

ЕС-1010 с памятью 16К слов и дисковой операционной системой (см., например, [5]); в [1, 2] описана реализация программного комплекса на ЭВМ без дисков, построенного по этим же принципам и допускающего проведение нескольких экспериментов в реальном времени.

ЛИТЕРАТУРА

1. Э. А. Талныкин. Программное обеспечение экспериментального информационно-измерительного комплекса.— В кн.: Вопросы построения систем автоматизации научных исследований. Новосибирск, Изд. ИАиЭ СО АН СССР, 1974.
2. С. М. Казаков, Э. А. Талныкин. Программное обеспечение машинного комплекса обработки экспериментальных данных.— В кн.: Системы автоматизации научных исследований. (Тезисы докладов конференции). Рига, «Зинатне», 1973.
3. С. В. Бредихин, А. Н. Гинзбург, Б. А. Мелешихин, П. М. Песляк, Э. А. Талныкин. Принципы построения программного обеспечения магистрального информационно-измерительного комплекса.— В кн.: Труды III Всесоюзного симпозиума «Системное и теоретическое программирование». Кишинев, Изд. КГУ, 1974.
4. Л. А. Андрианов, М. А. Ахметьев, П. Я. Белоусов, Ю. Г. Кириллов, А. М. Остапенко, С. Н. Ремесленникова, Э. А. Талныкин. Автоматизированная система с управлением и обработкой на базе ЭВМ HP2116B для изучения динамики флюоресценции монослоя живых клеток.— «Автометрия», 1975, № 2.
5. А. М. Остапенко, Э. А. Талныкин, Н. С. Яковенко. ФОТ — диалоговая система обработки данных.— «Автометрия», 1976, № 1.

Поступила в редакцию 25 июня 1975 г.

УДК 681.3.06

А. М. ОСТАПЕНКО, Э. А. ТАЛНЫКИН, Н. С. ЯКОВЕНКО

(Новосибирск)

ФОТ — ДИАЛОГОВАЯ СИСТЕМА ОБРАБОТКИ ДАННЫХ

Введение. В статье описывается конкретная реализация диалоговой системы обработки данных ФОТ, построенной на принципах, изложенных в [1], откуда будет взята введенная там терминология, поэтому читателю целесообразно сначала прочесть работу [1]. Более подробно будет описана структура системы и две подсистемы модулей, предназначенные для работы со стандартными носителями информации и микроденситометром "Fotomation", представляющим собой цифровую систему с высоким разрешением для считывания и записи полутонных изображений на фотопленку. Организация системы в совокупности с предлагаемым набором модулей полностью избавляет разработчика алгоритмов обработки от необходимости программировать операции над данными, располагающимися на внешних носителях. Программы обработки в системе ФОТ, по существу, не зависят от источников информации, которые определяются уже при работе системы и могут быть реальными изображениями, их копиями на одном из носителей или результатами работы других алгоритмов. Для каждой из операций: чтение, запись или конкретный алгоритм обработки — разрабатывается один программный модуль. Далее в режиме диалога оператор может набирать из имеющихся модулей всевозможные комбинации, такие, например, как чтение изображения с некоторого носителя, несколько последовательных алгоритмов обработки, каждый из которых работает над результатом предыдущего, и запись на другой

носитель. При разработке каждого из модулей такой цепочки нет необходимости знать его «соседей», так как настройка и дальнейшие коммуникации данных происходят под контролем управляющей программы. По специальному запросу (одному для всех модулей и любых источников данных) управляющая программа передает модулю фрагмент изображения, представленный в виде стандартного массива; аналогичная операция имеется для записи. Такой подход упрощает разработку алгоритмов обработки и позволяет оперативно управлять ими при работе системы. Например, можно производить отладку алгоритма на искусственно промоделированных изображениях, всегда имея возможность проверить его на реальных данных. Это значительно повышает эффективность отладки, так как данные, считанные непосредственно с изображением, всегда имеют элемент случайности и часто трудно реально получить критическую для алгоритма ситуацию.

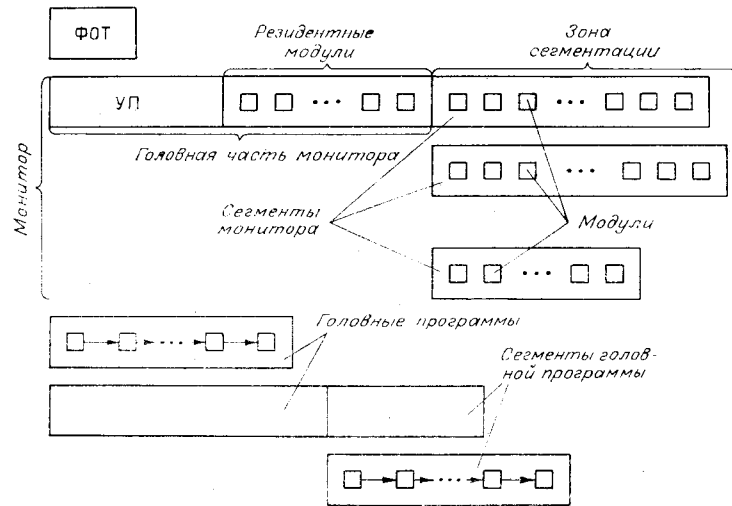
С использованием данного подхода были разработаны алгоритмы выделения характерных деталей изображения, отслеживания линий и различные вспомогательные алгоритмы, обеспечивающие необходимую базу системы обработки изображений. (Сюда можно отнести алгоритмы получения негативов, выделения порогов, сглаживания, повышения контрастности, нанесения раstra и т. п.) Имеются также средства, позволяющие организовать различные форматы хранения изображений, например упакованный формат, значительно экономящий память, или формат, требующий больше памяти, но обеспечивающий высокую скорость доступа при обработке. Один из упомянутых алгоритмов описан в [2].

В настоящее время система развивается в направлении создания новых подсистем модулей. Из уже имеющихся можно выделить подсистемы, ориентированные на задачи статистической обработки данных, Фурье-анализа, обработки текстовой информации (система автоматизации документирования [3]), графического отображения информации. Большую роль система ФОР сыграла при построении программного обеспечения первой очереди многомашинного комплекса ИАиЭ СО АН СССР [4]. Здесь она выполняет функции мониторинга системы, обеспечивающей управление и сбор данных с эксперимента, проводимого с использованием двух ЭВМ, одна из которых находится на линии с крейтом, выполненным в стандарте САМАС [5].

Система реализована на ЭВМ HP2116B на языке высокого уровня [6] и может работать на комплексе АСВТ М-6000 с дисковой операционной системой.

Организация системы. Структура системы и средства управления модулями определяются возможностями операционной системы, на которых мы здесь кратко остановимся. Используемая операционная система (DOS-M HP) имеет один рабочий уровень, т. е. область памяти, где исполняется каждая вызванная директивой оператора программа. Во время работы одна программа специальным запросом к операционной системе может вызвать любую другую. Новая программа загружается в ту же область памяти, может вызвать следующую и т. д. Этот способ представляет собой механизм передачи управления от одной программы к другой без возврата. Кроме того, каждая программа может иметь одну область для сегментации. При вызове сегмента головная программа остается в памяти и соответственно имеет более гибкие возможности по управлению подчиненными модулями.

Система ФОР функционирует на рабочем уровне операционной системы, т. е. все ее компоненты (см. рисунок) выполнены как обычные программы пользователя или их сегменты. Запуск системы производится вызовом по директиве оператора программы ФОР, в функции которой входит инициализация всех системных баз данных и вызов мони-



тора. Монитор также является обычной программой и имеет головную часть и несколько сегментов. В головной части монитора находится управляющая программа (УП), которая поддерживает контакт с оператором и управляет остальными компонентами системы в соответствии с его директивами. В головной части могут также располагаться несколько модулей, выполняющих самые важные или часто используемые в конкретной конфигурации системы функции. Это могут быть модули любых типов: функциональные, входные, выходные, проходные (см. [1]). В каждом из сегментов монитора располагается несколько модулей, обычно выполняющих родственные функции. Перед тем, как передать управление модулю, располагающемуся в сегменте, управляющая программа должна обеспечить присутствие этого сегмента в памяти. Это единственное отличие модулей в сегментах от резидентных модулей, располагающихся в головной части монитора. При настройке модулей в цепочки (см. [1]) управляющая программа следит за тем, чтобы в ней не присутствовало двух модулей из разных сегментов. При нарушении этого условия директива настройки отвергается. Если все же данная операция необходима, можно переписать промежуточные данные на быстрый носитель, а затем продолжить цепочку с чтения. С этой целью модули записи на диск и чтения с диска всегда делаются резидентными. Если цепочка модулей уже частично построена и один из модулей располагается в сегменте, управляющая программа отвергает все директивы, для которых реализующие их модули не могут быть подгружены в память.

Монитор может управлять не только сегментами, но и отдельными программами. Чтобы такая программа могла работать в системе, она не должна в процессе работы модифицировать системную область на диске, а должна по завершении своей работы вызвать монитор. Перед вызовом головной программы монитор спасает на диске все свои переменные базы данных, определяющие его состояние на данный момент. При каждом запуске монитор производит обратную операцию, т. е. восстанавливает свое прежнее состояние и продолжает работать так, как если бы он не терял управления. Программа ФОТ перед первым вызовом монитора приводит системную область на диске в исходное состояние. Таким образом, при вызове головной программы искусственно моделируется передача управления с возвратом и со стороны управляющей программы это более всего похоже на вызов функционального модуля. Поскольку головная программа после вызова работает самостоятельно без какого-либо вмешательства монитора, она может

иметь любую внутреннюю структуру, вызывать другие программы, сегментироваться, иметь для управления свой набор директив. Наряду со стандартными возможностями, предоставляемыми операционной системой, в системе ФОТ есть один дополнительный способ структурирования программ из модулей: специальная библиотечная программа, моделирующая работу управляющей программы-монитора в операциях настройки цепочек и коммуникациях данными. Из модулей, имеющих в головной части монитора или его сегментах, может быть сформирована цепочка с фиксированной настройкой и оформлена как отдельная программа. Для этого не требуется редактирования или перетрансляции модулей, и они продолжают функционировать самостоятельно. Тем не менее часто употребляемые и отработанные комбинации могут фиксироваться.

Управляется ФОТ в диалоговом режиме с помощью директив. В случае готовности к диалогу на системном терминале печатается символ

>

и оператор может вводить следующую директиву. Язык директив описан в [1].

Операции с магнитной лентой. В системе ФОТ имеется ряд модулей, предназначенных для записи изображений на магнитную ленту (при желании на перфоленту), считывания с магнитной ленты, а также выполняющих вспомогательные операции поиска массивов и позиционирования. Изображения располагаются на ленте несколькими зонами и отделяются друг от друга стандартным маркером «Конец файла».

Первая зона является заголовком массива и содержит управляющую информацию: растр и апертуру, установленные при считывании изображения, начальные и конечные отсчеты сканирования, дату формирования и дополнительную идентифицирующую информацию. Соответственно имеются два модуля для формирования и чтения заголовка. Первый модуль в режиме «вопрос — ответ» формирует заголовок и записывает его на ленту. Второй модуль считывает заголовок с ленты и распечатывает его на системном терминале в удобной для оператора форме. Оба эти модуля выполнены как отдельные программы.

В остальных зонах построчно размещаются результаты сканирования изображения. Для обеспечения доступа к записи изображения на ленте имеется входной модуль для построчного чтения и выходной для построчной записи. Операции поиска и позиционирования включают в себя пропуск нескольких массивов и зон, возврат на несколько массивов и зон, перемотку на начало, разгрузку ленты. Все эти модули располагаются в одном сегменте, а модули чтения и записи могут быть объявлены резидентными, размещаться в сегментах, а также использоваться в цепочках с фиксированной настройкой в качестве терминальных модулей.

Размещение на дисках. При размещении изображений на дисках используется формат системы управления файлами (EFMP) операционной системы. Изображения представляются записями фиксированной длины, по одной строке сканирования на запись. При этом используются все стандартные средства доступа, предоставляемые операционной системой. Механизм управления памятью, имеющийся в управляющей программе, позволяет оперативно управлять буферизацией, что в большинстве случаев значительно снижает общее время обработки. Имеются также два модуля для чтения и записи изображения на диск с именем стандартного файла EFMP. Эти модули включаются в резидентные, хотя могут размещаться в сегментах. Ввиду высокой скорости доступа их также очень удобно использовать в качестве терминальных модулей цепочки с фиксированной настройкой.

Считывание и запись изображений. Остановимся кратко на характеристиках используемого устройства. Микроденситометр "Fotomation" (система Р-1700 фирмы "Optronics", США) — высокоскоростная цифровая электромеханическая система, предназначенная для оцифровки изображений, зафиксированных на фотопленке, и вывода оцифрованных изображений на фотоноситель. Устройство позволяет обрабатывать изображения габаритами до 125×180 мм. Изображения считываются и записываются в виде прямоугольного растра, шаг которого устанавливается переключателем на панели устройства и может быть равен 25, 50, 100 мкм. Кроме того, можно установить диафрагму, определяющую квадрат с центром в точке растра с размерами 25×25 , 50×50 , 100×100 , 200×200 мкм. При считывании в каждой точке растра получаем цифровой код средней плотности изображения в выбранном квадрате. Имеется 256 градаций по плотности (яркости). При записи происходит засветка определяемого диафрагмой квадрата интенсивностью, пропорциональной выводимому коду. Развертка по оси Y происходит за счет вращения барабана, на котором закреплена пленка с изображением или фотоноситель. Координата по оси X задается положением каретки, на которой жестко закреплены считывающий и записывающий узлы, связанные каждый со своим барабаном, так что возможно совмещение считывания и записи. Перемещение каретки осуществляется шаговым двигателем, который управляется программно. Один шаг двигателя соответствует 12,5 мкм. При фиксированном положении каретки можно считывать или записывать последовательно код за кодом любой непрерывный участок растра с заданным значением X. При обмене данными ЭВМ должна обеспечить отработку каждого кода не более чем 30 мкс, иначе устройство вырабатывает сигнал «Ошибка».

Микроденситометр комплектуется интерфейсом к машинам серии HP2116 или HP2100, однако какое-либо программное обеспечение отсутствует. Была разработана стандартно организованная программа-драйвер, обеспечивающая возможность работы с устройством, как это принято в операционной системе. Драйвер обрабатывает следующие запросы:

1. Считывание строки при данном положении каретки и размещение полученной информации в память (побайтно). После обработки этой операции возможен переход влево или вправо на один шаг растра.
2. Операция, обратная первой (запись изображения).
3. Перевод каретки в левое крайнее положение.
4. Перевод каретки в правое крайнее положение.
5. Вывод каретки в заданную позицию.
6. Перемещение каретки влево или вправо в шагах растра или в миллиметрах.
7. Перевод целого числа из миллиметров в шаги растра.

На этой базе были разработаны два терминальных модуля для считывания и записи изображений. Оба модуля имеют по четыре параметра, определяющие прямоугольный участок, подлежащий обработке или предназначенный для записи. Параметрами являются координаты левого нижнего и правого верхнего углов прямоугольника в миллиметрах. После запуска цепочки происходит вывод каретки в позицию, определяемую правым отсчетом по оси X, и затем построчное сканирование или запись с движением по направлению ко второму отсчету по оси X. Оба модуля могут быть резидентными, располагаться на сегментах и участвовать в настройке фиксированных цепочек.

Таким образом, операции позиционирования осуществляются автоматически. Для повышения скорости введен один функциональный модуль, имеющий в качестве параметра текущее положение каретки в

миллиметрах, после чего позиционирование осуществляется относительно этого положения.

Формат, выбранный для представления изображений на других носителях, диктовался спецификой данного устройства. Прежде всего, имеется фиксированный способ развертки раstra, и поэтому естественно представлять каждую строку отдельной записью. Кроме того, для возможности обработки, по крайней мере, одна запись целиком должна размещаться в памяти.

Управление памятью. Для обеспечения работы модулей в цепочках и буферизации данных при обмене с внешними устройствами система ФОР имеет простой, но эффективный механизм управления памятью. Это не есть общий механизм динамического распределения памяти; его действие ограничивается временем работы цепочки, а при каждом запуске новой цепочки модулей память иницируется заново. Так как для представлений изображений принят единый построчный формат, то, если даже каждому модулю в цепочке потребуется свой размер буфера под данные, этих размеров будет вполне определенное число. В этой ситуации можно легко избежать фрагментации (когда образуется большое число несвязных областей памяти малого объема) без привлечения сложных алгоритмов «сборки мусора». Все свободные участки памяти связываются в список, и при выделении очередного блока управляющая программа учитывает историю предыдущих запросов.

Генерация системы. В основе работы управляющей программы лежит таблица информации о модулях, которая состоит из записей переменной длины. Каждой записи в таблице соответствует директива и реализующий эту директиву модуль. Ключевым полем записи, по которому осуществляется поиск, является имя директивы. Далее следует информация о типе и месторасположении модуля. Для резидентного модуля это просто адрес точки входа в модуль, для модуля, располагающегося в сегменте,— имя сегмента и адрес точки входа, а для головной программы — имя программы.

При генерации системы таблица информации о модулях представляет собой обычный объектный модуль, который может обрабатываться загрузчиком операционной системы. Управляющая программа имеет ссылку на таблицу, а таблица — на все необходимые в системе модули. Далее загрузчик на этапе редактирования связей присоединяет к системе лишь те модули, на которые имелись ссылки из управляющей программы, таблицы или из уже присоединенных модулей. После завершения загрузки * монитор системы готов к работе.

Имя головной программы в таблице фигурирует лишь как данные, и головные программы в загрузке монитора не участвуют, а загружаются отдельно. Запись в таблице для головной программы нужна, чтобы связать имя директивы с именем программы. Итак, для включения в систему отдельной программы разработку ее можно вести совершенно независимо. Допустим, что уже имеется некоторая программа *ADC*, требующая для работы двух параметров. Если ее еще нет в таблице, ее можно активизировать директивой

>MOD, ADC, P1, P2

Если пользователя из каких-то соображений это не удовлетворяет, при первой же регенерации системы эту программу можно внести

* Несколько незаконное употребление термина «загрузка» обусловлено тем, что в используемой системе программирования функции редактирования связей и загрузки совмещены в одной программе, называемой загрузчиком, которая формирует на диске готовый к использованию модуль.

в таблицу, например, под ключом АЦП. После этого ее можно будет вызывать также и директивой

>АЦП, P1, P2

Таким образом, генерация системы при наличии всех необходимых модулей сводится фактически к формированию таблицы, и далее работает стандартный процесс загрузки. Здесь возможны несколько подходов; укажем три, по-видимому, основные.

Первый подход состоит в разработке специальной программы-генератора, которая в режиме диалога получает необходимую информацию от оператора и формирует объектный модуль. Этот подход применялся в [7]. Разработка такой программы довольно трудоемка, а пользоваться ей удобно лишь на первых порах, так как при большом объеме системы диалог занимает много времени.

Второй подход заключается в составлении таблицы на языке, допускающем описание произвольных структур данных, включая внешние ссылки (например, это всегда можно сделать на языке ассемблера). Такой подход использовался в данной системе, так как язык ее реализации [6] имеет упомянутые средства. Здесь формирование таблицы сводится к редактированию текста и стандартному процессу трансляции. При наличии хорошего текстового редактора этот способ достаточно эффективен.

Третий подход состоит в описании таблицы на специализированном языке и построении транслятора с этого языка в объектный код. Этот подход предполагается применить при дальнейшем развитии системы. Диктуется это тем, что необходимо представить возможность генерации системы человеку, минимально знакомому с ее организацией, и требуется язык, наиболее естественный, а основано это на том, что в существующей версии языка [6] имеются средства для построения синтаксически управляемых систем и сделать это будет очень просто.

Заключение. Структура системы, воплощенная в управляющей программе, является ядром, специфические качества которого определяются составом модулей. В этом отношении ядро можно рассматривать как средство управления модульным пакетом программ любого назначения.

ЛИТЕРАТУРА

1. Э. А. Талныкин. Модульное программирование в задачах сбора и обработки экспериментальных данных.— «Автометрия», 1976, № 1.
2. А. М. Остапенко, Э. А. Талныкин. Об одном алгоритме обработки изображений.— «Автометрия», 1974, № 4.
3. В. А. Мелешихин, П. М. Песляк, Э. А. Талныкин. Система автоматизации документирования на базе ЭВМ «Минск-32».— «Автометрия», 1974, № 4.
4. С. В. Бредихин, А. Н. Гинзбург, В. А. Мелешихин, П. М. Песляк, Э. А. Талныкин. Принципы построения программного обеспечения магистрального информационно-измерительного комплекса.— В кн.: Системное и теоретическое программирование. (Труды III Всесоюзного симпозиума). Кишинев, Изд. КГУ, 1974.
5. В. Д. Бобко, Ю. Н. Золотухин, Ю. М. Крендель, З. А. Лившиц, А. П. Ян. Магистральная система обмена информацией.— «Автометрия», 1974, № 4.
6. П. М. Песляк, Э. А. Талныкин. Язык системного программирования для мини-ЭВМ.— «Автометрия», 1974, № 4.
7. Э. А. Талныкин. Программное обеспечение экспериментального информационно-измерительного комплекса.— В кн.: Вопросы построения систем автоматизации научных исследований. Новосибирск, Изд. ИАиЭ СО АН СССР, 1974.

Поступила в редакцию 22 июля 1975 г.