить несколько узловых этапов. На ранней стадии главное внимание уделялось разработке графических устройств. Несколько позднее самой актуальной стала проблема создания программного обеспечения. Затем была осознана необходимость конструирования алгоритмов и программ, которые можно применять в различных системах и с разными графическими устройствами. И, наконец, с появлением сетей ЭВМ на первый план выдвинулась проблема разработки графических протоколов, т. е., по существу, проблема стандартизации.

В отличие от многочисленных применений вычислительных машин в различных областях науки, техники и экономики, где использование таких машинно-независимых языков программирования, как АЛГОЛ, ФОРТРАН, КОБОЛ и др., позволило создать обширные библиотеки прикладных программ, практически каждое новое приложение интерактивных графических систем требует почти полного перепрограммирования. Графические системы оказываются, как правило, несовместимыми друг с другом.

Графические терминалы, кроме того, оснащаются неодинаковым числом устройств ввода-вывода: одни обладают почти полным набором устройств, в то время как другие имеют лишь одно или два (например, световое перо и клавиатуру или только одно световое перо). Это создает дополнительные трудности для тех пользователей графической системы, которые хотели бы работать со своими программами с терминалов, имеющих разные комплексы физических устройств. Появляется настоятельная потребность представить все устройства, с помощью которых пользователь осуществляет взаимодействие с системой, и их характеристики некоторым стандартным образом. В результате можно получить достаточно гибкий аппарат эквивалентной замены одного физического устройства другим без изменения существа программы.

Необходимость создания средств, позволяющих пользоваться различными графическими системами с помощью терминалов разного типа, стала особенно безотлагательной с появлением сетей ЭВМ. Возникновение сетей (в частности, сети ARPANET в США) поставило перед программистами целый ряд новых задач: нахождение оптимального

способа распределения нагрузки между локальным графическим процессором и обслуживающей ЭВМ, организация обмена информацией между ними для достижения наилучшего распределения ресурсов, синхронизация параллельных процессов и др.

В настоящем обзоре рассматриваются предпосылки для создания графических протоколов, текущее состояние дел в этой области машинной графики и перспективы ее развития.

2. **Предпосылки и пути стандартизации.** За последние 10—15 лет были исследованы (как умозрительно, так и экспериментально) различные подходы к решению проблемы независимости программного обеспечения от особенностей конкретных устройств. Предпринимались попытки использовать машинную независимость универсальных языков программирования: ФОРТРАНа, АЛГОЛа, ПЛ-1. Однако, как отмечается в [1], расширения существующих языков программирования, в которые графические элементы включаются в качестве новых операторов, функций и т. д., так и не получили широкого распространения.

Для достижения независимости от конкретных устройств большинство систем машинной графики реализуется в виде так называемой «перевернутой пирамиды» [2]. Выходные данные в таких системах являются общим интерфейсом со всеми графическими устройствами (рис. 1).

При этом в основу системы положен ограниченный набор элементарных функций (или программ), достаточных для генерации любого изображения. Набор таких функций может включать в себя, например, следующие: переместить луч, изобразить точку, провести отрезок, изобразить текст.

Разумеется, это не минимальный набор функций, так как точки и текст могут быть легко изображены с помощью двух оставшихся функций. Тем не менее вследствие того, что большинство графических дисплеев имеет аппаратные генераторы точек и символов, изображения этих элементов включаются как основные, базисные функции.

Стремление сузить нижний уровень, т. е. положить в основу минимальное количество программ нижнего уровня, приводило также к попыткам создать транспортабельный машинно-независимый генератор символов путем представления символов на языке высокого уровня. Одна из попыток такого рода — генератор символов SIGCHR [3], написанный на ФОРТРАНе.

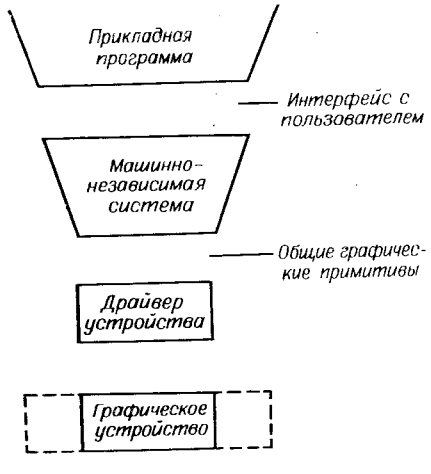

Рис. 1. Типичная организация графической системы.
(Штриховыми линиями указаны те потенциальные возможности устройства, которые не могут быть использованы в рамках данной системы).

Предлагалось, кроме того, стандартизовать сами программы нижнего уровня (и соответственно их имена).

Некоторые системы, в которых используется принцип «перевернутой пирамиды» (например, ГРАФОР [4] или GINO-F[5]), получили довольно широкое распространение и успешно применяются для сравнительно простых устройств типа графопостроителей, дисплеев на трубке с памятью и простейших дисплеев с регенерацией изображения.

Существенный недостаток всех этих подходов в том, что они сильно ограничивают использование разнообразных средств ввода-вывода, которые могут быть реализованы аппаратно (в жертву стандартизации приносится разнообразие). Такая

стандартизация тормозит развитие средств графического ввода-вывода.

Виртуальные устройства. Один из подходов к решению проблемы стандартизации устройств ввода-вывода заключается в том, что все графические устройства программно моделируются небольшим числом виртуальных устройств. Каждое из этих виртуальных устройств может быть легко реализовано соответствующими физическими устройствами, и, наоборот, каждое физическое устройство можно рассматривать как реализацию одного или нескольких типов виртуальных устройств.

Введение виртуальных устройств, в частности, становится совершенно необходимым, когда одну и ту же программу обработки графической информации приходится выполнять на терминалах, снабженных разными физическими устройствами. И здесь метод, состоящий в написании прикладных программ в терминах элементарных устройств и затем в использовании небольшого числа стандартных подпрограмм для отображения этих виртуальных устройств на физические, дает ряд значительных преимуществ.

Идея виртуального графического устройства была высказана У. Ньюменом [6], который разбил все способы ввода информации на семь категорий: текстовые строки с использованием символа возврата каретки и без него, десятичные и восьмеричные оценщики, указки, локаторы и кнопки клавиатуры.

Дальнейшее развитие принцип виртуальных устройств получил в работах И. Коттон [7], которая предложила классификацию устройств ввода применительно к сетям ЭВМ; Дж. Фоли и В. Уоллеса [8], предложивших в качестве виртуальных устройств взять следующие четыре типа: указку, кнопку, локатор и оценщик; Р. Спрулла и Э. Томаса [9] (см. п. 5); В. Уоллеса [10] и др. Представляет интерес также работа [11], в которой введен особый виртуальный терминал и разрабатывается протокол для управления этим терминалом в составе сети ЭВМ.

3. **Графические протоколы.** Основные результаты по стандартизации в области интерактивной машинной графики были получены применительно к ARPA-сети в США. Главные усилия были направлены на разработку системы графических протоколов для машинно-независимого представления графических данных (рис. 2).

По сети данные передаются только в стандартизованном виде, так что компьютер, посылающий некоторое сообщение, вообще говоря, не должен располагать какой-либо предварительной информацией о характеристиках компьютера, получающего это сообщение. Посылающий компьютер преобразует данные в стандартизованный формат, а получающий компьютер должен переводить из стандартизованного формата в форму, понятную для прикладной программы (за исключением тех случаев, когда прикладная программа в получающем

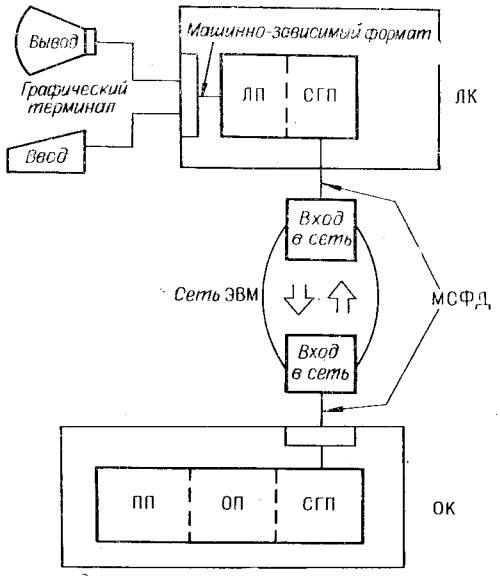

Рис. 2. Потоки данных с использованием сетевого графического протокола:
ЛП — локальная программа; СГП — сетевой графический протокол; ЛК — локальный компьютер; МСФД — машинно-независимый стандартизованный формат данных; ПП — прикладная программа; ОП — обслуживающая программа; ОК — обслуживающий компьютер.

компьютере может непосредственно интерпретировать стандартизованный формат). Во многих случаях это приводит к появлению специальных кодирующих и декодирующих программ (обозначенных СГП на рис. 2).

Для разработки графических протоколов в ARPA-сети была создана рабочая группа [1]. Ею, в частности, был решен такой фундаментальный вопрос, как полное отделение графического ввода информации от ее вывода (т. е. для устройства ввода должен быть определен отдельный независимый протокол).

Кроме того, было принято решение о том, что не будет одного всеобъемлющего протокола сразу для всех графических устройств. Он будет делиться на несколько уровней в соответствии со сложностью реализации функций, составляющих данный уровень [12, 13].

Протоколы нулевого уровня. Такие протоколы специально делались очень простыми, чтобы их могли реализовать не только все без исключения компьютеры (в том числе и самые маломощные), но даже и программируемые терминалы.

Протокол вывода нулевого уровня содержит только двенадцать команд [12, 13]. Три из них выполняют управляющие функции: «Пустая» команда, команды «Инициировать экран» и «Конец изображения». Шесть команд обеспечивают возможности позиционирования луча и изображения прямых линий и точек при абсолютном и относительном способах адресации. Две команды используются для изображения текстовых строк с возвратом луча в первоначальное положение и без него. И, наконец, последняя команда дает возможность перейти на код конкретного устройства.

Возможности протокола ввода нулевого уровня также очень ограничены. На нулевом уровне выделяются только два типа вводимых данных [14]: текстовая цепочка и простая позиция, выраженная в системе координат экрана.

4. **Протоколы вывода более высоких уровней.** Первоначально существовало мнение, что графические протоколы вывода должны строиться из произвольного числа уровней, надстраивающихся друг над другом [12, 13]. Такой подход, однако, оказался слишком (и неоправданно) сложным [1]. В действительности для протокола вывода требуется всего лишь два дополнительных уровня, которые соответствуют сегментированному и структурированному дисплейным файлам.

Различие между двумя этими уровнями заключается в способности терминала (или локального компьютера (ЛК), к которому подсоединен терминал) применять к дисплейному файлу такие преобразования, как масштабирование, поворот, сдвиг и т. д. Если терминал (или ЛК) недостаточно «разумен» и не может справиться с этими преобразованиями, то согласно протоколу он должен получать данные в сегментированном (не структурированном!) виде. Если же терминал (или ЛК) способен производить такие преобразования, то вместе с данными в ЛК может передаваться также и структура, описывающая логические зависимости между элементами изображения.

Сегментированный формат. Протокол, который соответствует этому уровню [9], используется для построения и модификации набора сегментов трансформированного (сегментированного) дисплейного файла, хранящихся в ЛК. Все преобразования координат производятся обслуживающей программой (ОП) в обслуживающем компьютере (ОК). Данные посылаются в ЛК уже в виде сегментов графических данных, состоящих из небольшого числа примитивов: точек, отрезков прямых линий и цепочек текста, каждый из которых имеет на экране дисплея некоторое свое фиксированное положение.

Каждый сегмент создается следующим образом. ОП посылает локальной программе (ЛП) в локальном компьютере специальную команду

«Открыть сегмент» и имя сегмента. Все графические примитивы, поступающие после этого в ЛП, добавляются в порядке поступления в этот «текущий» открытый сегмент. Закрывается сегмент командой «Закрыть сегмент».

С помощью команды «Включить» можно включить указанный сегмент в список сегментов, воспроизводящихся на экране. Команда «Выключить» используется для удаления сегмента из этого списка. Сегмент можно и полностью уничтожить с помощью команды «Ликвидировать». Команда «Продолжить» позволяет добавить графическую информацию в конец уже существующего сегмента.

Однако получение локальной программой перечисленных команд протокола отнюдь не влечет за собой немедленное изменение изображения на экране дисплея. Изображение остается неизменным до тех пор, пока не закончится вся совокупность модификаций и не поступит команда «Конец пакета изменений». И только после прихода этой команды изображение будет скорректировано в соответствии с поступившими изменениями (т. е. будут выполнены предыдущие команды «Включить», «Продолжить» и др.). Такой метод имеет целый ряд преимуществ по сравнению с мгновенной корректировкой изображения [9].

Непосредственно сразу за командой «Открыть сегмент» и именем сегмента может быть указано любое число атрибутов, которые будут связаны с новым сегментом. Это могут быть атрибуты выбора интенсивности линий, зрительного выделения, чувствительности сегмента к световому перу, выбора экрана (если в распоряжении пользователя их несколько), выбора начальной позиции и т. д. Несмотря на то что графические примитивы входящие в сегмент, не могут быть модифицированы, для атрибутов сегмента существует специальная команда «Изменить атрибут», с помощью которой может быть изменен любой атрибут сегмента.

Одно из основных преимуществ использования трансформированного формата заключается в том, что можно обойтись довольно простой локальной программой, так как ЛК не должен выполнять никаких сложных геометрических преобразований. В этом случае вся нагрузка перекладывается на ОК.

Структурированный формат. Протокол вывода структурированного формата аналогичен методу «групп и элементов» [15]. Он прежде всего предназначается для высокопроизводительных дисплеев с аппаратным выполнением преобразований, т. е. для дисплеев, способных интерпретировать структурированный дисплейный файл. Однако если терминал (или ЛК пользователя) не обладает всеми необходимыми аппаратными возможностями, вполне допустимо и их программное моделирование.

Согласно протоколу изображение состоит из «фигур», каждая из которых содержит произвольное число «элементов». Элементы могут представлять собой либо список графических примитивов (отрезков прямых линий, точек и текста), производящих изображение на экране дисплея, либо «обращения» к другим фигурам, включая такие геометрические преобразования «вызванной» фигуры, как поворот, сдвиг, масштабирование и т. д. Например, изображение диаграммы электрической цепи может быть построено с помощью фигуры, составленной из отрезков прямых линий, образующих резистор, и элементов, представляющих собой «обращения» к этой фигуре для каждого резистора цепи. Все преобразования координат и разности в интенсивностях определяются как параметры этих элементов-обращений.

Протоколы для структурированного формата еще окончательно не разработаны [9, 13, 16], и все вышеизложенное представляет собой лишь некоторые предварительные соображения на этот счет. Объясняется это главным образом трудностями разработки приемлемого

машинно-независимого способа проведения геометрических преобразований.

Протокол для представления текстовой информации. Уже описанные графические средства могут быть использованы также и для изображения текстовой информации на экране дисплея.

Средства работы с текстовой информацией были отделены от остального графического обеспечения в основном по двум причинам [9]:

а) пользователи, имеющие простые алфавитно-цифровые дисплеи (типа «Videoton»), часто способны реализовать у себя специальный протокол для работы с текстовой информацией, в то время как на реализацию полного графического протокола у них не хватает мощности;

б) специальный протокол для работы с текстами значительно упрощает разработку тех локальных программ, которые нуждаются прежде всего в обеспечении текстового интерфейса.

Поэтому графический текстовой протокол вывода полностью не зависит от трансформированного и структурированного форматов.

Протокол основывается на том, что ОП с помощью ЛП может создать текстовой сегмент — текстовое «окно» на экране, которое представляет собой прямоугольную область, в которой могут располагаться строки текста. ОП с помощью команд протокола может различным образом редактировать этот текст: заменять символы, перемещать строки и т. д., при этом все изменения, вносимые в текст командами протокола, должны немедленно появляться на экране.

5. **Протоколы ввода более высоких уровней.** Проблема обеспечения ввода графической информации является значительно более сложной, чем задачи графического вывода. Поэтому ввод информации — наиболее противоречивая и экспериментальная часть протокола.

Отметим сразу, что протокол детально не разрабатывает средства ввода графической информации. Подробная разработка этих средств предоставляется программистам ОП. Основная причина этого — в глубоком различии между операционными системами и аппаратурой графических устройств. Кроме того, пользователи имеют разные склонности и могут предпочитать, вообще говоря, слегка различающиеся манеры взаимодействия с устройствами. Протокол оставляет ЛП некоторую стилистическую гибкость, которая может быть использована программистом.

Однако несмотря на это, протокол четко определяет цель и назначение каждого устройства ввода. ЛП должна точно придерживаться буквы протокола, хотя детали могут варьироваться (протокол как бы определяет «русло», в котором нужно работать).

Протокол дает возможность ОП использовать два основных метода для работы с устройствами ввода:

1) считывание состояния устройства по требованию (например, текущего положения курсора в случае координатного устройства ввода);

2) использование интерактивных методов, называемых «событиями»; активация этих событий и получение «сообщений» о результатах действий пользователя.

Считывание состояния устройств ввода. В протоколе ввода [9] вопрос машинной независимости решается с помощью опроса. ЛП сообщает ОП список имеющихся в ее распоряжении устройств и ОП совместно с прикладной программой (ПП) (обе находятся в ОК!) принимает решение о том наборе устройств, который необходим для функционирования ПП.

Исходя из протокола ввода, каждое вводное устройство можно рассматривать как физическую реализацию одного или нескольких типов виртуальных устройств ввода (см. п. 2). Протокол предлагает классификацию, в основе которой лежат пять типов:

1. Координатные устройства. Состояние этих устройств определяется парой координат в системе координат экрана. Эти координаты могут быть получены с планшета, с алфавитно-цифровой клавиатуры, светового пера или откуда-либо еще.

2. Линейные устройства. Это устройства типа различных рукояток и реостатов. Результатом их работы является дробь в диапазоне от 0 до 1, которая и передается в ОП.

3. Функциональные кнопки. Эти устройства могут представлять собой как одну кнопку (включена/выключена), так и совокупность кнопок, образующих целый набор кодовых комбинаций.

4. Алфавитно-цифровая клавиатура. Это устройство дает возможность вводить символы из некоторого стандартного набора (например, ASCII [9] или IA5 [11]). Состояние клавиатуры — это код символа, который был нажат последним, или нуль, если этот символ был уже опрошен (введен).

5. Время. Если ЛП имеет в своем распоряжении устройство ввода «время», то она может сообщать ОП измерения текущего времени. Протокол не определяет детально способ измерения времени, за исключением того, что интервалы между измерениями должны быть не больше 10—100 мс.

В протоколе предусмотрена команда, с помощью которой ОП может запросить состояние сразу целого ряда вводных устройств. Эти состояния должны быть считаны (почти одновременно) и переданы назад в ОП.

События. Кроме проблемы машинной независимости при проектировании графических систем (особенно сетевых), большие трудности вызывают и некоторые эксплуатационные вопросы.

Одна из проблем такого рода — в следующем: если данные с каждого устройства ввода будут передаваться в обслуживающий компьютер для обработки их ПП и ОП, то все изменения в изображении будут поступать из обслуживающего в локальный компьютер и обрабатываться ЛП. И только после этого пользователь сможет увидеть на экране результаты своей деятельности. Таким образом, очень трудно (а порой невозможно) применить многие интерактивные графические методы.

Если же ОП будет иметь возможность «поручать» ЛП самой использовать в совокупности с устройствами ввода некоторые интерактивные методы и сообщать ОП только о результатах взаимодействия, то эту проблему можно обойти. Назовем такие методы «событиями».

Протокол ввода выделяет следующие виды событий:

1. События, связанные с алфавитно-цифровой клавиатурой. Эти события возникают при нажатии на клавишу алфавитно-цифровой клавиатуры. «Сообщение» для этого события — код нажатой клавиши.

2. События, связанные с функциональными кнопками. «Сообщение» об этом событии поступает в ОП в том случае, когда включается или выключается функциональная кнопка. Это «сообщение» несет ту же информацию, что и в предыдущем случае.

3. События, связанные с линейными устройствами. Эти события происходят, когда показания какого-либо линейного устройства превышают определенную пороговую величину. Более детально это событие определяет сама ЛП.

4. Позиционирование. С помощью координатного (или какого-либо другого) устройства находится некоторое положение на экране или его координаты. Например, пользователь может указать позицию простым нажатием карандаша графического планшета. Часто ЛП для облегчения позиционирования использует следящее перекрестие. Детали этого события также определяются ЛП.

5. Указывание. Это событие заключается в том, что координатное (или какое-либо другое) устройство используется для идентификации

некоторого сегмента, фигуры (в случае структурного описания изображения) или участка текста, которые изображены на экране и сделаны чувствительными к световому перу. Пользователь таким образом получает возможность непосредственно указывать объекты на экране. ОП передается имя указанного объекта и координаты точки на экране, в которой произошло событие. ЛП может использовать для указывания целый ряд разнообразных устройств и методов.

6. «Рисование». Координатное (или какое-либо другое) устройство используется для вычерчивания произвольной кривой. ЛП обеспечивает слежение, оставляет след позади курсора (состоящий из точек или мелких штрихов, соединенных друг с другом) и сообщает ОП координаты точек вдоль кривой, снимаемые с определенным шагом. Детальное описание события остается за ЛП.

7. Многократное «рисование». Это событие полностью аналогично предыдущему, за исключением того, что оно используется для записи сразу целой последовательности кривых линий. Многократное «рисование» очень удобно для реализации метода распознавания символов в режиме «on line» (см., например, [15]).

8. Перемещение. В этом случае используется координатное устройство для перемещения различных элементов изображения по экрану дисплея без какого-либо вмешательства со стороны ОП. Этот метод можно назвать «высокоинтерактивным», и далеко не каждый графический терминал имеет возможность реализовать его.

9. События, связанные с замыканием и размыканием выключателя светового карандаша. Это частные случаи позиционирующих событий, которые могут иметь значение в некоторых приложениях. Замыкание происходит, когда карандаш касается поверхности планшета, размыкание — когда касания нет.

ЛП в ответе на запрос сообщает ОП об имеющихся в ее распоряжении событиях. При этом каждое из этих событий должно быть «разрешено» (демаскировано) прежде, чем оно будет регистрироваться.

Предусмотрены команды, выдаваемые ОП для маскирования и демаскирования событий. Если событие замаскировано, ЛП может полностью игнорировать соответствующее устройство (т. е. нет необходимости в обработке событий, возникающих на неактивированных устройствах).

О некоторых достоинствах протоколов. Протокол оставляет пользователю широкую свободу действий по реализации средств графического ввода. Терминал может иметь произвольные (как максимальные, так и минимальные наборы устройств и событий, поскольку о наличии тех или иных средств ввода ОП легко может узнать при помощи опроса. Такая гибкость протокола позволяет на терминалах с недостаточно развитым аппаратным обеспечением моделировать недостающие вводные устройства с помощью имеющихся устройств.

Таким образом, немаловажным преимуществом использования графических протоколов является легкость, с которой устройства ввода могут заменять друг друга в прикладных задачах, имитировать различные функции других устройств и расширять свои собственные возможности [1, 7]. Вообще говоря, с помощью каждого выделенного в протоколе устройства ввода можно моделировать любое из оставшихся устройств [7]. Например, даже такое простое устройство ввода, как алфавитно-цифровая клавиатура, можно использовать для моделирования сложных устройств: планшета или светового пера; используя световое перо или планшет, можно легко имитировать действие любой клавиатуры, в том числе и алфавитно-цифровой [7, 8, 15].

Кроме того, наличие стандартного протокола ввода значительно облегчает включение в систему новых устройств ввода (для этого достаточно добавить лишь небольшой модуль).

6. **Реализация протоколов.** Хотя описание протокола занимает довольно много места, сам он очень прост. Протокол разрабатывался таким образом, чтобы ЛП могла быть реализована практически на любом «разумном» терминале и, конечно, на любой ЭВМ, соединенной с графическим устройством.

Относительная простота реализации протокола стала возможной по нескольким причинам. Одна из них заключается в том, что значительная часть команд протокола не является обязательной для реализации во всех графических системах. Для большинства приложений вполне достаточно и некоторого подмножества протокола. На сложных и дорогих моделях графических терминалов протокол реализуется в полном объеме, в то время как на простых и дешевых устройствах он может быть очень простым.

Кроме того, в сложных и противоречивых областях (например, ввода информации) протокол умышленно не детализируется, позволяя ЛП распорядиться имеющимися у нее возможностями наилучшим образом и предоставляя пользователю широкую свободу действий.

Использование машинно-независимых графических протоколов, однако, имеет и свои теневые стороны [16]. Так, при выводе результатов вычислений на экран дисплея приходится решать две дополнительные задачи по «перекодировке» (генерация протокола в ОК и его интерпретация в ЛК (см. рис. 2)), в то время как в одномашинной графической системе это всего лишь совокупность обращений к стандартным подпрограммам. Такая дополнительная работа часто приводит к непроизводительным затратам машинного времени.

Более того, использование схемы графического протокола требует от ЛК способности интерпретировать протокол, генерировать код для дисплейного процессора и, кроме того, производить некоторые другие действия: масштабирование нормализованных координат экрана и т. д. (обычно в реальном времени). Все это может привести к перегрузке, как правило, небольших по мощности ЛК.

В заключение отметим, что так как сложность организации структур сетей очень высока и, кроме того, в сетях используется большое число различных по своим возможностям типов графических устройств, было бы наивным считать, что хорошие модели протоколов могут быть разработаны достаточно быстро [11]. На самом деле это длительный итеративный процесс: разработка — внедрение — новая разработка с учетом полученного опыта.

Авторы с благодарностью отмечают, что значительная часть материалов, использованных в статье, была любезно предоставлена Информационным центром по сетям ЭВМ в Стенфордском исследовательском институте.

ЛИТЕРАТУРА

1. **Cotton I. W.** Standards for network graphics communications.— "Computers and Graphics", 1975, vol. 1, N 1, p. 45—47.
2. **Bergernn R. D.** Picture primitives in device independent graphics systems.— "Computers and Graphics", 1976, vol. 10, N 1, p. 57—60.
3. **Wright T.** SIGCHR — a portable character generator.— "Computers and Graphics", 1976, vol. 10, N 4, p. 18—34.
4. **Баяковский Ю. М., Лазутин Ю. М., Михайлова Т. Н., Мишакова С. Т.** ГРАФОР: комплекс графических программ на ФОРТРАНе. Вып. 5. Структура и основные принципы.— Препринт № 90. М., изд. ИПМ АН СССР, 1975.
5. **Woodsford P. A.** The design and implementation of the GINO 3D graphics software package.— "SOFTWARE-Practice and Experience", 1971, vol. 1, p. 335—365.
6. **Newman W. M.** A system for interactive graphical programming.— In: Proc. AFIPS 1968 SJCC. Vol. 32. AFIPS Press, Montvale, N. J., 1968, p. 47—54.

7. **Cotton I. W.** Network graphic attention handling.— In: ONLINE72 Conf. Proc. Vol. 2. Brunel Univ., Uxbridge, England, Sept. 1972, p. 465—490.
8. **Foley J. D., Wallace W. L.** The art of natural graphic man-machine conversation.— "Proc. IEEE", 1974, vol. 62, N 4, p. 462—471.
9. **Sproull R. F., Thomas E. L.** A network graphics protocol.— In: Techn. Report, ARPA Network, NIC 24308, August 16, 1974.
10. **Wallace V. L.** The semantics of graphic input devices.— "Computers and Graphics", 1976, vol. 10, N 1, p. 61—65.
11. **Schicker R., Duenki A.** Virtual terminal definition and protocol.— "Comput. Commun. Rev.", 1976, vol. 6, N 4, p. 1—18.
12. **Michener J., Cotton I., Kelley K., Liddle D., Meyer E.** Graphics protocol — level 0 only.— In: Network working group, ARPA Network; RFC: N 292; NIC 8302 (12 Jan. 1972).
13. **Michener J., Cotton I., Kelley K., Liddle D., Meyer E.** Graphics protocol.— In: Network Working Group, ARPA Network; RFC: N 493; NIC 15358 (April 26, 1973).
14. **Cotton I. W.** Level 0 graphic input protocol.— In: Network Working Group, ARPA Network; RFC: N 336; NIC 9929 (5 May 1972).
15. **Ньюмен У., Спрулл Р.** Основы интерактивной машинной графики. М., «Мир», 1976.
16. **Cohen D., Taft E.** An interactive network graphics system.— "Computers and Graphics", 1975, vol. 1, N 1, p. 27—31.

Поступила в редакцию 21 сентября 1977 г.

УДК 681.3.068

В. Л. КАТКОВ
(Новосибирск)

РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ МАШИННОЙ ГРАФИКИ ДЛЯ МВК «ЭЛЬБРУС»

Введение. Разрабатываемый многопроцессорный вычислительный комплекс (МВК) «Эльбрус» [1] предназначен для решения широкого круга научно-технических и планово-экономических задач в режимах местной и дистанционной пакетной обработки и разделения времени. Производительность комплекса зависит от количества процессоров: при максимальном числе процессоров порядка 10 производительность достигает 12 млн. опер./с. Оперативная память может наращиваться секциями, емкость памяти изменяется от 576 до 4608 *К* байт 72-разрядных слов; объем виртуальной памяти практически не ограничен (2^{32} слов).

Один из основных принципов построения МВК — модульность, за счет которой повышается надежность комплекса (резервирование устройств), появляется возможность адаптации комплекса к классу решаемых задач и обеспечивается постепенное наращивание мощности и развитие комплекса без нарушения работы введенных средств.

Внешнее оборудование подключается к МВК через процессоры ввода-вывода (ПВВ); к ним относятся накопители на барабанах, дисках, лентах и системные пульты. Удаленные терминалы подключаются к ПВВ через процессоры передачи данных (ППД), общее число каналов достигает 2560. К МВК «Эльбрус» может подключаться несколько терминальных ЭВМ через процессор ввода-вывода и (или) процессор передачи данных.

С точки зрения программного обеспечения машинной графики МВК имеет ряд достоинств, облегчающих создание такого обеспечения. Отметим главные из них. Разнообразные типы данных (вещественные, целые, байтовые, битовые, наборы, указатели и т. д.) и форматы позволяют эффективно организовывать древовидно-списочные структуры,