ГОСУДАРСТВЕННЫЙ КОМИТЕТ
ПО ИСПОЛЬЗОВАНИЮ АТОМНОЙ ЭНЕРГИИ СССР

ИНСТИТУТ ФИЗИКИ ВЫСОКИХ ЭНЕРГИЙ

ИФВЭ 88-104
ОУНК

Ю.В.Куянов

Q – ОПЕРАЦИОННАЯ СИСТЕМА ДЛЯ УПРАВЛЯЮЩИХ ЭВМ

Серпухов 1988

УДК 234.384.6. М-24

Аннотация

Куянов Ю.В. Q - операционная система для управляющих ЭВМ: Препринт ИФВЭ 88-104. - Серпухов, 1988. - 28с., библиогр.: 16.

Описана мультипрограммная исполнительная операционная система, реализованная для ЭВМ типа "Электроника-60", PDP-11 и других ЭВМ, совместимых с ними на уровне машинных команд.

Показано, что при использовании Q в системах автоматизации исследований и управления упрощается организация интерфейса с нестандартным электронным оборудованием.

Рассматриваются вопросы применения ЭВМ DEC-10 как инструмента при создании программного обеспечения для управляющих ЭВМ.

Abstract

Kuyanov Yu.V. Q - Operating System for Control Computers: IHEP Preprint 88-104.-Serpukhov, 1988. -p28., refs.: 16.

A multiprogramming executive operating system realized for "Electronika-60", PDP-11 and other computers, compatible with them in the instruction set level is described.

It is shown that using Q, the interface organization with non-standard electronics equipment in research and control automation systems is simplified.

Some questions of using the DEC system 10 as an instrument to produce an application software for controlling computers are considered.

© Институт физики высоких энергий, 1988.

ВВЕДЕНИЕ

Построение автоматизированных систем для научных исследований с использованием управляющих ЭВМ (УВМ) выдвигает проблему такой организации аппаратно-программного обеспечения, которая способствует эффективному решению конкретных задач на всех этапах жизненного цикла. Типична ситуация, когда в процессе создания и эксплуатации такой системы вследствие изменения или уточнения решаемых задач, выявления узких мест или появления новой аппаратуры с лучшими техническими характеристиками возникает необходимость в изменении структуры аппаратно-программных средств.

Для адаптации к пользователю и изменяющимся условиям эксплуатации достаточно применить ЭВМ общего назначения с набором внешних устройств и универсальной операционной системой (ОС), обеспечивающей разработку программного обеспечения (ПО) с использованием языков высокого уровня, а в необходимых случаях ассемблера.

Однако выбор такой ЭВМ в качестве управляющей ограничивают следующие факторы:

- относительно высокая стоимость и большие габариты ЭВМ с необходимым набором внешних устройств;
- значительное энергопотребление, что является ограничением в распределенных системах, состоящих из нескольких УВМ;
- трудно устранимое замедление реакции УВМ на внешние события (прерывания) из-за случайных совпадений по времени с обращениями ОС к внешней памяти на дисках.

ЭВМ общего назначения является как бы "одинаково неэффективной" для всех решаемых в ней задач. С другой стороны, специализированный процессор, как правило, не менее чем в 10 раз дешевле равномощного ему универсального /1/. Этим обусловлена практика внедрения микроЭВМ с распределением функций по принципу - одна машина на одну функцию. Во многих случаях отдельные функции УВМ не являются автономными. Возникает необходимость даже при децентрализованном управлении локализовать часть связей через общую память. Современным методом решения этой проблемы на уровне аппаратуры является создание мультимикропроцессорных систем с многовходовой памятью /2,3/.

Совмещение машин на однопроцессорной ЭВМ приводит к необходимости организации мультипрограммной структуры таких машин, что реализуется путем создания ОС. Ограничения ОС существенно затрудняют применение новых аппаратных средств, которые составляют главное отличие УВМ от ЭВМ общего назначения.

В данной работе дается общая характеристика ОС, обозначаемой в дальнейшем "Q" из соображений удобства идентификации.

В силу преимущественной ориентации на использование УВМ без внешней памяти на дисках Q по существу является исполнительной системой, объединяющей необходимые программные компоненты и загружаемой извне вместе с ними в оперативную память. Разработка и сопровождение ПО возлагается на развитую кросс-систему на основе ОС ЭВМ коллективного пользования или интерактивной ОС общего назначения на мини- или микроЭВМ.

1. ОСНОВНЫЕ ТРЕБОВАНИЯ К ПРОГРАММНОМУ ОБЕСПЕЧЕНИЮ УВМ

Какие факторы приобретают решающую роль в организации ПО УВМ?

Параллелизм и реконфигурируемость процессов. При постановке задачи, как правило, ясно представляют себе всю совокупность параллельно выполняемых процессов и синхронизирующих их событий (сигналов). В целях обеспечения эффективности процессов и в особенности требуемого времени отклика на внешние события, внутрисистемные процессы нижнего уровня, синхронизируемые аппаратно, не должны быть скрыты от разработчика ПО. Необходимы гибкая организация и полный контроль над структурой всех процессов. Кроме того, по ходу дела может потребоваться включение в совокупность новых процессов и сигналов, что должно поддерживаться наиболее простыми средствами.

Отказоустойчивость. Асинхронно работающие процессы верхнего уровня, в дальнейшем именуемые задачами, должны обладать запасом жизнеспособности. В случаях аварийного выключения любой задачи, другие слабо связанные с ней задачи должны сохранять работоспособность. Для этого выключившаяся задача должна освободить используемые ресурсы и, насколько это возможно, быть готовой к повторному включению.

В случае полной потери работоспособности нужно предусмотреть средства самовосстановления, например быстрый перезапуск, а также средства, облегчающие идентификацию причин отказов.

Сегментация процессов и распределение памяти. Для улучшения рабочих характеристик допустимо отказаться от сложных способов защиты памяти. Важнее обеспечить быструю связь между задачами, а также возможность разделения сегментов данных и процедурных сегментов, чтобы устранить дублирование кодов идентичных процессов.

Свободная оперативная память должна быть общим ресурсом, динамически распределяемым между задачами по запросам от них. Это обусловлено необходимостью размещать массивы данных, динамически изменяющие свой объем в широких пределах, что, например, имеет место в системах сбора данных от различных источников /4/.

Модульность. ПО должно иметь модульную структуру с согласованными интерфейсами, чтобы в наибольшей степени способствовать экономичности реализации УВМ.

Интерактивность. Должен быть реализован простой и легко расширяемый человеко-машинный интерфейс, который, в первую очередь, должен предусматривать диалоговые возможности с использованием терминалов различных типов, а также средства динамической отладки задач и диагностики ошибок.

Терминальный интерфейс имеет самостоятельное значение во многих приложениях. Но даже в тех случаях, когда УВМ рассчитана на автономную работу, включение пультового терминала в отладочные версии ПО позволяет воспользоваться набором пультовых команд динамической отладки таких, как установка контрольных точек, активизация заданной задачи, выдача на экран (или распечатка) содержимого общих регистров, стека и указанных областей памяти.

2. ОБЩАЯ ХАРАКТЕРИСТИКА ОПЕРАЦИОННОЙ СИСТЕМЫ Q

Удовлетворению перечисленным требованиям для УВМ, программно совместимых с PDP-11, в значительной мере служат системы программирования, базирующиеся на мини-ЭВМ PDP-11 /5,6/. Хорошим решением является реализация ПО УВМ на базе ОС UNIX /7,8/. В случае применения в качестве инструментальной программно несовместимой ЭВМ требуется поиск иного решения. Примером может служить работа /9/, где ЭВМ PDP-11 и LSI-11 использованы как мультиплексоры и концентраторы терминалов, а также как специализированные процессоры для сбора данных. Настоящая работа является дальнейшим развитием концепций /9/ для широкого класса решаемых в ИФВЭ задач, базирующихся на УВМ типа "Электроника-60".

При наличии даже медленного канала связи между инструментальной и управляющей ЭВМ обеспечивается поэтапная и комплексная отладка программного обеспечения (ПО) и аппаратуры на реальной УВМ или специализированном стенде. Из этого следуют два важных требования к Q:

- простота реконфигурации структуры ПО и объединяемых программных компонентов;
- средства, облегчающие отладку, должны содержаться в операционной системе, хотя и лишенной средств разработки ПО.

Первое требование в значительной мере удовлетворяется путем

описания структуры ПО на непроцедурном языке с последующей автоматической обработкой файла описания на инструментальной ЭВМ с помощью специальной программы - генератора QGEN. Результатом работы QGEN является объектный модуль данных системы, объединяемый впоследствии с необходимыми программными компонентами. С помощью макроинтерпретатора команд ОС на инструментальной ЭВМ значительно упрощается получение модуля, готового к загрузке в УВМ.

Нетривиальная последовательность внешних событий определяет мультипрограммную структуру процессов - задач, обеспечивающих требуемую реакцию на совокупность событий. В простейшем случае каждая задача реагирует на свой набор внешних событий. Задачи работают асинхронно друг с другом, но синхронизируются внешними событиями и конкурируют в основном за центральный процессор УВМ.

Более типичный случай, когда задачи взаимодействуют друг с другом путем изменения состояния либо через общую память, либо передавая сообщения. Кроме того, они могут оказывать дополнительные синхронизирующие воздействия. Требование быстроты такого взаимодействия в сочетании с малым потреблением вычислительных ресурсов является главной причиной совмещения задач на одном процессоре и общей памяти.

Синхронизация задач с внешними событиями и между собой - важнейшие функции ОС. Первая функция обеспечивается на этапе объединения объектных модулей путем подключения специализированных программ - обработчиков внешних прерываний, вторая функция допускает реализацию различными способами /10/.

В данной работе синхронизирующие взаимодействия задач осуществляются через управляющую программу нижнего уровня - монитор. Вызов монитора является единственным механизмом взаимного исключения. В мониторе реализованы все критические функции ОС, которые выполняются при заблокированных прерываниях. Такими функциями являются также распределение устройств и распределение области динамической памяти. В отдельных случаях допустимо заключение критического участка в операторные скобки IOFF; - ION;, что блокирует вход в критический участок других процессов, а также их влияние на проверяемые внутри участка условия. Однако при этом программист принимает на себя полную ответственность за предотвращение тупиковых ситуаций.

Устройство - это ресурс среды, предоставляемый задаче по специальному вызову на все время вплоть до вызова для освобождения устройства. Кроме традиционно понимаемых устройств, необязательно поддерживаемых системой прерываний, возможны также чисто программные устройства типа "почтовый ящик". Q связывает с устройством статическую область памяти, выделяемую генератором QGEN. Эта область может использоваться независимо от того, к какой задаче прикреплено в данный момент устройство.

Задачи, работая логически параллельно, последовательно (конкурентно) используют центральный процессор УВМ. Механизм ОС, ответственный за предоставление процессора задачам, сосредоточен в управляющей программе нижнего уровня - диспетчере. Диспетчер просматривает список задач и предоставляет процессор первой из них, готовой к исполнению. Таким образом, положение задачи в списке определяет относительный приоритет задачи. Всякий вызов монитора автоматически оканчивается вызовом диспетчера. Кроме того, диспетчер периодически вызывается по прерыванию от таймера. И, наконец, любой обработчик внешних прерываний может вызвать диспетчер явно.

Для того, чтобы реализовать обработчик прерываний, обеспечивающий активизацию логически связанной с внешним событием задачи, и связать обработчик прерываний с заданным вектором прерываний, требуется объявить при генерации устройство (в приложении 2 содержится более подробная информация об этом).

Логическая связь задачи с внешним событием (соответствующим обработчиком прерываний) осуществляется однократно при инициализации ОС путем прикрепления устройства к задаче. В более сложных случаях задачи могут освобождать устройства, разрывая тем самым логическую связь с ними и делая их доступными для других задач. Задача может также, вызывая монитор, передать связанное устройство другой задаче. Во всех таких случаях, а также при наличии однотипных устройств целесообразно объединение в программный модуль процедур, реализующих функции программно-аппаратного интерфейса задач с логически связанным устройством данного типа. Вызову таких процедур должен предшествовать вызов монитора для прикрепления нужного устройства.

В отличие от интерфейсных процедур, вызываемых задачами, обработчик прерываний реализует аппаратно-програминый интерфейс по инициативе устройства через вектор прерываний. Для однотипных устройств, различающихся вектором прерываний, возможна реализация общего для группы устройств обработчика прерываний. Это поддерживается определяемой при генерации статической областью памяти, связанной с каждым устройством, которая может содержать до четырех адресов аппаратных регистров, принадлежащих данному устройству. Значения адресов указываются при генерации.

Предусмотрена также связь устройств с такими событиями, как пуск всей совокупности задач, выключение питания УВМ и нештатное прекращение работы задачи, к которой прикреплено устройство. Во всех этих случаях автоматически вызывается объявленная при генерации процедура восстановления (ПВ). В ПВ можно индивидуально для каждого устройства запрограммировать необходимые действия: маскирование прерываний, переключение на ручной режим, включение индикации и т.п.

В число ресурсов ОС вместе с набором устройств и процессором входит также динамическая память (ДП). Границы области ДП можно определить явно при генерации. По умолчанию под ДП отводится вся свободная область памяти УВМ. ДП выделяется буферами любой длины по специальному вызову монитора. Затем задача может, вызвав монитор, либо освободить буфер, либо передать его явно в распоряжение другой задаче. Передача буферов между задачами - наиболее экономный и весьма эффективный способ обмена данными в тех случаях, когда оперативная память является критическим ресурсом.

При нештатном завершении работы задачи происходит освобождение всех ресурсов, в том числе устройств и ДП. Дополнительно через специальный "аварийный почтовый ящик" может быть выдано диагностическое сообщение на пультовой терминал. Задача переводится в исходное состояние. Последнее свойство особенно полезно при отладке ПО УВМ и частично обеспечивает защиту других задач от отлаживаемой. Реализация аварийного почтового ящика обеспечивает диагностику ошибок, затрагивающих более чем одну задачу.

Более тяжелый случай представляет так называемый "крах" в результате ошибок ПО. В этом случае Q идентифицирует причину сбоя, а также выдает на пультовой терминал информацию о последовательности последних вызовов монитора. После этого происходит автоматический начальный пуск всей совокупности задач.

Полезно включать пультовой терминал в отладочные варианты Q вместе с секциями общего назначения, которые реализуют интерпретацию таких пультовых команд, как установка контрольных точек, активизация заданной задачи, выдача на экран (или распечатка) содержимого общих регистров, стека и указанных областей памяти. В дальнейшем как терминал, так и соответствующие программные секции можно легко исключить из Q. В других случаях набор команд с терминала можно изменить или дополнить. В Q предусмотрен расширяемый диалог с одного или нескольких терминалов.

Программные секции общего назначения, реализующие такие функции, как, например, обслуживание таймера УВМ, форматный вывод на терминал, интерпретатор команд с терминала, преобразование данных, получение информации о текущем состоянии системы, арифметические операции над вещественными числами и др., можно выборочно объединять с конкретными секциями специального назначения, формируя оптимальную среду для задач управления.

3. ГЕНЕРАЦИЯ И ФУНКЦИОНИРОВАНИЕ ОС Q

Штатные инструментальные средства ЭВМ DEC-10 обеспечивают компиляцию с языка PL-11 /11/ для ЭВМ, программно совместимых с PDP-11, и объединение выходных модулей компилятора. Кроме того,

для реализации генератора QGEN использован компилятор с языка BCPL /12/. Для загрузки УВМ, связанных с DEC-10 по терминальным линиям связи, разработана программа загрузчик /13/, работающая в DEC-10, и специальная программа инициализатор /14/, размещаемая в ПЗУ УВМ типа "Электроника-60".

Выбору реализации способствовало распространение микроЭВМ "Электроника-60" минимальной конфигурации. Система программирования основана на использовании только языка программирования PL-11, являющегося оптимальным компромиссом между простотой программирования и сопровождения, а также машинной эффективностью готовых программ.

Принципиально важен механизм вызова монитора. При каждом вызове требуется запись в стек двух аргументов. Затем по символическому имени вызывается соответствующая процедура монитора. Вызов монитора реализован с помощью процедуры TRAP /11/ (программного прерывания), т.е. через экстракод, приводящий к блокировке внешних прерываний на время выполнения функций монитора. Монитор освобождает стек вызвавшей задачи от аргументов, а в конце вместо выхода вызывает диспетчер, который отдает процессор наиболее приоритетной задаче, готовой к работе.

Пользователю достаточно знания функций монитора и аргументов для вызова (приложение 1). Детали определения всех функций содержатся в системном файле, который следует включать в исходный текст программы при помощи управляющей строки

[INCLUDE QUSER.INC[5,3]

Задачам и устройствам пользователь самостоятельно присваивает логические имена на этапе описания. Каждое имя длиной не более 8 знаков ASCII, первый из которых буква, можно употреблять в программе пользователя для получения внутренних идентификаторов задач и устройств посредством вызова процедур GETTASK и GETDEV. Внутренние идентификаторы являются указателями на структуры данных, соответствующих задаче или устройству.

Структуры данных задач и устройств формируются генератором QGEN и объединяются в списки задач и устройств (более подробно о генерации Q в приложении 2). При генерации предусмотрено расширение области памяти, выделяемой каждой структуре на указанное пользователем число слов. Это дает возможность применения техники чистых процедур для параллельной обработки данных по общему алгоритму. Простейшим примером является программный модуль QT, обеспечивающий интерфейс задачи с терминалом. Единственный экземпляр этого модуля достаточен для обслуживания любого числа объявленных при генерации терминалов.

Применение техники чистых процедур не навязывается пользователю, но рекомендуется в условиях ограниченной оперативной памяти и поддерживается ОС не только путем выделения статической

области памяти требуемого размера для каждой задачи, но и посредством индивидуального стека задачи, в котором также сохраняется содержимое общих регистров, когда задача не активна.

В Q реализован расширяемый интерпретатор команд с терминала в виде задачи, одноименной терминалу и автоматически включаемой в систему, если это указано при объявлении устройства типа терминал. Интерпретатор использует пакет чистых процедур, реализующих в основном функции, полезные при отладке.

Структура команд интерпретатора предусматривает двухсимвольный идентификатор команды. Первый символ должен быть буквой, а имя интерпретирующей процедуры должно начинаться именно с этих двух символов. Далее через любое количество пробелов или табуляторов могут следовать индивидуальные для каждой команды списки аргументов (обычно числа или ключевые слова). Аргументы передаются на вход интерпретирующей программы с помощью указателя. Задача интерпретатор активизируется после ввода закрывающего символа возврата каретки (RETURN, CR или ВК) и вызывает соответствующую интерпретирующую процедуру, передавая ей значение указателя на строку аргументов.

Путем вызова фильтра ввода DECODE каждая интерпретирующая процедура получает список аргументов в преобразованном виде, где аргументам предшествуют маркеры типа: -1 для чисел, -2 для знака "+", значение длины знаковой последовательности для ключевых слов и 0 (нуль) для обозначения конца списка аргументов. Набор команд столь просто устроенного интерпретатора несложно расширить или заменить, исходя из потребностей конкретных применений.

Интерпретатор терминальных команд является полезным средством даже для УВМ, предназначенных для работы вообще без терминалов. В этом случае генерация Q с временно подключенным терминалом является лучшим практическим средством, чем эмуляция на другой ЭВМ, так как позволяет производить отладку в условиях, приближенных к эксплуатационным, с использованием реальных внешних устройств и сигналов, скорее всего отсутствующих на другой ЭВМ.

Особенно сложна отладка параллельных процессов в реальном времени. Включение интерпретатора терминальных команд в систему вместе с задачами, работающими в реальном времени, обеспечивает отображение состояния задач и устройств в контрольных точках, оперативное включение и отключение задач, вывод на терминал содержимого выбранных областей памяти, общих регистров и стека задач, а в особо тяжелых случаях "краха" полезна хронологическая регистрация последних вызовов монитора.

ПО УВМ на основе Q можно условно разбить на 3 уровня иерархии. Верхний уровень образуют проблемно-ориентированные модули задач для конкретных применений (не рассматриваемые в данной работе). Нижний уровень представлен четырьмя программными модулями

и двумя модулями данных. Первый модуль данных всегда занимает область младших адресов. Его структура и размер определяются конфигурацией системы. Объектный код модуля строится генератором QGEN, на вход которого подается файл пользователя с описанием на непроцедурном языке. В результате полностью определяется структура области векторов прерываний, системные параметры и списки структур задач и устройств. Второй модуль данных содержит параметры Q, не зависящие от конфигурации. Он получается автоматически при компиляции первого программного модуля QM0 и выделен в глобальный сегмент, доступный из всех модулей, имеющих строку

[INCLUDE QROOT[5,3]

Программный модуль QM0 функционально объединяет монитор и диспетчер, а модуль QM3 содержит инициализатор ОС, который после автоматической загрузки начинает работать первым при старте с нулевого адреса. После остановки УВМ его можно запустить вручную с адреса 40_8.

Эти четыре модуля необходимы при всех конфигурациях Q. Размер трех из них, не зависящий от конфигурации составляет 2092 байта.

Модуль QM1 содержит процедуру CRASH, выполняющую функции диагностики при "крахе". Процедура CRASH может быть вызвана вручную с адреса 44_8.

Модуль QM2 содержит обработчик прерываний по таймеру УВМ для программно реализованного датчика времени суток, а также вызова диспетчера по истечении установленных задачами таймаутов и периода диспетчеризации по кругу (round robin), устанавливаемого по необходимости при генерации. Последние 2 модуля можно исключить из Q при отсутствии терминала или таймера.

Средний уровень образован модулями устройств и программ обслуживания. Модульная структура этого уровня организована по функциональному, а не по иерархическому принципу. Простейший модуль устройства содержит описание структуры данных устройства, набор процедур обращения к устройству с верхнего уровня, процедуру обработки прерываний и процедуру восстановления, вызываемую с нижнего уровня. Такая группировка структуры данных и элементарных функций устройства в одном модуле облегчает включение устройств в сборку различных версий ПО управляющей ЭВМ.

В более сложных реализациях устройств целесообразно разбиение на несколько модулей. Тогда удобно совместно используемую структуру данных определить в отдельном файле и связывать ее с процедурными модулями при помощи оператора INCLUDE. Разбиение ПО на модули, исходный текст которых умещается на двух листах АЦПУ, способствует лучшему пониманию реализованных программных средств и облегчает внесение модификаций, которое неизбежно в течение жизненного цикла создаваемых систем. Описание типовых процедур в реализованных модулях содержится в приложении 3.

4. ПРИМЕНЕНИЕ ОС Q

Система программирования, базирующаяся на описанной ОС Q, эксплуатируется в ИФВЭ с 1982 г. Созданное программное обеспечение работает на четырех ЭВМ PDP-11/40 и одиннадцати ЭВМ типа "Электроника-60".

Уже первый опыт применения ОС Q для решения задачи стабилизации интегрального поля в многомодульном ударном магните системы быстрого вывода на ускорителе ИФВЭ /15/ показал, что программный интерфейс с новым электронным оборудованием /16/ реализуется простыми и понятными средствами, комплекс в целом достаточно надежен и имеет резерв функционального расширения путем добавления новых процессов и устройств.

В дальнейшем эта ЭВМ была объединена с семью другими ЭВМ "Электроника-60" в систему управления выводом пучка из кольцевого ускорителя ИФВЭ. ОС Q объединяет от трех до девяти совместно работающих процессов в каждой из этих ЭВМ, обеспечивая их автономную работу. По линиям связи необходимая информация передается в те процессы других ЭВМ, которые ее запрашивают.

В процессе эксплуатации ЭВМ DEC-10 в 1980-1984 гг. значительно возросли требования к ее коммуникационной подсистеме. Образующие ее три ЭВМ PDP-11/40 не использовали все свои резервы из-за сложной, но примитивной штатной коммуникационной программы. Замена программы на более совершенную, работающую под управлением ОС Q, была произведена в 1983 г. При этом все штатные функции были воспроизведены полностью. В последствии, были добавлены новые, что привело к созданию терминальной сети QNET.

Адаптируемость ОС Q к изменению условий эксплуатации особенно полезна при автоматизации исследований новейшего оборудования на специализированных стендах. Такие стенды созданы и эксплуатируются вместе со специализированными программами, работающими под управлением ОС Q:

- для исследования импульсных генераторов ударных магнитов УНК;
- для настройки и испытания контроллера прямого доступа к памяти ЭВМ СМ-4 в локальной информационно-вычислительной сети;
- для исследования высокостабильных источников питания зарядных устройств импульсных систем УНК.

Предполагается также применение Q при создании микропроцессорных контроллеров системы автоматизированных стендов источников питания УНК.

ЗАКЛЮЧЕНИЕ

Автор выражает благодарность руководству ОМВТ за предоставленную возможность выполнить работу, А.П.Елину и Н.Н.Трофимову за оптимизм при использовании экспериментальных версий ОС, В.Г.Кузьменко, О.В.Курнаеву, А.А.Лебедеву и А.И.Федосееву за полезные обсуждения и замечания.

Список литературы

1. Вишневский Ю.Л., Котов В.Е., Марчук А.Г. Модульная асинхронно развиваемая система. // Кибернетика. 1984. N 3. С. 22-29.
2. Altaber J., Innocenti P.G., Rausch R. Multi-microprocessor Architecture for LEP Storage Ring Controls. // Proceedings of the 6th Annual Workshop on Distributed Compuier Control Systems, International Federation of Automatic Control. - Monterey. 1985.
3. Nash T., Areti H., Biel J., Case G., Cook A., Fischler M., Gains I., Hance R., Husby D., Zmuda T. The ACP Multiprocessor System at Fermilab. // Proceedings of the XXIII International Conference on High Energy Physics. - Berkeley, California. 1986.
4. Кибиткин В.В. Особенности операционных систем микроЭВМ. // УСиМ. 1982. № 1. С. 31-35.
5. RSX-11S System Generation and Installation Guide. - DEC Maynard, Massachusetts, USA, 1977.
6. Wirth N. Modula-2. // Institut fur Informatik ETH CH-8092, Zurich, 1980.
7. Lycklama H., Christensen C. A Minicomputer Satellite Processor System. // The Bell System Technical Journal. 1978. Vol.57. № 6. Part 2. P. 2103-2113.
8. Егошин Л.А., Казаков П.Н., Куянов Ю.В., Сахаров В.П. Расширение средств операционной системы ДЕМОС для задач реального времени: Препринт ИФВЭ 86-228. - Серпухов, 1986.
9. Parkman C.F., Lee J.G. Omnet - High Speed Data Communications for PDP-11 Computers. - CERN 79-11. Geneva, 1979.
10. Цикритзис Д., Бернстайн Ф. Операционные системы. - М.: Мир, 1977.
11. Russel R.D. PL-11: A Programming Language For The DEC PDP-11 Computer. - CERN 74-24 Rev. Geneva, 1978.
12. Richards M. The BCPL Programming Manual. - The Computer Lab., Univercity of Cambridge, 1973.

13. Куянов Ю.В., Петухов В.А, Савин Н.П. Аппаратно-программные средства загрузки программ в удаленные ЭВМ "Электроника-60" и PDP-11 по асинхронным линиям связи: Препринт ИФВЭ 82-129. - Серпухов, 1982.
14. Куянов Ю.В., Петухов В.А, Савин Н.П. Расширение терминальной сети ЭВМ DEC-10 с помощью группового контроллера на базе микроЭВМ "Электроника-60": Препринт ИФВЭ 83-107. - Серпухов, 1983.
15. Комаров В.В. и др. // Труды Восьмого Всесоюзного совещания по ускорителям заряженных частиц. Протвино, 1982. - Дубна: 1983.
16. Трофимов Н.Н. Каркасный контроллер для микроЭВМ "Электроника-60": Препринт ИФВЭ 82-116. - Серпухов, 1982.

Рукопись поступила 11 мая 1988 г.

Монитор

Функции монитора осуществляются управляющими примитивами. Вызов любого примитива, вообще говоря, приводит к приостановке вызвавшей его задачи, так что следующей будет работать задача с наивысшим приоритетом, находящаяся в активном состоянии.

Объявления имен примитивов Q содержатся в файле QUSER.INC, который необходимо командой INCLUDE включать в исходные тексты программ, вызывающих монитор. Для вызова монитора используется аппарат программных прерываний (TRAP). Прежде, чем вызвать монитор, задача помещает в свой стек 2 аргумента. Монитор извлекает их из стека. Общие регистры задач не изменяются монитором, за исключением специально оговоренных случаев.

Список примитивов монитора

1. GETBUF	2. RELBUF	3. MOVBUF	4. ATTACH
5. DETACH	6. HANDOVER	7. CHECKIO	8. RELEASE
9. DELAYTIC	10. DELAYSEC	11. HOLDTASK	12. RUNTASK
13. TERMINATE			

Некоторые примитивы используются обработчиками прерываний путем прямого вызова внешней процедуры (имя процедуры соответствует имени примитива с добавленной в конце буквой 'M', например, RUNTASKM). Эта особенность вызова монитора обусловлена тем, что обработчик прерываний не имеет собственного стека, а заимствует его у случайно прерванной задачи.

Следующие 3 примитива управляют динамической памятью (ДП).

1. **GETBUF** - получить буфер из ДП.

Q выделяет для задачи свободный блок динамической памяти требуемого размера. Указатель на начало буфера передается задаче в регистре R4. Если нет свободной области памяти подходящего размера, то действие Q зависит от знака числа байтов, определяемого при вызове монитора. Если положительный, то задача будет приостановлена, ожидая памяти (состояние 'M'). Если отрицательный, то приостановки задачи не будет, но в R4 нуль.
Аргументы: (1) число байтов, (2) не используется.
Возвращаемый результат: R4 - адрес буфера или нуль.

2. **RELBUF** - освободить буфер ДП.

Q освобождает область динамической памяти, которая была выделена задаче при помощи примитива GETBUF, и, если необходимо, распределяет ее среди других задач в порядке их приоритета. Если освобождаемый буфер примыкает к другому свободному участку ДП, то он сливается с ним, образуя участок большего размера.
Аргументы: (1) адрес буфера, (2) не используется.

3. **MOVBUF** - передать буфер динамической памяти другой задаче.

Q передает буфер от вызвавшей задачи к другой задаче.
Аргументы: (1) адрес буфера, (2) указатель задачи, которой передается буфер.

Буфер должен быть получен вызвавшей задачей из динамической памяти, а задача-получатель должна существовать.

Следующие примитивы управляют устройствами.

4. ATTACH - присоединить устройство к вызвавшей задаче.
Q проверяет, свободно ли устройстао. Если да, то присоединяет его к задаче (переводит устройство в состояние 'A'). Если нет, то приостанавливает вызвавшую задачу (состояние 'D').
Аргументы: (1) указатель устройства, (2) не используется.

5. DETACH - отсоединить устройство.
Q выполняет функцию отсоединения устройства от данной задачи, переводя устройство в состояние "свободно" ('F'). Если устройство выполняет ввод/вывод, т.е. находится в состоянии "занято" ('B'), то его отсоединение будет отложено до окончания ввода/вывода путем перевода устройства в состояние "освобождается" ('R'). Отсоединение устройства заключается в переводе его в состояние "свободно" и, если есть задачи, требующие присоединить его, то перевод наиболее приоритетной из них в активное состояние ('A') с присоединением данного устройства.
Аргументы: (1) указатель устройства (в R1 для прямого вызова), (2) не используется.

6. HANDOVER - передать устройство.
Q передает присоединенное устройство другой задаче. В результате устройство оказывается присоединенным к другой задаче.
Аргументы: (1) указатель устройства, (2) указатель задачи, которой передается устройство.

7. CHECKIO - приостановить задачу до окончания процесса ввода/вывода.
Q проверяет, выполняет ли устройство ввод/вывод (состояние 'B'), и в этом случае приостанавливает задачу до завершения ввода/вывода (состояние 'I').
Аргументы: (1) указатель устройства, (2) не используется.
Замечание: Задача обязана перевести устройство в состояние "занято" перед инициализацией ввода/вывода путем операции:

```
DEVBUSY => DEVSTAT;
```

Предварительно нужно поместить указатель устройства в R1.

8. RELEASE - завершить ввод/вывод.
При помощи этого примитива обработчик прерываний прекращает обмен данными с устройством ввода/вывода. Устройство переводится из состояния "занято" ('B') в состояние "присоединено к задаче" ('A'). Если к этому времени от задачи уже поступил запрос на отсоединение устройства, то Q действительно выполняет эту функцию.
Аргументы: (1) указатель устройства (в R1 при прямом вызове), (2) не используется.

Следующие два примитива используют системный таймер и позволяют организовать простую и условную приостановку задач. Если

первый аргумент нуль, то задача приостанавливается (состояние 'C') на интервал времени, определяемый вторым аргументом. Если первый аргумент является указателем присоединенного устройства, то приостановка задачи произойдет, если только устройство находится в состоянии "занято" ('B').

9. DELAYTIC
Аргументы: (1) указатель устройства или нуль, (2) интервал в единицах таймера (20 мсек.).

10. DELAYSEC
Аргументы: (1) указатель устройства или нуль, (2) интервал в секундах.

В обоих приведенных выше примитивах максимально допустимый интервал равен 2^{16} - 1 единиц таймера (около 22 мин.).

Следующие три примитива управляют активностью задач.

11. HOLDTASK - остановиться.
Q останавливает задачу при условии, что другая задача к этому времени не потребовала активизации данной задачи при помощи примитива RUNTASK. В последнем случае остановки задачи не будет.
Аргументы: (1) указатель задачи или нуль, (2) не используется.

Если задан указатель устройства, то по окончании ввода/вывода на этом устройстве произойдет активизация данной задачи. Устройство должно заранее быть присоединеным к данной задаче. Указатель устройства заносится в WAITADD дескриптора данной задачи. Задача останавливается (переводится в состояние 'H').

12. RUNTASK - активизировать (включить) задачу.
Q активизирует (переводит в состояние 'A') остановленную задачу, определенную первым аргументом. Если задача уже активна, то требование активизации запоминается, и немедленная активизация осуществляется в момент последующего вызова HOLDTASK для остановки задачи.
Аргументы: (1) указатель задачи (в R1 при прямом вызове), (2) не используется.

13. TERMINATE - выключить задачу.
Q прекращает работу задачи и автоматически освобождает все используемые ею ресурсы. При отсоединении устройства там, где это предусмотрено при генерации Q, выполняются процедуры восстановления устройства.
Аргументы: (1) указатель задачи или нуль, (2) 2 знака ASCII или нуль.

Если первый аргумент - нуль, то выполнение обратившейся задачи прекращается. Если второй аргумент равен нулю, то выполняется простое выключение указанной задачи. Иначе 2 знака ASCII интерпретируются как код аварийного выключения задачи. Этот код можно выбирать произвольно. Важно только, чтобы он не совпадал с системными кодами аварийного выключения.

Приложение 2

Генерация операционной системы G

Процесс генерации состоит в компиляции текстового файла FILE.DEF (вместо FILE имя файла, определяемое пользователем), описывающего конфигурацию ОС Q на простом непроцедурном языке. На выходе компилятора QGEN получаются два файла: FILE.STS - объектный модуль, включаемый затем в сборку первым, и FILE.LST - листинг компиляции. Ниже приводится краткое описание языка.

1. **Формат операторов.** Каждый оператор состоит из трех полей, из которых первое может быть пустым.

1.1. **Поле метки.** Если не пусто, то должно начинаться с первой позиции строки и заканчиваться двоеточием (:). Метка позволяет организовать ссылки из других операторов на помеченный оператор.

1.2. **Поле команды.** Всегда не пусто и не должно начинаться с первой позиции. Поле команды заканчивается первым встретившимся символом "пробел". Поле команды заполняется одним из ниже приведенных ключевых слов.

1.3. **Поле аргументов.** Следует через один или несколько пробелов после поля команды и присутствует всегда. Аргументы вместе с командами подробно описаны ниже. Синтаксически поле аргументов состоит из элементов следующего вида:

```
        ARG
или     ARG = <значение>
или     ARG = <строка знаков>
или     ARG = <символическое имя>
```

ARG - это ключевое слово, определяющее аргумент, поэтому порядок расположения аргументов не важен. <Значение>, <строка знаков> или <символическое имя> выбираются пользователем. Значение может быть десятичным, восьмеричным или шестнадцатиричным. Представление по умолчанию десятичное. Остальные обозначаются следующим образом:

```
$NNNNNNN        для восьмеричного       (0 <= N <= 7),
#NNNN           для шестнадцатиричного  (0 <= N <= F).
```

Строка знаков определяется контекстуально. В тех случаях, когда строка содержит разделители (пробелы или запятые), ее следует заключить в апострофы (').

2. **Продолжение.** Элементы поля аргументов разделяются запятыми. Пробелы, не являющиеся элементами строки знаков, игнорируются. Поэтому продолжение поля аргументов в следующей строке исходного текста устанавливается автоматически, если последний элемент в строке заканчивается запятой.

3. **Сообщения об ошибках**, производимые компилятором QGEN, помещаются в листинг и бывают двух типов:

- синтаксические ошибки, выводимые вместе с текстом операторов;

- синтаксические ошибки, которые выявляются на втором проходе компилятора.

Пользователю следует внимательно следить, чтобы его листинг не содержал сообщений об ошибках.

Объявление задачи

Формат оператора:

```
<метка>: TASK NAME = <строка>, STARTADD = <имя>,
              PRIORITY = <значение>, STACKSIZE = <значение>,
              TCBSIZE = <значение>, TCB0 = <имя>, HOLD
```

QGEN создает структуру данных задачи, состоящую (в порядке размещения в оперативной памяти) из стека задачи (размер по умолчанию - 13 слов), дескриптора задачи (17 слов) и локальной памяти задачи (размер по умолчанию - 1 слово).

Аргументы:

NAME - определяет символическое имя задачи, которым данная задача может отличаться от других задач. Это имя выводится в списке задач по запросу 'DI' задачи KEYTASK.

STARTADD - задает имя глобальной процедуры, являющейся "входом" в задачу. Если процедура "чистая" (не имеет собственных состояний), то она может использоваться несколькими идентичными задачами.

PRIORITY - этот аргумент определяет положение задачи в списке, используемом диспетчером. <Значение> служит для обозначения приоритета в диапазоне от 0 до 127. Диспетчер всегда просматривает список задач, начиная с задачи с наивысшим приоритетом. Если несколько задач объявлены с одинаковым приоритетом, то они включаются в список в порядке объявления.

STACKSIZE - определяет размер стека задачи в словах.

TCBSIZE. Задачи, использующие "чистые" процедуры, нуждаются в локальной рабочей области, именуемой ниже TCB (это не символическое имя). Размер этой области в словах определяется значением этого аргумента. Задачам, связываемым с терминалом, требуется лишь одно слово TCB, гарантируемое по умолчанию. Использование остальных резервируемых слов TCB не ограничено.

TCB0 - необязательный аргумент, позволяющий при генерации Q заполнить первое слово TCB указателем терминала, который используется задачей. Это необходимо для обращения к терминальному сервису Q. Имя должно быть меткой оператора объявления терминала или устройства, но не значением.

HOLD - этот необязательный аргумент означает, что при начальном пуске Q данная задача будет находиться в неактивном состоянии. Точно так же, в случае нормальной или аварийной остановки, данная задача будет оставаться в неактивном состоянии. Если же этот аргумент опущен, то во всех перечисленных случаях задача будет автоматически выполняться с начала, определяемого аргументом STARTADD.

Объявление устройства

Формат оператора:

```
<метка>: USERDEVICE NAME = <строка>, SIZE = <значение>,
         ITHAND = <имя>, ITVECT = <значение>, CLEANSEQ = <имя>,
                BUSADDR0 = <значение>, BUSADDR1 = <значение>,
                BUSADDR2 = <значение>, BUSADDR3 = <значение>
```

QGEN создает структуру данных устройства, состоящую из дескриптора устройства (8 слов) и локальной памяти устройства (по явному определению).

Аргументы:

NAME служит для объявления имени устройства. Определяемое строкой до 8 знаков имя выводится задачей KEYTASK по команде с клавиатуры терминала 'DE'. Доступ к устройству по имени возможен из задач.

SIZE - размер (в словах) локальной памяти устройства.

ITHAND - это необязательный аргумент. Он определяет символическое имя обработчика прерывания, связанного с определенным вектором прерывания ЭВМ. QGEN организует ссылку на внешнюю глобальную процедуру с указанным именем.

ITVECT - необязательный аргумент, определяющий значение адреса вектора прерываний.

CLEANSEQ - этот необязательный аргумент определяет имя процедуры восстановления. Процедура восстановления - это программа, которая будет выполняться всякий раз при нормальной или аварийной остановке задачи, к которой присоединено устройство. QGEN автоматически обеспечивает необходимую ссылку на внешнюю процедуру с указанным именем. Процедура восстановления вызывается также при начальном или повторном пуске системы Q. Процедура восстановления не должна изменять содержимое регистра R1, а также вызывать монитор и системные процедуры.

BUSADDR0 - BUSADDR3 - это необязательные аргументы, которые структуру данных устройства минимальной длины 16 байтов дополняют словами с определяемыми значениями адресов, связанных с устройством.

Замечание: вектор прерывания не будет определен, если пропущен хотя бы один из аргументов 'ITHAND' или 'ITVECT'.

Объявление терминала

Формат оператора:

```
<метка>: TERMINAL NAME = <строка>, TYPE = <имя>, CONSOLE,
                BUSADDR = <значение>,  ITVECT = <значение>,
                WAITPAGING, KEYTASK, KEYSTACK = <значение>,
                KEYPRIORITY = <значение>,  INITOPT = <имя>,
                PAGELENGTH = <значение>, LINELENGTH = <значение>
```

В отличие от устройства, терминал характеризуется двумя смежными векторами прерывания. Дескриптор терминала имеет унифицированную структуру, совпадающую в младших адресах со структурой дескриптора устройства, но дополненную локальными данными, одинаково организованными для терминалов различных типов. Это позволяет ограничиться процедурным модулем QT для всех объявленных терминалов.

Аргументы могут принимать следующие значения.

NAME - этот аргумент объявляет имя терминала. Имя может быть длиной до 8 символов. Это имя выводится в списке устройств по запросу с клавиатуры 'DE' к интерпретатору терминальных команд.

TYPE - устанавливает тип терминала. <Имя> может быть одним из следующего списка.

TTY	- 72-колонный телетайп;
LA36	- 132-колонный телетайп фирмы DEC;
VT340	- алфавитно-цифровой дисплей "Видеотон-340";
GENERAL	- японский алфавитно-цифровой дисплей KDE;
VT50	- алфавитно-цифровой дисплей фирмы DEC;
TEKTRONIX	- графический дисплей (4010,4012 или 4014).

CONSOLE - это необязательный аргумент, определяющий данный терминал пультовым, на который выводятся все системные сообщения. Только один терминал можно определить с этим аргументом.

BUSADDR - помещает в дескриптор значения адресов 4-х интерфейсных регистров терминала. <Значение> соответствует первому из этих регистров (состояние ввода).

ITVECT - задает адрес первого из двух смежных векторов прерываний.

WAITPAGING - этот аргумент нужен для тех терминалов, которые не могут остановить передачу данных во время стирания экрана. Необходимый тайм-аут организован с помощью системного таймера.

KEYTASK - необязательный аргумент, приводящий к генерации задачи - интерпретатора терминальных команд с клавиатуры этого терминала.

KEYSTACK - для задачи KEYTASK по умолчанию выделяется стек размером в 20 слов. <Значение>, соответствующее числу слов, позволяет определить другую длину стека.

KEYPRIORITY - необязательный аргумент, позволяющий заменить значение по умолчанию 4 приоритета задачи KEYTASK на значение этого аргумента. Этот аргумент интерпретируется только при наличии аргумента KEYTASK.

INITOPT - необязательный аргумент, определяющий символическое имя программы, которая вызывается в начале работы задачи KEYTASK. Поэтому вместе с этим аргументом необходимо объявление KEYTASK.

PAGELENGTH - необязательный аргумент, позволяющий определить явно число строк на экране терминала.

LINELENGTH - необязательный аргумент аналогичного действия по отношению к длине строки терминала, обычно определяемой по умолчанию.

Объявление параметров ОС Q

Формат оператора:

```
SYSTEM NAME = <строка>, MCNAME = <строка из 4-х букв или цифр>,
       SCHEDTIME = <значение>, MAXCORE = <значение>,
       ENDOFTASKS = <значение>, TRACE = <значение>, K150
```

Цель состоит в определении различных системных параметров. Каждый аргумент является необязательным.

Значения аргументов:

NAME - определяет строку длиной до 255 знаков, которая выводится задачей KEYTASK на терминал при начальном пуске Q.

MCNAME - определяет 4-символьное имя микроЭВМ (зарезервировано для использования в распределенных системах).

SCHEDTIME - служит для определения частоты циклической перестановки в списке задач одинакового приоритета. <Значение>, равное нулю или 255, соответствует выключенному механизму циклической перестановки. <Значение> между этими величинами задает интервал в единицах системного таймера (20 мсек.) между последовательными включениями механизма.

MAXCORE - При начальном пуске Q очищается область динамической памяти, которая по умолчанию находится между последним адресом, переданным при загрузке, и максимальным адресом 28К. Q автоматически находит верхнюю границу существующей памяти. <Значение> используется для явного ограничения этой области сверху.

ENDOFTASKS - А этот аргумент определяет нижнюю границу динамической памяти.

TRACE - при крахе Q полезно иметь информацию о последовательности событий, которые предшествовали этому состоянию. Каждая системная процедура, управляющая последовательностью работы задач, при вызове оставляет след в трассировочном кольцевом буфере. Весь трассировочный буфер в обратном хронологическом порядке выводится на пультовой терминал при крахе ОС Q. Каждый элемент буфера состоит из трех слов. <Значение> определяет число элементов этого буфера.

K150 - регистрирует наличие в ОС Q контроллера К-150 для сохранения его состояния в стеках задач при переключениях, а также для ограничения по умолчанию 24К верхней границы динамической памяти. Без этого аргумента ограничение по умолчанию - 28К.

Объявление команд задачи KEYTASK

Формат:

```
KEYOPTS <имя>,<имя>, ...
```

Этот оператор определяет перечень всех команд, вводимых с клавиатуры терминала и интерпретируемых задачей KEYTASK. Каждое имя должно быть именем глобальной процедуры, вызываемой задачей KEYTASK. Во всех случаях для вызова команды пользователю достаточно напечатать только первые 2 знака соответствующего <имени>.

Пример

```
/ Действующий стенд для исследования импульсных генераторов УНК.
/              Лаборатория источников питания УНК.
/ Пультовой терминал: \
    TERMINAL    TYPE=TTY, ITVECT=$60, BUSADDR=$177560,
                CONSOLE, KEYTASK, PAGELENGTH=24,
                LINELENGTH=78, KEYPRIORITY=10,
                NAME=MONITOR
/ Линия связи с ЭВМ коллективного пользования: \
    USERDEVICE NAME=HOSTLINK, ITVECT=$404, CLEANSEQ=RESETHOST,
                ITHAND=HOSTHANDLER, BUSADD0=$141000,
                BUSADD1=$141020, BUSADD2=$161000, SIZE=5
/ Псевдоустройство для соединения пультового терминала с ЭВМ
/    коллективного пользования в режиме "GENERAL": \
    USERDEVICE NAME=TTY0LINK, ITVECT=$230, CLEANSEQ=RESETTTY,
                ITHAND=TTY0HANDLER, BUSADD0=$177560,
                BUSADD1=$177562, BUSADD2=$177560,
                BUSADD3=0, SIZE=5  / 0 для пультового терминала \
/ Линия связи с удаленным дисплеем СМ-7209 или ЭВМ ДВК-3
/ для их соединения с ЭВМ коллективного пользования: \
    USERDEVICE NAME=TTY1LINK, ITVECT=$504, CLEANSEQ=RESETTTY,
                ITHAND=TTY1HANDLER, BUSADD0=$161000,
                BUSADD1=$161020,   BUSADD2=$161000,
                BUSADD3=1, SIZE=5  / 1 для дисплея СМ-7209 \
/ Псевдоустройство для передачи буферов данных между задачами \
/     (Замечание: SIZE = 3 + число используемых буферов):     \
DAT:USERDEVICE NAME=DATABOX, SIZE=7, CLEANSEQ=RESETBOX
/ Устройство синхронизации измерений: \
    USERDEVICE NAME=SYNCRO, ITVECT=$530,
                CLEANSEQ=SYNCRESET, ITHAND=RUNHUNDLER
/ Устройство сброса счетчиков между измерениями: \
    USERDEVICE NAME=COUNTER, ITVECT=$470,
                ITHAND=C60HAN, CLEANSEQ=C60RESET
/ Устройство регистрации ошибок аппаратуры КАМАК: \
ERR:USERDEVICE NAME=INT400, CLEANSEQ=CLEANERROR, SIZE=3,
                ITVECT=$400, ITHAND=CAMERROR
/ Параметры операционной системы: \
    SYSTEM      MAXCORE=$116000, TRACE=10, SCHEDTIME=5,
                NAME='- UNK PULSE GENERATORS INVESTIGATION. '
/ Набор команд монитора, доступных с пультового терминала: \
    KEYOPTS     DIOPT, DEOPT, TIOPT, DBOPT, PROPT, TEOPT,
                DUOPT, MOOPT, REOPT, STOPT, CHECK, SYNCHECK,
                KILL, SHOW, RUNMEAS, HISTO, HELP, HOSTTERMINAL
/ Процесс сбора данных: \
    TASK        NAME=MEASURE, PRIORITY=60, TCB0=DAT,
                STARTADD=MEASURE, STACKSIZE=30, HOLD
/ Процесс управления сбором данных и выдачей результатов: \
    TASK        NAME=IGNITRON, PRIORITY=50, TCBSIZE=18,
                STARTADD=IGNITRON, STACKSIZE=30
/ Процесс, управляемый с терминала TTY1: \
    TASK        NAME=DEC10TTY, PRIORITY=10,
                STARTADD=TTY1TERMINAL, STACKSIZE=30
```

Модули типовых процедур

Модуль QI. Системное информационное обслуживание

GETTASK - процедура, выдающая адрес дескриптора задачи и адрес ее рабочей области.
Вход: R4 - адрес строки длиной 8 знаков - символического имени задачи. Если R4 = 0, то подразумевается текущая задача.
Выход: R2 - адрес дескриптора задачи, или R2 = 0, если нет такой задачи,
R4 - адрес рабочей области (TCB) выполняющейся задачи.

GETTCB - процедура, выдающая в регистре R4 адрес TCB.

GETDEV - процедура, выдающая адрес дескриптора устройства.
Вход: R4 - адрес строки длиной 8 знаков - символического имени устройства.
Выход: R4 - адрес дескриптора или нуль, если нет устройства.

DEVIND - процедура, выдающая порядковый номер данного устройства в списке устройств системы.
Вход: R4 - адрес дескриптора устройства.
Выход: R4 - порядковый номер или -1, если нет такого устройства.

GETTIME - процедура, выдающая текущее время в виде строки знаков вида: чч:мм:сс
Вход: R4 - адрес буфера длиной 8 знаков для результата.
Выход: R4 - адрес буфера, увеличенный на 8,
R5 - не определен.

SETTIME - процедура для установки значения датчика времени суток по заданной строке знаков вида: чч/мм/сс
Вход: R4 - адрес строки знаков.
Выход: R4 - адрес буфера, увеличенный на 8.

SETPRT - процедура изменения приоритета задачи.
Вход: R4 - адрес дескриптора задачи,
R5 - приоритет.
Выход: R4 - не определен. Ничего не происходит, если не заданы дескриптор или значение приоритета.

SETABORT - процедура для замены системной процедуры аварийного завершения данной задачи на другую процедуру, адрес которой задан в регистре R5.

RESETABORT - процедура восстановления системной процедуры аварийного завершения данной задачи.

Модуль QU. Вспомогательные процедуры

DECTOBIN - процедура преобразования числа, заданного строкой знаков в десятичной системе счисления, во внутреннее (двоичное) представление.
Вход: R4 - указатель на строку знаков.
Выход: R5 - число (в случае переполнения R5 = -1),
R4 - указатель на закрывающий символ, т.е. не цифру.

OCTTOBIN - процедура преобразования во внутреннее (двоичное) представление числа, заданного строкой знаков в восьмеричной системе счисления.
Вход: R4 - указатель на строку знаков.
Выход: R5 - число (переполнение исключено, так как берется не более 6 знаков),
R4 - как и в DECTOBIN.

DECODE - процедура, которая входную строку знаков преобразует к специальному виду для облегчения ее дальнейшего разбора.
Вход: R3 - определяет систему счисления по умолчанию для чисел, заданных в исходной строке: R3 = 0 - десятичная система, R3 = 1 - восьмеричная система (для ввода чисел в другой системе счисления используются специальные знаки перед числом: $ - для восьмеричной и @ - для десятичной системы счисления),
R4 - указатель на входной буфер,
R5 - указатель на выходной буфер (четный адрес).
Выход с неизменными регистрами.

BINTODEC - процедура преобразования числа из двоичного в десятичное знаковое представление.
Вход: R3 - размер поля для преобразованной строки знаков: если R3 < 0, то старшие незначащие разряды заменяются символом '0', если R3 > 0, то старшие незначащие разряды заменяются пробелами, если R3 = 0, то выводимое число выравнивается по левой границе поля;
R4 - указатель на выходной буфер (сдвигается при выходе из процедуры);
R5 - преобразуемое число.
Выход с преобразованным числом (возможно, со знаком '-') в выходном буфере. В случае недостаточного размера поля оно автоматически заполняется знаками '*'.

BINTOOCT - процедура преобразования числа из двоичного в восьмеричное знаковое представление.
Вход: R4 - указатель на буфер для строки знаков,
R3 - длина поля,
R5 - преобразуемое число.
Выход: R4 - сдвинутый указатель, R3 и R5 не изменяются.

Процедуры для редактирования буферов.

BLANKFILL заполняет буфер пробелами или любыми другими символами, определяемыми по регистру R5.

Вход: R4 - адрес буфера,
R3 - его длина,
R5 содержит символ-заполнитель (R5 = 0 соответствует заполнению пробелами).

Выход: R4 сдвинут на значение R3. Остальные регистры не изменяются.

MOVETEXT копирует содержимое одного буфера в другой буфер.

Вход: R4 - указатель выходного буфера,
R5 - указатель входного буфера,
R3 - число символов.

Выход: R4 и R5 сдвинуты, а R3 сохраняется.

SKIPBLANK сдвигает указатель через ведущие пробелы.

Вход: R4 - указатель,
R3 - максимальная величина сдвига. Если R3 <= 0, то максимальная величина сдвига равна 6.

Выход: R4 - сдвинутый указатель,
R5 - число пропущеных позиций,
R3 не изменяется.

Модуль QF. Форматирование и вывод данных

FORMAT - процедура, объединяющая функции формирования строки знаков по заданному формату и списку аргументов с последующим выводом этой строки на присоединенный или пультовой терминал.

Вход: R3 - указатель на строку формата,
R4 - указатель на строку выводимых знаков,
R5 - указатель на список аргументов.

Выход: R4 указывает на текущую позицию в выходном буфере, а если был указан вывод или перевод строки, то на начало буфера;
R5 содержит код завершения (0, если все нормально, и -2, если вывод прерван с клавиатуры с помощью CTRL-A).

Строка формата состоит из элементов, состоящих из одного или двух слов. Первое слово указывает что нужно делать. Если действие предполагает операнд, то в слове содержится также информация о типе адресации и, если необходимо, о длине поля для операнда. Второе слово - это непосредственный операнд, или смещение в строке аргументов, или адрес операнда.

Мнемоника кодирования операций форматирования представлена в файле QUSER.INC. Коды образуют две группы.

1. Операции, использующие операнд.
FOTEXT - скопировать в выходной буфер строку знаков;
FOTEXTC - скопировать строку знаков, используя счетчик из первого байта строки;

FODEC - преобразовать операнд в число в десятичном представлении и поместить результат в поле заданной длины в выходном буфере;
FODECB - аналогично FODEC, но с байтовым операндом;
FOOCT - преобразовать операнд в число в восьмеричном представлении и поместить результат в поле заданной длины в выходном буфере;
FOFILL - заполнить поле заданной длины заданным кодом ASCII;
FOBNAM - заполнить поле заданной длины именем дескриптора задачи или устройства, представленным операндом.

2. Операции, не использующие операнд.
FOFIN - окончить форматирование (последнее действие);
FOPRINT - вывести буфер на присоединенный терминал;
FOMESS - срочно вывести буфер на пультовой терминал;
FONEWL - перевести строку;
FONEWP - стереть экран (новая страница);
FOTIME - записать в выходной буфер текущее время;
FOBLANK - заполнить поле заданной длины пробелами;
FONAME - заполнить поле заданной длины именем этой задачи.

Мнемоника типа адресации (тип по умолчанию - непосредственный операнд: значение или адрес текста):
FOINDEX - операнд задан смещением в строке аргументов,
FOINDIR - операнд предполагает косвенную адресацию;
FOINDIREX - смещение + косвенная адресация.

Пример

```
ARRAY BYTE TEXT = ' X = ';  INTEGER X = -123;
ARRAY 80 BYTE BUF;
ARRAY LOGICAL FORM =
  ( FOTIME,
    FOTEXT + LENGTH(TEXT), REF(TEXT),
    FODEC  + 4 + FOINDIR,  REF(X),
    FONEWL,  FOPRINT,      FOFIN   );
REF(FORM) => R3;  REF(BUF) => R4;  FORMAT;
```

Будет выведено:

```
07*45*00 X = -123
```

Модуль QT. Интерфейс с терминалом

KREAD - процедура ввода с терминала.
Вход: R4 - указатель на входной буфер,
R5 - длина буфера.
Выход: R5 - число введенных байтов (в конце ввода символ CR),
-1 в случае ввода CTRL-C или -2, если введен CTRL-A.

TWAIT - процедура ожидания окончания ввода/вывода. Ее рекомендуется помещать после процедуры KREAD перед обращением к буферу ввода.

PRINT - процедура вывода на терминал.
Вход: R4 - указатель на строку знаков,
R5 - число выводимых знаков.

NEWPAGE - процедура гашения экрана дисплея. В случае телетайпа пропускается одна строка.

NEWLINE - процедура перевода строки.

SETBITS - процедура установки некоторых битов в дескрипторе присоединенного терминала. Мнемонические имена битов определены в файле QUSER.INC:

ECHOINGX - режим "эха" вводимых символов,
NOWPAGINGX - включение таймаута при гашении экрана,
ABORTABLEX - блокировка аварийного окончания задачи.

Вход: R5 содержит устанавливаемые биты.

CLEARBITS - процедура обнуления перечисленных битов.
Вход: R5 содержит обнуляемые биты.

Замечание. Для вызова всех приведенных выше процедур, а также процедуры FORMAT с выводом на терминал в дескрипторе задачи по адресу TCB0 должен находиться адрес дескриптора терминала.

MESSAGE - процедура вывода срочных (внеочередных) сообщений на пультовой терминал. Сообщение длиной до 72 знаков копируется в буфер системного псевдоустройства MESSBOX, которое обеспечивает задержку срочных сообщений от других задач в случаях столкновений.

Вход: R4 - адрес сообщения,
R5 - его длина (если больше физической длины объявленного пультового терминала или 72, то "лишние" байты не выводятся).
Выход: R5 неопределен, остальные регистры не изменяются.

В этом файле имеются также 2-й и 3-й модули, содержащие обработчики прерываний и процедуры восстановления, которые запрашивает QGEN при объявлении терминалов.

Модуль QK. Основной набор процедур задачи KEYTASK

В этом файле находится процедура KEYTASK, являющаяся основанием расширяемого интерпретатора команд с клавиатуры пультового терминала, и процедуры, реализующие отдельные функции. Эти процедуры приведены здесь вместе с форматом соответствующих терминальных команд, которые они реализуют. При этом аргумент "имя_задачи" запоминается в рабочей области KEYTASK, и в дальнейшем его можно опускать при вызове других команд.

ACOPT - эта процедура активизирует задачу:
AC имя_задачи
EXOPT активизирует задачу и передает ей пультовой терминал:
EX имя_задачи

TEOPT прекращает выполнение задачи (переводит ее в исходное состояние):

TE имя_задачи

PROPT изменяет приоритет задачи:

PR номер_приоритета имя_задачи

DIOPT выдает на терминал полный список задач. Вместе с именем задачи выводятся также ее текущее состояние, приоритет и адреса дескриптора задачи (он же является основанием стека задачи) и начала рабочей области:

DI

DEOPT выдает на терминал полный список устройств. В каждой строке выводится адрес дескриптора устройства, имя устройства, имя связанной с устройством задачи или пробел, состояние устройства и его тип.

DE

Модуль QKT. Процедура TIOPT интерпретатора

Процедура TIOPT служит для индикации (вызов без параметра) и для установки значения системного таймера:

TI чч:мм:сс

Модуль QDB. Процедуры, полезные при отладке

REOPT выдает содержимое общих регистров заданной задачи:

RE имя_задачи

STOPT выдает содержимое стека заданной задачи:

ST имя_задачи

DUOPT выдает содержимое заданной области ОЗУ или регистра внешнего устройства (в восьмеричном представлении):

DU начальный_адрес конечный_адрес

или DU начальный_адрес+число_слов

или DU адрес

MOOPT изменяет содержимое ячейки ОЗУ или регистра внешнего устройства:

MO адрес значение

DBOPT устанавливает контрольную точку в выполняемую задачу:

DB адрес имя_задачи

Задача останавливается по заданному адресу, а после активизации с контрольной точки полностью восстанавливается модифицированная ячейка программы.

Модуль ARITHM. Процедуры арифметического процессора

Процедуры этого модуля предназначены для удобства составления программ, использующих арифметические действия с вещественными числами. Выполнение этих процедур возможно лишь на тех УВМ, процессор которых может выполнять команды FIS расширенного набора арифметики вещественных чисел. Заметим, что все остальные модули Q не имеют этого ограничения и могут функционировать на любой ЭВМ семейства PDP-11 и LSI-11.

BINTOFL преобразует целое число в вещественное число.
Вход: R4 - адрес вещественного числа (2 слова),
R5 - целое число со знаком.
Выход: R4 и R5 не определены.

DBINFL преобразует целое число двойной точности в вещественное число (во внутреннем представлении).
Вход: R0 - старшие разряды целого числа,
R1 - младшие разряды целого числа.
Выход: R0 - старшие разряды вещественного числа,
R1 - младшие разряды вещественного числа,
R5 - порядок вещественного числа.

FLTOASC преобразует вещественное число из внутреннего представления в знаковое представление числа.
Вход: R3 - число знаков мантиссы,
R4 - указатель на строку знаков,
R5 - адрес вещественного числа.
Выход: R3 - число незначащих нулей мантиссы и ведущих пробелов,
R4 - сдвинутый указатель,
R5 0, если нет ошибок, иначе -1.

ASCTOFL преобразует вещественное число, заданное строкой знаков (в формате 'F') во внутреннее представление числа.
Вход: R3 - адрес результата преобразования,
R4 - указатель на строку знаков.
Выход: R3 не изменяется,
R4 - сдвинутый указатель,
R5 -1, если неправилен формат или диапазон числа не соответствует формату, иначе 0 (все нормально).

FLADD - процедура сложения двух вещественных чисел.
Вход: R3 и R4 - указатели слагаемых.
Выход: R3 и R4 не изменяются, R5 не определен. Сумма замещает слагаемое по адресу R4.

FLSUB - процедура вычитания вещественных чисел.
Вход: R3 - указатель вычитаемого,
R4 - указатель уменьшаемого.
Выход: R3 и R4 не изменяются, R5 не определен. Разность замещает уменьшаемое.

FLMUL - процедура умножения двух вещественных чисел.
Вход: R3 и R4 - указатели сомножителей.
Выход: R3 и R4 не изменяются, R5 не определен. Произведение замещает сомножитель по адресу R4.

FLDIV - процедура деления вещественных чисел.
Вход: R3 - указатель делителя,
R4 - указатель делимого.
Выход: R3 и R4 не изменяются, R5 не определен. Частное замещает делимое.

Ю.В.Куянов.
Q- операционная система для управляющих ЭВМ.
Редактор Н.В.Ежела. Технический редактор Л.П.Тимкина.

Подписано к печати I.06.88. Т-II686. Формат 60x90/I6.
Офсетная печать. Печ.л.I,75. Уч.-изд.л. 2,23. Тираж 250.
Заказ 558. Индекс 3622. Цена 33 коп.

Институт физики высоких энергий, I42284, Серпухов Московской области.

33 коп. Индекс 3622.

П Р Е П Р И Н Т 88-I04, И Ф В Э, I988