терпретируемому виду, а также средствами их разметки (в основном для конструкций «форма» и «элементарная функция»).

На втором этапе при необходимости изменяется состав базовых функциональных форм. Подключение имеющихся форм производится автоматически. Новые формы (как и определения рекурсивных функций, задающие схему вычислений подобно ALL или ITR) добавляются к рекурсивной программе управления выполнением функциональных выражений [5], предоставляющей средства для ведения динамической памяти. Особенности реализации базовой системы связаны с возможностью использования автоматического и «ручного» управления памятью, сочетанием списочного и линейного размещения данных.

Элементарные функции объединены в библиотеки подпрограмм обработки экспериментальных данных и взаимодействия с используемыми модулями КАМАК. Настройка на обращение к новой функции осуществляется расширением таблицы функций интерпретатора: Дополнительно в этой функции требуется обеспечить прием стандартных управляющих параметров.

Предоставляются средства отладки перечисленных компонент разрабатываемой системы, включая трассировку основных структур данных (стек автомата, таблицы разметки, стек интерпретатора форм, параметры базовых функций).

ЛИТЕРАТУРА

1. **Backus J.** Can programming be liberated from von Neumann style? A functional style and its algebra of programs.— Comm. ACM, 1978, v. 21, N 8, p. 613—641.
2. **Backus J.** The algebra of functional programs: Functional level reasoning, linear equations and extended definitions.— Lect. Notes in Comp. Sci., 1981, v. 107, p. 1—43.
3. **Куликов В. В., Скобелев П. О.** Система автоматизированного построения проблемно-ориентированных диалоговых процессоров.— В кн.: Интерактивные системы: Материалы V школы-семинара. Тбилиси: Мецниереба, 1983.
4. **Куликов В. В., Скобелев П. О.** Об одном алгоритме построения $LR(k)$-анализаторов.— В кн.: Автоматизация научных исследований. Куйбышев, 1984.
5. **Inone K., Tanizawa T., Taniguchi K., Okamoto T.** Implementations of a functional programming language FPL.— Systems Computers Control., 1982, v. 13, N 3, p. 1—10.

Поступила в редакцию 20 марта 1986 г.

УДК 681.3.06

Р. ВЕРБОВА, А. НЯГОЛОВ, О. ПОПОВ, Д. СОФКОВА

(София, Болгария)

ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ МИКРОКОМПЬЮТЕРНОЙ СИСТЕМЫ АВТОМАТИЗАЦИИ ЦЛАНП 0270

1. Общее описание микрокомпьютерной системы ЦЛАНП 0270. Микрокомпьютерная система автоматизации ЦЛАНП 0270 является двухкрейтовой системой модульного типа в стандарте КАМАК [1, 2]. Ее можно применять для решения широкого круга задач, связанных с созданием проблемно-ориентированных комплексов автоматизации научных исследований и технологических процессов.

Состав технических средств базовой конфигурации включает микрокомпьютер, крейт-контроллеры двух крейтов, оперативную память до 48 Кбайт, модуль управления клавиатурой (дисплеем) и модуль управления гибкими магнитными дисками типа ЕС 5014 [3].

Использование мультипроцессорной архитектуры в значительной степени повышает эффективность системы [4]. Основной микропроцессор обеспечивает общее управление системой и обработку данных. Управление модулями в крейте осуществляется специализированным КАМАК-процессором. Для каждого КАМАК-процессора выделена область памяти с двойным доступом — со стороны основного микропроцессора и со стороны КАМАК-процессора соответствующего крейта (память КАМАК-программ), в которую основной микропроцессор заносит КАМАК-команды или цепочки КАМАК-команд, выполняемые специализированными КАМАК-процессорами. Кроме специализированного КАМАК-процессора, в состав крейт-контроллера микрокомпьютерной системы ЦЛАНП 0270 входит процессор прерываний, предназначенный для векторизации и обслуживания запросов прерывания от всех КАМАК-модулей соответствующего крейта.

Программное обеспечение системы ЦЛАНП 0270 включает резидентный монитор, операционную систему на гибких магнитных дисках, трансляторы с языков высокого уровня, набор драйверов, кросс-средства.

2. Монитор и операционная система СДОС. 2.1. Резидентный монитор предназначен для управления минимальной конфигурацией системы и выполнения следующего набора основных функций: а) проверка и обновление содержимого оперативной памяти и программно-управляемых регистров; б) запуск пользовательской программы; в) обработка программных прерываний; г) обмен информацией с внешними устройствами; д) загрузка дисковой операционной системы.

Резидентный монитор обеспечивает выполнение тестовых процедур и некоторых отладочных функций.

2.2. Операционная система на гибких магнитных дисках СДОС обеспечивает разработку, отладку и выполнение программ для микрокомпьютерной системы ЦЛАНП 0270. Ее структура включает ядро, содержащее основные функции системы, и набор перекрывающихся программ, предназначенных для выполнения отдельных функций.

Ядро операционной системы содержит программы инициализации, набор системных функций, программы управления поддерживаемыми периферийными устройствами.

В состав операционной системы СДОС включен и набор системных функций, осуществляющий унифицированный (независимый от конкретных устройств) ввод (вывод), обмен информацией на физическом уровне между дисковыми файлами и оперативной памятью, операции управления регистрами микропроцессора и обработки символьных строк. Вызов этих функций производится с помощью обращения к супервизору системы. Необходимую для их выполнения информацию нужно заранее занести в программные регистры микропроцессора. Способ реализации вызова системных функций основан на использовании системы обработки программных прерываний микропроцессора. Предусмотрена возможность расширения набора системных функций со стороны пользователя, которая обеспечивает создание оптимального набора для конкретных применений. Использование системных функций позволяет в значительной степени облегчить программирование часто встречающихся операций.

2.3. Управление файлами. Большая часть программ операционной системы СДОС связана с управлением и поддержкой файловой системы на гибких магнитных дисках. Использование СДОС дает возможность осуществлять следующие основные функции управления файлами: вывод информации с каталога файлов; удаление файлов; изменение имен и атрибутов файлов; объединение файлов; различные способы создания и изменения содержимого файлов с помощью программ редактирования, трансляции, физического доступа к абсолютным и объектным файлам; загрузку в оперативную память файлов, содержащих программный код, и их выполнение.

Операционная система СДОС поддерживает следующие типы файлов: текстовые файлы; объектные файлы; абсолютные файлы, содержа-

щие исполняемый программный код; двоично-кодированные файлы в коде ASCII; командные файлы.

3. Управление КАМАК-аппаратурой. Процесс управления КАМАК-аппаратурой в микрокомпьютерной системе ЦЛАНП 0270 сводится к предварительной генерации необходимых КАМАК-команд (или цепочек КАМАК-команд) в памяти КАМАК-программ, после чего производится выполнение требуемых в данный момент КАМАК-команд. Аппаратная возможность отделения процесса генерации КАМАК-команд от процесса их выполнения значительно облегчает работу по решению критических ко времени задач; генерацию КАМАК-команд можно осуществить в инициализирующем участке пользовательской программы, т. е. до начала обработок в реальном масштабе времени.

Это преимущество полностью реализовано в КАМАК-библиотеке Фортрана и в меньшей степени в КАМАК-библиотеке МПЛ. Средства управления КАМАК-аппаратурой на уровне КАМАК-Бейсика удовлетворяют требованиям к КАМАК-системам традиционного типа, т. е. генерацию и выполнение КАМАК-команд нельзя разделить во времени.

3.1. КАМАК-Бейсик. Распространение алгоритмического языка Бейсик в микрокомпьютерных системах обусловливается его доступностью для широкого круга непрофессиональных программистов. В программное обеспечение системы ЦЛАНП 0270 включен интерпретирующий траслятор КАМАК-Бейсик. Интрепретатор работает в диалоговом режиме и снабжен средствами для арифметических операций над числами с плавающей запятой и набором встроенных функций, обеспечивающим обработку символьных данных. Управление КАМАК-модулями реализуется с помощью оператора CAM, имеющего формат

CAM ⟨тип операции⟩ (⟨*F*⟩, ⟨*A*⟩, ⟨*N*⟩ [, ⟨переменная⟩])
⟨тип операции⟩ ⟨переменная⟩

Здесь ⟨тип операции⟩ может принимать следующие значения:

R — чтение (чтение выбранного регистра выбранной станции в поле ⟨переменная⟩);

W — запись (запись содержимого поля ⟨переменная⟩ в указанный регистр указанной станции);

P — выполнение команд, не использующих шин чтения (записи);

E — генерация сигнала *C* (сброс);

Z — генерация сигнала *Z* (инициализация);

Q — установка содержимого поля ⟨переменная⟩ в «0» или «1» в зависимости от состояния сигнала *Q*;

X — установка содержимого поля ⟨переменная⟩ в «0» или «1» в соответствии с сигналом *X*.

При выполнении операций типа *Q* и *X* в специфицированное поле заносится состояние шин после выполнения последней к этому моменту КАМАК-команды.

Поля ⟨*F*⟩ — код КАМАК-функции, ⟨*N*⟩ — номер станции и ⟨*A*⟩ — субадрес содержат информацию в соответствии со стандартом КАМАК.

Поле ⟨переменная⟩ указывает на адрес переменной или элемента массива и используется в соответствии с требованиями конкретной операции.

Оператор ATLAM предназначен для задания информации, необходимой для обработки запросов (LAM) с КАМАК-модулей. Формат оператора следующий:

ATLAM ⟨позиция⟩, ⟨номер оператора⟩.

Здесь ⟨позиция⟩ задает номер станции, на которой находится требуемый модуль, а ⟨номер оператора⟩ указывает на номер строки, с которой начинается соответствующая программа обработки прерывания. Оператор ATLAM должен предваряться разрешением сигнала LAM для соответствующего модуля.

Интерпретирующий транслятор КАМАК-Бейсик предназначен для решения некритических ко времени задач, для настройки модулей и функциональной проверки КАМАК-систем.

3.2. КАМАК-библиотека МПЛ. Алгоритмический язык МПЛ предназначен для широкого круга задач. Пополнение языка КАМАК-библиотекой [5] дает возможность пользователю управлять КАМАК-модулями системы ЦЛАНП 0270 обычными вызовами подпрограмм.

В соответствии с назначением программные модули библиотеки можно сгруппировать следующим образом: декларативные модули (формируют адресные переменные и массивы); операторы единичного действия (создают КАМАК-команды, которые выполняются за один цикл КАМАК-процессора); операторы блочного обмена (осуществляют обмен блоками данных); операторы мультимодульных действий (дают возможность выполнения одной или нескольких КАМАК-функций с массивом КАМАК-адресов и повторения одной и той же КАМАК-команды, элементы которой модифицируются перед каждым выполнением); операторы управления запросами; операторы общего управления КАМАК; операторы управления памятью КАМАК-программ.

КАМАК-библиотека МПЛ обеспечивает возможность генерации как простых (единичных), так и цепочек КАМАК-команд (КАМАК-программ), что позволяет использовать аппаратные возможности системы ЦЛАНП 0270 для одновременной работы основного микропроцессора и специализированного КАМАК-процессора.

3.3. КАМАК-библиотека Фортрана. Алгоритмический язык Фортран находит широкое применение в создании прикладного программного обеспечения в области научных исследований. Адаптация языка Фортран для управления системой ЦЛАНП 0270 выполнена на базе библиотечного расширения [6]. Пользователю предоставляются следующие основные возможности: генерировать КАМАК-команды; управлять доступом к крейт-контроллеру с помощью сигналов сброса, запуска, запрета, инициализации, тестирования; формировать определения, массивы; модифицировать адреса прерываний, форму представления данных, адреса в КАМАК-командах; тестировать поля КАМАК-команд в памяти КАМАК-программ.

Вызов всех подпрограмм КАМАК-библиотеки Фортрана полностью подчиняется синтаксическим правилам языка и не требует дополнительных усилий со стороны прикладного программиста.

4. Управление специализированными арифметическими КАМАК-модулями. Так как в системе команд микропроцессоров практически отсутствуют арифметические операции, целесообразно использование специализированных арифметических устройств. Распространенные трансляторы с языков высокого уровня не располагают средствами управления такими устройствами. В системе ЦЛАНП 0270 предусмотрены программные средства управления специализированным арифметическим процессором на уровне Ассемблера и с помощью библиотеки алгоритмического языка высокого уровня Фортран.

4.1. Интерпретатор на уровне Ассемблера. В состав программного обеспечения ЦЛАНП 0270 включен интерпретатор [7], предназначенный для управления специализированным арифметическим модулем в стандарте КАМАК на базе быстрого арифметического процессора AM 9511. Интерпретатор можно вызывать в любом месте программы на Ассемблере. При описании вычислительных алгоритмов возможно использование операторов типа FOR NEXT, IF THEN ELSE, FORMULA.

Оператор FORMULA задает последовательность действий арифметического процессора в виде таблицы, каждый элемент которой содержит необходимую для процессора информацию: код операции, адреса операндов, индикатор вычисления индексов и т. д. Содержание таблицы интерпретируется в процессе вычисления арифметического выражения.

Интерпретатор облегчает процедуру управления специализированным арифметическим КАМАК-модулем. Его структура может использоваться при модификации существующих трансляторов с языков высокого уровня.

4.2. Библиотека быстродействующей арифметики Фортрана. Из-за небольших вычислительных возможностей микропроцессоров все трансляторы с языков высокого уровня используют пакеты программной арифметики, что ограничивает быстродействие прикладных программ и в большинстве случаев делает их неприменимыми для работы в реальном масштабе времени. Замена пакетов программной арифметики наборами программ, управляющих специализированными арифметическими модулями, приводит к значительному повышению эффективности программ, использующих арифметические операции [8].

Применение пакета быстродействующей арифметики Фортрана позволяет управлять специализированным арифметическим модулем АМ 1, реализованным в стандарте КАМАК на базе быстрого арифметического процессора АМ 9511, и дает возможность выполнять любые программы, написанные на Фортране, в 4—6 раз быстрее.

Во время работы транслятора проводится анализ операторов языка и при необходимости генерируется последовательность обращений к библиотеке программной арифметики, содержащих имена и параметры ее модулей. Соответствующие модули библиотеки быстродействующей арифметики имеют те же имена и подключаются автоматически к тексту программы на этапе связывающего редактирования, если имя библиотеки быстродействующей арифметики указано до упоминания имени стандартной библиотеки Фортрана. Таким способом из одной и той же программы можно получить абсолютный модуль как с быстродействующей, так и с обычной программной арифметикой.

В библиотеку включены следующие группы операций.

1. Арифметические и логические операции. В эту группу входят подпрограммы, реализующие все предусмотренные в Фортране арифметические действия для целых и вещественных чисел. Так как логические операции сводятся транслятором к арифметическим действиям, их быстродействие тоже увеличивается.

2. Вычисление индексов.

3. Встроенные функции. В эту группу входит набор встроенных функций языка Фортран для ЦЛАНП 0270, расширенный дополнительными функциями арифметического процессора АМ 9511 и допускающий как вещественные, так и целочисленные значения аргументов.

5. Средства отладки и драйверы. 5.1. Диалоговый резидентный монитор. Разработка прикладного программного обеспечения систем автоматизации часто связана с применением разных языков для создания конкретных программных систем. Отдельные программные модули реализуются на подходящем для их структуры языке, после чего связываются между собой в общую программу. Этот способ, который в значительной степени облегчает разработку проблемно-ориентированных программных систем, приводит к некоторым затруднениям в процессе их отладки. В случаях работы с перемещаемыми программами необходимо провести дополнительные вычисления, чтобы получить требуемый абсолютный адрес.

Диалоговый монитор для отладки программного обеспечения предназначен для упрощения процесса отладки программ в системе ЦЛАНП 0270 [9]. Монитор обеспечивает контроль работы отлаживаемой программы и выполняет отладочные функции при прохождении выполнения программы через определенные точки. Адреса этих точек и требуемые отладочные функции задаются пользователем с помощью управляющих команд монитора. Вне заданных точек отлаживаемая программа выполняется независимо от монитора.

Диалоговый монитор может работать автономно (находясь в постоянной памяти) или загружаться в оперативную память с помощью дисковой операционной системы. Отладочные функции монитора осуществляются следующими группами команд.

1. Управление памятью. Команда дает возможность вычислить абсолютный адрес в памяти по базовому адресу и по заданному в поле

операнда остальных команд относительному адресу. Команду можно использовать в любой момент работы монитора. Заданный таким способом базовый адрес остается в силе до получения новой команды установки базового адреса. В начале работы подразумевается базовый адрес 0, т. е. все относительные адреса воспринимаются как абсолютные.

2. Контроль выполнения программы. В эту группу входят команды задания контрольных точек, удаления контрольных точек, шагового выполнения отлаживаемой программы, продолжения исполнения отлаживаемой программы, трассировки участка программы. Нужно отметить, что способ реализации команд контроля программы связан с записью кодов программного прерывания в текст отлаживаемой программы, так что для программ, находящихся в постоянной памяти, эти команды невыполнимы.

3. Команды визуализации. Группа команд визуализации обеспечивает визуализацию абсолютных адресов контрольных точек и участков памяти в виде десятичных целых или вещественных чисел разной длины. Существует также возможность управления устройством визуализации (системный дисплей или печатающее устройство).

Работа монитора использует возможности микропроцессора по обработке программных прерываний. Создание контрольных точек осуществляется путем записи кода программного прерывания в указанном пользователем месте отлаживаемой программы. Вычисленный монитором абсолютный адрес контрольной точки и содержимое соответствующей ячейки сохраняются в специальной таблице. Размер таблицы допускает одновременное наличие не более 30 контрольных точек в отлаживаемой программе. Удаление контрольной точки приводит к восстановлению содержимого соответствующей ячейки и сжатию таблицы контрольных точек. Пошаговое выполнение отлаживаемой программы связано с анализом текущей команды. В результате анализа создается таблица внутренних контрольных точек, которая содержит информацию об одной или двух контрольных точках в зависимости от характера команды. Сразу после выполнения инструкции внутренние контрольные точки удаляются и содержимое памяти восстанавливается.

5.2. Драйверы. Анализ разработок проблемно-ориентированных систем на базе ЦЛАНП 0270 показывает, что набор программных модулей связи с объектом сравнительно постоянен. С целью сокращения потерь времени на разработку прикладных программных систем целесообразно создание универсального набора управляющих программ (драйверов), который можно использовать после настройки некоторых параметров, касающихся числа каналов, размера области обмена информацией и т. д.

Для микрокомпьютерной системы ЦЛАНП 0270 создан набор подпрограмм управления АЦП, ЦАП, входными и выходными регистрами, модулями счетчиков, модулями связи, модулем цветного дисплея и т. д. [10]. Структура этих драйверов соответствует специфике аппаратного обеспечения системы. Инициализирующая часть обеспечивает загрузку КАМАК-команд в определенную область памяти КАМАК-программ, модификацию адресов программ обработки прерываний, занесение необходимой управляющей информации. Программы обработки прерываний обеспечивают физический обмен информацией между КАМАК-модулями системы и оперативной памятью.

6. **Кросс-средства.** Ограниченные возможности 8-разрядных микропроцессоров и их операционных систем обусловливают ориентацию на создание кросс-программных продуктов на развитых вычислительных комплексах. Задача кросс-систем состоит в создании файла для объектного компьютера (объектного файла) на инструментальном компьютере.

6.1. Кросс-трансляторы. При создании кросс-средств системы ЦЛАНП 0270 был предложен подход [11], характеризующийся использованием макрогенератора инструментального компьютера, при помощи которого генерируется соответствующий объектный файл. Воз-

можность вложения макроопределений на нескольких уровнях упрощает работу с однотипными функциями языка и лежит в основе структурного построения кросс-обеспечения на нескольких уровнях. На самом низком уровне находится кросс-ассемблер для объектного компьютера, на следующем уровне на основе уже созданного аппарата можно создавать кросс-компиляторы языков промежуточного уровня, а на высшем уровне — кросс-компиляторы языков высокого уровня. Кросс-транслятор, созданный на базе рассматриваемого подхода, включает три основные программные фазы: фазу перекодировки (переводит исходную программу в формат, определенный синтаксисом макрогенератора); фазу генерации (генерирует необходимый объектный файл и печатает диагностическую информацию); фазу вывода (переносит сгенерированный объектный файл на внешний носитель).

Созданное таким способом кросс-обеспечение дает возможность оптимально использовать операционную систему инструментального компьютера. На базе этой структуры в ЦЛАНП БАН реализованы кросс-ассемблер для микропроцессора М6800, кросс-транслятор с языка промежуточного уровня ИМЛ-270 и кросс-транслятор с языка высокого уровня МПЛ-270.

6.2. Антиассемблер. В состав кросс-обеспечения системы ЦЛАНП 0270 включена программа, предназначенная для восстановления исходного текста программ по их абсолютному коду [12]. При корректном задании участков антиассемблирования можно получить необходимую информацию о структуре обрабатываемой программы, областях данных, относительных передачах управления, вызовах подпрограмм и т. д. Антиассемблер облегчает анализ уже созданного программного обеспечения, его модификацию и адаптацию.

ЛИТЕРАТУРА

1. EUR 4100. CAMAC. A modular instrumentation system for data handling.— ESONE Committee, 1972.
2. **Тренев А.** Система КАМАК.— Автоматика и вычисл. техника, 1982, № 6.
3. **Димитров В., Няголов А.** Модулна микрокомпютърна система за автоматизация ЦЛАНП 0270.— В кн.: Автоматизация и научное приборостроение: 2-й Междунар. симп. Варна, 1983.
4. **Angelov A., Antonov L., Ianev K., Trenev A.** Arhitecture of a multiprocessor CAMAC system with shared functions of the processors.— In: Real-Time Data Handling and Processor Control. Amsterdam, 1980.
5. **Ефремова Р., Попов О., Влахова К.** Расширение алгоритмического языка МПЛ для управления системами КАМАК.— В кн.: Автоматизация и научное приборостроение: 2-й Междунар. симп. Варна, 1983.
6. **Влахова К. П., Ефремова Р. Д.** Применение языка ФОРТРАН для управления микрокомпьютерными системами в стандарте КАМАК.— В кн.: 1-я Междунар. школа по АНИ. Тез. докл.— М., 1982.
7. **Аврамов И., Няголов А., Сараиванова Е., Янев К.** Интерпретатор арифметических выражений в системе с арифметическим процессором.— В кн.: Автоматизация и научное приборостроение: 1-й Междунар. симп. Варна, 1981.
8. **Вербова Р., Няголов А.** Алгоритмический язык ФОРТРАН с библиотекой быстродействующей арифметики для МСА ЦЛАНП 0270.— В кн.: Автоматизация и научное приборостроение. 3-й Междунар. симп. Варна, 1985.
9. **Няголов А., Ефремова Р.** Диалоговый резидентный монитор для отладки программного обеспечения DEBUG.— В кн.: Автоматизация научных исследований: Междунар. конф. Пловдив, 1984.
10. **Андреев А.** Драйвер линк-модуля в системе КАМАК.— Там же.
11. **Влахова К., Ефремова Р.** Кросс-программное обеспечение на нескольких уровнях для управления микрокомпьютерными системами КАМАК на базе макродефиниционного подхода.— В кн.: 1-я Междунар. школа по АНИ. СССР. Пущино, 1982.
12. **Томов Б., Янев К.** Семантично реасемблиране на програмни текстове: 10-та пролетна конференция на СМБ.— НРБ, Слънчев бряг, 1981.

Поступила в редакцию 17 января 1986 г.