УЧЕНЫЕ ЗАПИСКИ

КИБЕРНЕТИЗАЦИЯ НАУЧНОГО ЭКСПЕРИМЕНТА

6

Министерство высшего и среднего специального образования
Латвийской ССР
Латвийский ордена Трудового Красного Знамени
государственный университет имени Петра Стучки
Проблемная лаборатория физики полупроводников

Ученые записки
Латвийского государственного университета
имени Петра Стучки
том 225

КИБЕРНЕТИЗАЦИЯ НАУЧНОГО ЭКСПЕРИМЕНТА

Выпуск 6

Латвийский государственный университет
Рига 1975

КИБЕРНЕТИЗАЦИЯ НАУЧНОГО ЭКСПЕРИМЕНТА

Выпуск 6. 1974.

В четырех частях сборника рассматриваются вопросы применения вычислительной техники для управления научным экспериментом. Первая часть посвящена общим вопросам автоматизации программируемого эксперимента, в особенности принципам построения программного обеспечения эксперимента. Вторая часть содержит статьи по конкретным разработкам программного обеспечения эксперимента для двухпроцессорных систем (базисная машина "Днепр-2I"), а также обеспечение на основе универсальных программных модулей однопроцессорных систем. Третья часть включает статьи по режиму диалога при организации эксперимента и обработки его результатов. В последней части рассматривается ряд технических вопросов: описывается система для теплофизических экспериментов, обсуждается конфигурация ЭВУМ в системе управления экспериментом по сбору диагностической информации.

Сборник предназначен для научно-технических работников и студентов, интересующихся практическими вопросами применения вычислительной техники для целей непосредственного управления научными и техническими экспериментами средней сложности.

К 30500-040у 296-74
М 812(II)-75

УДК 681.3.01:007.5

Ю.Я.Кузьмин

О НЕКОТОРЫХ ПРИНЦИПАХ УНИВЕРСАЛЬНОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ЭКСПЕРИМЕНТА

Предлагаются четыре принципа программирования экспериментов (модульность, универсальность, интерфейс, диалог), позволяющие создать универсальное программное обеспечение, пригодное для определенных классов экспериментов. Отмечаются преимущества и область применимости универсального программного обеспечения.

Универсальное программное обеспечение эксперимента (УПОЭ) – это одна из альтернатив решения сложной задачи создания фонда программ, предназначенных для автоматизации экспериментальных исследований. Основная идея УПОЭ состоит в создании общих алгоритмов управления экспериментом и в разработке на этой основе некоторого числа программных модулей, которые учитывают основные функции, присущие каждому эксперименту (сбор, контроль, управление, диалог и др.). Таким образом, системы, содержащие УПОЭ, не ориентированы на конкретный эксперимент, а стадия программирования таких систем вырождается в настройку УПОЭ на конкретный состав технических средств и конкретную методику экспериментирования.

Работы в этом направлении начаты рядом зарубежных фирм. Среди универсальных систем подобного рода наиболее известна система программных модулей МАДАМ [1], разработанная и использованная для многих десятков задач фирмой Siemens (ФРГ).

В настоящей статье рассмотрены четыре основных принципа, которые, по мнению автора, могут лечь в основу разработки УПОЭ.

Причины появления интереса к УПОЭ в Проблемной лаборатории физики полупроводников ЛГУ им.П.Стучки были сугубо практические и связаны с разработкой одной из довольно крупных для общефизического эксперимента системы тепловых испытаний НИС "GUNDEGA-2". * Разработка системы велась в сжатые сроки и пре-

* См.настоящий сборник, с. III.

следовала решение таких задач как создание технических средств и одновременное создание достаточно гибкой программы управления этими средствами.

Уже на начальной стадии работы выяснилась непригодность традиционного способа разработки программ эксперимента, требующего полную определенность в том, какой метод будет положен в основу эксперимента и какие технические средства будут в нем заложены. Поэтому была поставлена цель разработать программное обеспечение эксперимента, минимально зависимое от количества аппаратуры и конкретной методики эксперимента, в дальнейшем получившее название УПОЭ-"GUNDEGA-2". Естественно, определенная информация о типе аппаратуры и классе экспериментов имелась; кроме того, по мере возможности оказывалось влияние на оформление каналов аппаратуры. При разработке УПОЭ учитывались и некоторые требования со стороны других, имеющихся или проектируемых систем.

В последнее время появился термин "системные программы"; это новый тип программ, в основу которых положены не прикладные задачи, а универсальные функции системы, решающей прикладные задачи. Бо́льшая часть системных программ связана с так называемым сервисом, т.е. с теми функциями системы, которые прямо или косвенно позволяют существенно сократить время программирования и управления процессом прохождения программ. При разработке УПОЭ сделана попытка применить идеологию системных программ, базируясь не на знании конкретного эксперимента, а на знании типа аппаратуры, с которой придется работать, и знании общего характера задач эксперимента. Таким образом мы шли не от эксперимента, а от системы.

Для проведения необходимых обобщений нам пришлось отвлечься от семантики входных и выходных каналов, это позволило перейти от понятия состояния канала, которым все привыкли пользоваться, к понятию состояние системы и сделать его основным объектом программирования. Оказалось, что принятие этого принципа открывает определенные возможности построения универсальных для некоторого класса задач

программ. Под состоянием системы в нашем случае понимается значение входных и выходных каналов системы, некоторых вычислительных параметров, а также состояние самих универсальных программ – модулей. Программирование состояний системы обладает рядом достоинств.

I. Повышается надежность системы, ибо под наблюдением оказываются не отдельные каналы, а вся система. Имея более полную информацию, система принимает, естественно, более верное решение в соответствующих ситуациях.

2. Естественным образом вводится преемственность УПОЭ малых систем и больших систем. Это достигается путем простого расширения полей описания системы и полей параметров. Для очень больших систем возможна установка процессоров по числу модулей УПОЭ, например, процессора для сбора информации, процессора для контроля, процессора для управления каналами и т.д.

3. Возникает возможность организовать многоконтурное управление, так как система работает не с каждым каналом последовательно, а со всей группой каналов одновременно. Это особенно важно для многоканальных систем, когда отработка каналов идет продолжительное время и к моменту отработки последнего канала первые могут сильно разрегулироваться.

4. Облегчается режим диалогового управления, потому что взаимодействие осуществляется не с переменным количеством программ, а стандартизованно и с фиксированными программными модулями. Появляется возможность одним приказом приводить в определенное состояние не только отдельные каналы, но и всю систему, что с успехом было использовано в УПОЭ-"GUNDEGA-2".

5. Упрощается задача реакций системы в аварийных ситуациях, поскольку возможно блокирование запрещенных состояний системы и перевод ее в специальные состояния простыми программными средствами.

6. Появляется возможность отладки системы и ее проверки без наличия конкретного алгоритма эксперимента на основе использования некоторых основных состояний.

Чтобы пояснить принцип программирования состояний, а также другие принципы, заложенные нами в основу УПОЭ, отме-

тим одну характерную особенность стендовых испытаний, которая присуща, на наш взгляд, также и экспериментам в области общей физики. Это - циклический характер эксперимента. Действительно, для получения каждого кванта информации (например одной точки спектра излучения образца) нужно последовательно отработать определенные функции системы: сбор информации, ее анализ, вычисление режимов эксперимента, контроль системы, выдача информации, связь с оператором и т.п. Обычно эти функции отрабатываются в соответствии с методикой конкретного эксперимента, при этом каждая из них может отрабатываться в разных местах программы эксперимента, согласно принципу отработки состояния каналов. Если в качестве исходного принять принцип отработки состояния системы, то необходимым следствием будет централизация каждой из вышеупомянутых функций в отдельном программном модуле, который должен быть универсальным, так как им охватываются все формы отрабатываемой функции для данного класса экспериментов. Универсальный программный модуль в свою очередь требует стандартизации форматов информации, с которой он работает. Стандарт должен быть общемодульным, поскольку с одной и той же информацией может работать несколько различных программных модулей. В нашем случае, например, входная информация имела следующий формат: (N, a), где N - идентификатор параметра в системе, a - значение параметра, выраженное в физических единицах (например, температура в градусах Цельсия, давление - в атмосферах и т.д.). За меру дискретных величин принят процент пребывания устройства в состоянии "I" ("включено") за некоторый промежуток времени (I сек).

Помимо очевидных достоинств стандартизации форматов можно отметить еще и следующие:

I) уменьшение числа согласующих программ для стыковки программных модулей;

2) упрощение программных модулей вывода информации, поскольку они должны воспринимать ограниченное количество форматов;

3) доступность всей информации о параметрах системы в любой комбинации, ибо вывод не привязан к определенным каналам.

Следует отметить, что разработка УПОЭ существенным образом зависит от архитектуры периферии УВМ. Особенно это относится к каналам приема измерительной и контрольной информации и выдачи управляющей информации. В этом смысле для УПОЭ наиболее подходят системы с хорошо разработанной логикой развития периферии, например, АСВТ-М и САМАС.

Следующим важным принципом, заложенным в УПОЭ, является независимость универсальных программных модулей от набора технических средств, что достигается за счет введения интерфейсных таблиц. В них, например, указывается связь физического адреса устройства с его идентификатором в системе. При использовании интерфейсных таблиц, всякое изменение технических средств влечет за собой вычеркивание или добавление соответствующих строк в таблицах, но не требует перепрограммирования системы.

В интерфейс системы "GUNDECA-2" входит: указанная выше таблица описания каналов, коэффициенты линеаризации и приведения к физическим единицам измерений, функции описания логики работы устройств, уставки, режимы работы устройств, диагностические таблицы для определения аварий в узлах, таблица для определения реакций системы и другие таблицы.

Задача экспериментатора состоит в заполнении этих таблиц каждый раз при изменениях в системе, либо в методике эксперимента; при этом программирование как таковое отсутствует.

Следующий принцип, положенный в основу УПОЭ, состоит в широком использовании режима диалога. Этот режим существенно необходим для поисковых экспериментов, когда отдельные манипуляции с системой нужно производить в ходе эксперимента, например, задание нужных режимов, выбор регистрируемых параметров, управление отдельными каналами и т.п. Кроме того, режим диалога позволяет существенно упростить все УПОЭ за счет использования трудно программируемых возможностей человека. Например, оставив за экспериментатором задачу принятия решения в неопределенных ситуациях, мы освобождаем память УВМ от соответствующих громоздких и, как правило, малоэффективных программ. То же касается задачи распознавания ситуаций, например, определения степени аварийности системы, информативности эксперимента при заданных условиях и т.п.

В статье Л.М.Кузьминой и Я.П.Цирулиса [*] обсуждаются некоторые аспекты построения достаточно универсальных средств диалога.

Следует отметить, что принятие одного принципа существенно облегчает выполнение других; так, проведение диалога с универсальным модулем проще, чем предусматривать места для диалога в специализированных программах.

Для специалистов, никогда не пытавшихся программировать состояния системы, представляется сомнительной возможность решения ряда практически важных процессов.

I. Приоритетность процессов. Имеется в виду возможность оперативного и своевременного перехода от одних функций системы к другим, в зависимости от реальной ситуации в эксперименте. Обычно в качестве альтернативы предлагается жесткая схема приоритетов процессов. Однако УПОЭ не имеет ограничений или запретов на введение динамического приоритета системы модулей. Некоторые варианты рассмотрены Ю.Я.Кузьминым, И.А.Кельман и С.В.Гвоздевым. [**] Появление временных потерь при смене модулей в оперативной памяти, если она недостаточно велика для хранения всего УПОЭ, характерно не только для альтернативы УПОЭ. Эта проблема встает всегда, как только структура комплекса программ выходит за рамки последовательной цепочки модулей.

2. Избыточность входной информации. Примитивный вариант УПОЭ предполагает наличие модуля, который циклически выполняет функцию отображения множества сигналов на входах всех датчиков системы на некоторую область памяти УВМ. Чтобы не потерять информацию, это отображение в простейшем случае должно быть согласовано с максимальной частотой изменения входных сигналов. Такое решение позволяет существенно упростить структуру модуля сбора информации. Опять же в УПОЭ нет ни ограничений, ни запретов на введение процедуры селекции каналов во времени согласно некоторым условиям, зависящим от текущего состояния системы. Введение такой селекции предполагает наличие специального модуля селекции каналов [***] и механизма учета результатов селекции в модуле информации. Все сказанное относится и к выходным каналам системы.

---

[*] См. настоящий сборник, с.99.

[**] См. настоящий сборник, с. 72.

[***] См. там же.

Можно, однако, сделать следующее общее замечание по поводу избыточности информации. Обычно имеется в виду избыточность в смысле превышения объема необходимой информации для расчетов текущих параметров объекта исследования. Последовательные алгоритмы традиционно строятся, исходя из принципа программирования только тех операций, которые нужны в текущий момент для получения информации об объекте исследования по той или иной методике эксперимента. Программист-физик обычно неявно мыслит категориями эксперимента, управляемого вручную, перелагает на программу те операции, которые он выполнял бы вручную, не будь УВМ. Естественно, что он представляет эксперимент как последовательный процесс работы с отдельными каналами управления и измерения, при этом контроль системы также осуществляется последовательно и зависит от того, какой канал, согласно применяемой методике, должен использоваться в каждый конкретный момент. Таким образом остальные каналы выпадают из поля зрения программиста. Неисправность каждого из них определяется только в момент работы с этим каналом. Однако возможны ситуации, когда сбой неконтролируемого в данный момент канала может оказать влияние на достоверность информации другого, нормально работающего, с которым действует программа в данный момент. Если каналы, например, подсоединены через один коммутатор к аналого-цифровому преобразователю, то сбой в параллельном канале может привести к искажению информации по другим каналам. Восприятие всей входной информации не страдает подобным недостатком, поскольку позволяет вести общий контроль системы наряду с отработкой операций, непосредственно связанных с методикой эксперимента.

3. Распараллеливание процесса программирования. Имеется в виду распределение задач программирования между различными специалистами – физиками, инженерами, математиками. В данном случае это решается следующим образом. Разработка программных модулей проводится математиками, описание технической части системы (таблицы сопряжения каналов, таблицы функционирования каналов, логика контроля системы и т.п.) – инженерами системы, а от физиков требуется описать граф эксперимента, дополнить таблицы описания каналов и контроля системы логическими условиями, связанными с методикой экспери-

мента, а также перечень реакций системы специфичными для методики эксперимента реакциями.

Таким образом, видим, что предлагаемые принципы не противоречат возможности создания гибкого программного обеспечения эксперимента.

Далее рассмотрим область применения УПОЭ. Следует различать принципиальную неприменимость УПОЭ и ограниченность конкретного варианта из-за его неразвитости.

УПОЭ состоит из универсальных модулей. За универсальность, естественно, приходится платить лишними командами и потерей времени процессора, поэтому, например, УПОЭ-"CUMBECA-2" непригодно для экспериментов, в которых основной цикл меньше времени обегания модулей УПОЭ. Однако, как отмечалось выше, уже сейчас имеются идеи минимизации временных потерь за счет введения модуля селекции каналов и даже селекции модулей. Другой путь состоит в увязке УПОЭ с системой прерывания, которая хорошо развита у современных машин, например, у мини-ЭВМ М-6000.

УПОЭ нецелесообразно для простых экспериментов непоискового характера, каналы которых должны работать последовательно и изолированно.

УПОЭ непригодно для систем, у которых каждый программный модуль не помещается целиком в оперативное запоминающее устройство УВМ, например, в случае, когда оперативное запоминающее устройство очень мало.

В настоящее время не исследован вопрос о применимости УПОЭ для систем, описываемых очень сложной временной логикой, а также для систем, построенных на основе локальных автоматов.

Не изучался вопрос особенностей применения УПОЭ в системах с разделением времени.

Последующее развитие УПОЭ намечается путем его перенесения на современную ЭВМ М-6000. Планируется разработать средства генерации интерфейса как перед, так и во время эксперимента. Это необходимо при подключении, исключении или замене приборов в экстренном порядке, то же самое касается режимов или отдельных параметров. Предусматривается также дальнейшее развитие режима диалога. В частности, совместно с Я.П. Цирулисом и Л.М.Кузьминой разрабатывается универсальный модуль, по-

зволяющий в ходе эксперимента вводить и отрабатывать составные директивы. * Упрощенный вариант такого модуля был описан в статье [2], а также в статье Л.М.Кузьминой и А.Н.Назаровой **.

Помимо вышеизложенного намечается развить средства минимизации временных потерь в УПОЭ, а также средства диалога.

Основные идеи УПОЭ в настоящее время находятся в стадии становления и развития, поэтому многие положения требуют детальных исследований и усовершенствования.

ЛИТЕРАТУРА

1. Hirschberg G. Zielsetzung und Anwendung des Programmsystems MADAM. - "Siemens-Zeitschrift", 1971, 45, H.10, S.793-795.

2. Кузьмин Ю.Я. Реализация фраз в режиме диалога. - Уч. зап. ЛГУ им.П.Стучки, т.196. Кибернетизация научного эксперимента, вып.5. Рига. 1973, с.50-54.

---

* См.настоящий сборник, с. 90.

** См.настоящий сборник, с. 104.

УДК 681.3.01:007.5:53

А.А.Бернуп-Бернхоф

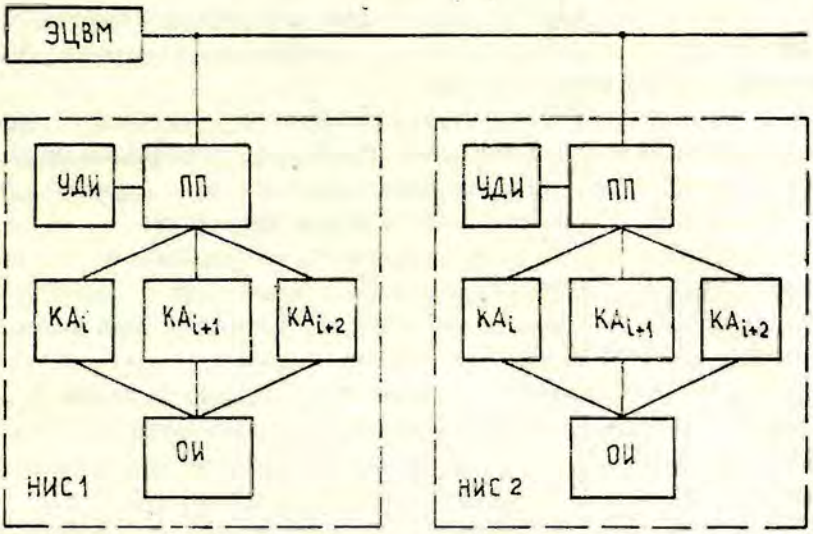
ПРОГРАММИРОВАНИЕ ЭКСПЕРИМЕНТА В ДВУХПРОЦЕССОРНОЙ НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ СИСТЕМЕ

Рассматриваются особенности разработки программного обеспечения двухпроцессорной научно-исследовательской системы. Вместо альтернативы групповых программ предлагается иерархическая мультипрограммная декомпозиция математического обеспечения, включающая элементы для раздельной координации каждого физического канала научно-исследовательской системы и всей системы в целом. Анализируются последствия применения этого способа декомпозиции для алгоритмов некоторого класса физических методов исследований.

В области физики твердого тела имеется ряд методов организации экспериментов [1], автоматизация которых обещает существенное повышение эффективности научных исследований. Для достижения этой цели может быть создан комплекс (рис.1) на базе центральной ЭЦВМ, работающей в режиме разделения ресурсов между несколькими научно-исследовательскими системами (НИС), и периферийных процессоров (ПП), программы которых хранятся или в их местной памяти [2], или в памяти ЭЦВМ [3].

Конструкция этих ПП гораздо проще современных минимашин и в то же время они способны управлять измерительно-воздействующей аппаратурой (ИВА) эксперимента, вырабатывать сигналы прерывания ЭЦВМ, а также реагировать на прерывающие сигналы центральной машины и осуществлять с нею обмен информацией. Предполагается, что в каждой НИС находится отдельный ПП, а для ее обслуживания в центральной ЭЦВМ программа СУПЕРВИЗОР выделяет ряд приоритетных уровней, временные характеристики которых фиксированы и независимы от остальных пользователей.

Чтобы отличать научно-исследовательскую систему, которую одновременно обслуживают два процессора, введем термин - двухпроцессорная НИС. Указанные ограничения режима работы центральной ЭЦВМ позволяют в таком случае предполагать, что с точки зрения программирования к этому классу может быть отнесена также каждая отдельная НИС комплекса (см.рис.1).



Р и с. I. Блок-схема аппаратуры комплекса НИС.
ОИ – объект исследования, ПП – периферийный процессор, $КА_i$ – i-й физический канал, УДИ – устройство диалога с системой.

В соответствии с глобальной целью проектирования НИС [4] основными критериями качества программного обеспечения эксперимента могут рассматриваться доступность, производительность и стоимость разработки НИС. Под доступностью в этой статье понимаются удобства разработки, модификации и применения программ на уровне физика-исследователя (пользователя НИС). Производительность определяется быстродействием системы. При этом важно отметить, что структура внутреннего программного обеспечения НИС должна создать условия для оперативного изменения алгоритмов управления экспериментом и выполнения рабочих программ с учетом временных ограничений, которые определяются особенностями реализуемого процесса. Опыт разработки показывает, что достижение указанных качественных показателей программного обеспечения наиболее

перспективно путем разделения процесса программирования между инженерами и пользователем НИС.

Рассмотрим задачу декомпозиции алгоритмов эксперимента на модули , а также требования к организации их связей. Определим разработчиков программ.

Известен метод групповых программ [5], который применен для декомпозиции программ, работающих в реальном масштабе времени. Отказ от применения этого метода в случае двухпроцессорной НИС связан со следующими факторами:

- метод [5] почти не рассматривает возможности приоритетной организации программ, в результате чего увеличение числа агрегатов ИВА или повышение их быстродействия может вызвать перегрузку ЭЦВМ по быстродействию;
- ввод всех данных в машину предусмотрен с частотой, которую определяет наиболее интенсивно опрашиваемый датчик НИС. Такая избыточность данных может привести к перегрузке каналов связи ЭЦВМ с локальными процессорами;
- отсутствие формального описания языка программирования, соответствующего предложенной интерпретирующей системе.

Следует однако отметить, что в настоящее время начата разработка перечисленных проблем *.

Рассмотрим альтернативу декомпозиции алгоритмов эксперимента, предполагающую иерархическую мультипрограммную организацию программ, в которой для работ определенного характера выделяются фиксированные приоритеты. Перед тем, как определить функции основных программных модулей разрабатываемой системы, перечислим некоторые факторы, влияющие на ее качественные характеристики. С точки зрения доступности НИС желательно иметь возможность обращаться с уровня программ пользователя к таким процедурам как:

- управление конкретным физическим параметром объекта исследований;
- измерение некоторого физического параметра объекта;
- обработка данных по определенному алгоритму;

---

* См. настоящий сборник, с. 73.

– обмен информацией между НИС и ее пользователем.

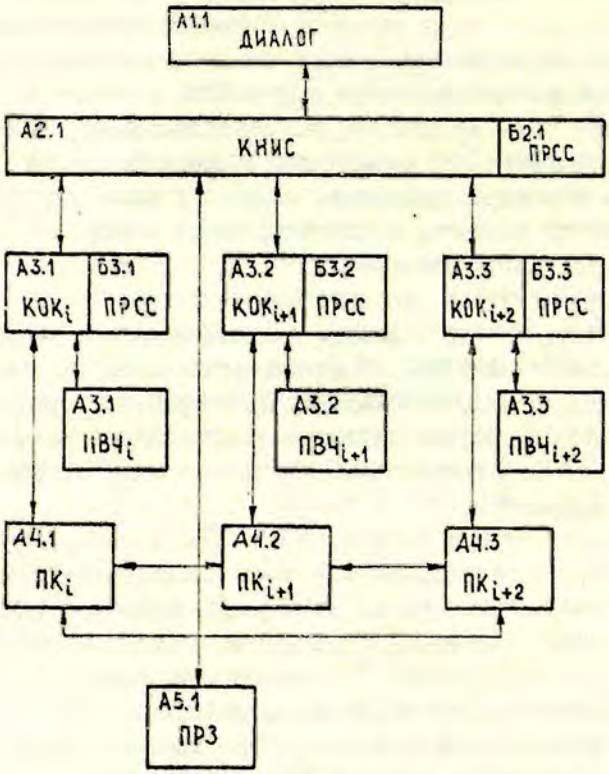
В случаях, когда необходимо модифицировать алгоритм эксперимента, наиболее трудоемкой бывает переделка программ, непосредственно взаимодействующих с ИВА. Фактически этот процесс может быть рассмотрен как микропрограммирование. Стоимость разработки НИС может быть существенно понижена, если агрегаты ИВА можно разделить на подсистемы, для которых все существенные варианты применения описываются небольшим числом стандартных подпрограмм.

Вышеизложенные соображения учитываются при декомпозиции программ ИВА по признаку принадлежности к определенному физическому каналу НИС. Под физическим каналом будем понимать комплект ИВА, необходимый для выработки конкретного физического воздействия на объект исследования или для измерения информации, необходимой при определении одного его физического параметра.

С точки зрения быстродействия НИС желательно, чтобы время реализации эксперимента в основном определялось быстродействием каналов НИС, но не временными характеристиками ЭЦВМ. Это означает, что выбор процедур обработки данных и процедур обмена информацией между НИС и ее пользователем следует выполнять с учетом особенностей обслуживания каналов.

На примере конкретного проекта (рис.2) рассмотрим организацию программных модулей НИС, основанную на вышеизложенных соображениях для класса задач [I]. Для таких несрочных процедур как трансляция, обращение к библиотеке во внешней памяти или обработка результатов измерений и представление их пользователю введен уровень программ рядовых задач (ПРЗ). Он снабжен механизмом динамической памяти и является единственным элементом организации программ, которому доступны все стандартные внешние устройства ЭЦВМ.

Программа обслуживания ИВА i-го канала ($ПК_i$) осуществляется в ПП и служит для выдачи в реальном времени управляющих сигналов на соответствующие агрегаты, а также для приема информации с их выходов. Процедура управления физическим каналом часто связана со вспомогательными расчетами, которые невозможны в ПП. Поэтому в ЭЦВМ предусмотрен приоритет для



Р и с. 2. Иерархическая организация программных модулей.

Индексом А обозначен основной процесс уровня, индексом Б – процесс реакции на случайные ситуации. Первая цифра за индексом – номер приоритетного уровня, вторая цифра – номер параллельного процесса на соответствующем уровне. Уровень А4(I–3) обслуживается при помощи ПП, остальные уровни – при помощи ЭЦВМ.

программ вспомогательных вычислений (ПВЧ$_i$) и программы координации канала (КОК$_i$). Последняя выполняет функции управления и синхронизации программы обслуживания ИВА, работающей в ПП, и процесса вспомогательных вычислений, осуществляя тем самым процедуру, которая имеет физический смысл для пользователя.

Алгоритмы класса [I] предусматривают параллельную работу физических каналов. Поэтому СУПЕРВИЗОР НИС для каждого канала в ЭЦВМ и в ПП выделяет постоянный квант времени и обеспечивает их обслуживание по кругу.

Над уровнем каналов расположен приоритет программы координации научно-исследовательской системы (КНИС). На этом уровне запланированы возбуждение координаторов каналов и процедур уровня рядовых задач, запрос начальных и текущих значений параметров процесса эксперимента от пользователя, а также контроль исправной работы системы. В программе координации НИС также отмечены состояния, позволяющие по указаниям пользователя изменить алгоритм эксперимента и состояния, в которые система должна возвращаться в случае сбоев.

На уровне рядовых задач применена организация так называемой динамической памяти, которая освобождает пользователя от необходимости планировать размещение соответствующих программ и данных в памяти ЭЦВМ. Однако эта задача должна быть решена в координаторе НИС относительно собственных полей, а также программ каналов. Это связано с особенностями алгоритмов экспериментов, которые только в определенных состояниях допускают задержки, связанные с обращением к внешней памяти ЭЦВМ.

Первый приоритет в системе предусмотрен для координирующей программы ДИАЛОГ, при помощи которой пользователь получает текущие результаты измерений, меняет алгоритм эксперимента, если это необходимо, а также управляет процессом ликвидации аварийных ситуаций НИС.

Каждая программа-координатор в пределах своего приоритета имеет возможность заказать ряд подпрограмм реакции на случайные ситуации (ПРСС). Их активизацию при заданных условиях осуществляет программа ИНТЕРПРЕТАТОР, которая связана с СУПЕРВИЗОРОМ НИС.

Первые результаты внедрения такой организации программ на ЭЦВМ "Днепр-21" и ПП типа [2] показывают, что декомпозиция алгоритмов эксперимента на модули специализированных процедур, подчиненных координаторам физических каналов и координатору НИС, создает благоприятные условия для целенаправленного разделения труда программирования между пользователем и инженерами НИС. Если программы вспомогательных вычислений и прог-

раммы рядовых задач составляются программистами-математиками, а программы координации каналов и обслуживания ИВА создаются системными инженерами, пользователь должен разрабатывать только координатор НИС и знать формат обращения к координируемым подпрограммам.

Остановимся на особенностях языка программирования и рассмотрим соответствующие средства трансляции программ вышеизложенной организации. Известно, что с точки зрения критерия быстродействия мультипрограммная работа НИС может дать весьма хорошие результаты, если транслятор, переводящий программы ЧИС с входного языка на машинные коды работает достаточно эффективно. Для программирования систем, работающих в реальном масштабе времени, поэтому часто применяются макроассемблеры [6]. Опыт внедрения показывает, что практически не возникают трудности также при разработке транслятора, объединяющего макроассемблеры двух неоднородных процессоров (центральной ЭЦВМ и ПП).*

Макроассемблер может служить для разработки быстродействующих программ последовательного типа. Чтобы обеспечить мультипрограммную реализацию нескольких процессов, язык программирования модулей любого уровня должен быть расширен средствами распараллеливания и координации программ. В качестве такого средства может быть применена система директив.**

Разработка средств компиляции программ двухпроцессорной НИС, содержащих директивы координации параллельных процессов, существенно увеличивает стоимость разработки НИС. Поэтому целесообразно применить смешанную систему трансляции, по которой в процессе компиляции в машинные коды переводятся только те строки программы НИС, которые записаны в макроассемблере. Строки, соответствующие директивам распараллеливания и координации программ, на этом этапе подвергаются только синтаксическому контролю и переводятся в некоторый промежуточный язык. Дальнейшая обработка директив реализуется в специальном трансляторе ИНТЕРПРЕТАТОРЕ, который связан с СУПЕРВИЗОРОМ НИС.

Применение ИНТЕРПРЕТАТОРА директив существенно упрощает не только мультипрограммирование ЭЦВМ, но также организацию ее

---

* См. настоящий сборник, с. 40.

** См. настоящий сборник, с. 21.

взаимодействия с ПП. В конкретной системе этот процесс связан с передачей программ, массивов данных, координирующих директив и ответных сигналов, вырабатываемых в координируемых элементах. В каждом координаторе канала соответствующая программа обслуживания ИВА может быть рассмотрена как линейный ряд подпрограмм, в которые перед выдачей на ПП в заранее фиксированных местах вносятся значения рабочих аргументов.

Однако следует отметить, что интерпретирующие системы являются весьма медленнодействующими. Понижение быстродействия НИС, вызванное ИНТЕРПРЕТАТОРОМ, может быть сведено к минимуму, если наиболее часто употребляемые программные модули содержат небольшое число интерпретируемых директив. К таким в основном следует отнести программы, разрабатываемые системными инженерами. Опыт программирования НИС показывает, что для класса алгоритмов экспериментов [I] иерархическая организация программ этому требованию удовлетворяет.

Первые результаты декомпозиции алгоритмов эксперимента по принятой нами альтернативе позволяют сделать следующие выводы и замечания:

– можно обеспечить координацию двухпроцессорной НИС и гибкую модификацию алгоритмов эксперимента при помощи относительно несложной программы пользователя. Система координирующих директив * может быть оценена как основная составляющая проблемно-ориентированного языка, на котором пользователь описывает процессы эксперимента;

– иерархический способ декомпозиции алгоритмов эксперимента позволяет унифицировать требования процессов НИС, связанные с режимом их осуществления в реальном масштабе времени. Для модуля любого приоритета они могут быть заданы при помощи временных параметров $\{\tau, t, v\}$, где τ – максимальное время возбуждения программы после ее запроса; t – максимально допустимое время прерывания данной программы между любыми двумя ее операциями; v – минимальное значение средней скорости выполнения операций возбужденной программы, измеряемое в заданном интервале Δt в произволь-

---

* См.настоящий сборник, с. 21.

ном месте на оси времени;

- класс задач, решаемых при помощи выбранной организации программ в основном определяется функциональными возможностями и быстродействием ИНТЕРПРЕТАТОРА. В процессе разработки последнего желательно исследовать координируемость [7] НИС по критерию быстродействия путем применения имитационной модели ИНТЕРПРЕТАТОРА совместно с типовыми моделями того класса задач, для которого система проектируется;

- вышерассмотренная организация программ может быть применена как в двухпроцессорных НИС, в которых память ПШ и ЭЦВМ централизована [3], так и в системах с децентрализованной памятью [2]. Следует однако отметить, что первая альтернатива менее трудоемка при разработке ИНТЕРПРЕТАТОРА координирующих директив.

ЛИТЕРАТУРА

I. Тале И.А. Системный анализ и оценка целесообразности спектрально-кинетических люминесцентных методов исследования твердого тела. - Уч.зап.ЛГУ, т.160. Кибернетизация научного эксперимента, вып.3. Рига, 1972, с.11-25.

2. Кибернетизация научного эксперимента, вып.I. Глава 6. "Физпульт" - Машина, управляющая научным экспериментом, с.181-201. Рига, ЛГУ им.Петра Стучки, 1968.

3. Бернуп-Бернхоф А.А. Выбор варианта связи ЭЦВМ с системами экспериментирования. - "Автоматика и вычислительная техника", 1973, №2, с. 64-70.

4. Бернуп-Бернхоф А.А. Формализация цели проектирования научно-исследовательской системы.-Уч.зап.ЛГУ, т.160: Кибернетизация научного эксперимента, вып.3. Рига, 1972, с.37-55.

5. Кузьмин Ю.Я. Об одной альтернативе программирования эксперимента. - Уч.зап.ЛГУ, т.196. Кибернетизация научного эксперимента, вып.5. Рига, 1974., с.3-16.

6. Липаев В.В., Колин К.К., Серебровский Л.А. Математическое обеспечение управляющих ЦВМ. М., "Советское радио", 1972.

7. Месарович М., Мако Д., Тахара И. Теория иерархических многоуровневых систем. М.," Мир ",1973.

УДК 681.3.06:007.5:53

А.А.Бернуп-Бернхоф, Х.Р.Краузе-Крузе

СИСТЕМА ДИРЕКТИВ ДЛЯ РАСПАРАЛЛЕЛИВАНИЯ И КООРДИНАЦИИ ПРОЦЕССОВ НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ СИСТЕМЫ

Рассматривается способ повышения эффективности труда пользователей научно-исследовательских систем. Формулируется синтаксис и излагается семантика системы директив. Определяются основные принципы координации иерархической структуры программного обеспечения научно-исследовательской системы, которые могут быть реализованы средствами системы директив.

При разработке научно-исследовательских систем (НИС) важно и решать проблему повышения эффективности труда пользователей таких систем, в частности физиков-экспериментаторов. А это в большой мере зависит от объема труда, затрачиваемого пользователями НИС на разработку и модификацию программ конкретного эксперимента. Пути уменьшения этих затрат связаны с решением задач:

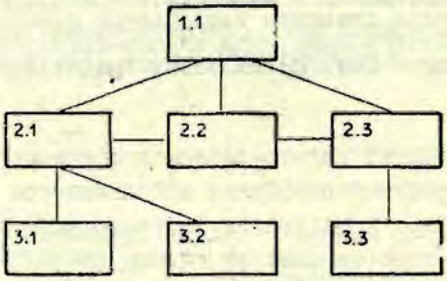
- выяснения способа декомпозиции программ эксперимента на модули;
- выбора принципов координации полученной системы программных модулей;
- определения границы, разделяющей синтаксис и семантику соответствующего проблемно-ориентированного языка программирования.

Учитывая трудности формулировки законченного варианта такого языка, авторы статьи выбрали достаточно абстрактный способ декомпозиции программ *, основные принципы которого частично рассмотрены также в работе [1], а теоретические основы - авторами работ [2 и 3]. Соответственно было предусмотрено разработать независимую от конкретной машины систему директив, позволяющую в процессе ее применения исследовать и

* См.настоящий сборник, с. 12.

определить основные принципы координации иерархической структуры программы. Учитывалась также необходимость на практике проверить те принципы координации работающих в реальном масштабе времени систем, которые авторами были выяснены в результате анализа алгоритмов экспериментов (см.[4]).

Сущность выбранного способа декомпозиции программ заключается в том, что программные модули эксперимента располагаются по приоритетным уровням некоторой иерархической структуры (рис.I).



Р и с. I. Иерархическая структура программы НИС.

Первая цифра элемента иерархии указывает номер приоритетного уровня. Вторая цифра – номер псевдопараллельного процесса внутри соответствующего уровня.

Внутри каждого уровня может быть предусмотрено несколько мест (элементы иерархической структуры программы НИС), обслуживание которых ЭЦВМ организует псевдопараллельно во времени. Такому элементу присваивается определенная область памяти и приоритет обслуживания соответствующего уровня. Число уровней иерархии и число элементов внутри уровней выбирается пользователем НИС. Взаимодействие псевдопараллельно работающих программ элементов обеспечивается при помощи ИНТЕРПРЕТАТОРА, который осуществляет также разделение машинного времени между ними.

Программа, выполняемая в некотором элементе, координирует процессы нижерасположенных элементов иерархии. Такое решение

позволяет осуществить принцип разделения труда программирования, согласно которому пользователь разрабатывает только программу верхнего уровня иерархии, а инженеры НИС – программы элементов нижних уровней.

Первые результаты применения рассмотренной ниже системы директив показывают, что она пригодна не только для программирования задачи координации НИС на уровне пользователя, но и для описания процессов взаимодействия программных модулей нижних уровней иерархии. Отметим, что система директив предусмотрена для работы с процессами и событиями в реальном масштабе времени и в настоящее время используется как расширение языка макроассемблера ЭВМ "Днепр-21" *.

Синтаксис системы директив

Для описания синтаксиса будем пользоваться особыми знаками макроассемблера ␣,/ ↓ , а также металингвистическими обозначениями < > { }n , | , :: = , | , смысл которых разъяснен в настоящем сборнике (см.с. 47).

<процесс> :: = <основной процесс> { <процесс реакция> }n | <основной процесс>

<основной процесс> :: = { <агрегат> }n

<процесс реакции> :: = { <агрегат> \ <особая директива> }n

<агрегат> :: = { <строка> }n <директива> | <директива>

<директива> :: = ↓<название директивы> ␣ <НПИ> ␣ { <аргумент> }n , | ↓<название директивы> ␣ { <имя события> ␣ { <аргумент> }n }n , | ↓<название директивы> ␣ { <аргумент> }n .

<особая директива> :: = <директива РЕСР> | <директива ПЛАН> | <директива УНПР> | <директива НАЧТ>

<НПИ> :: = <номер процесса иерархии> :: = <натуральное число> А <натуральное число> | <натуральное число> Б <натуральное число>

<имя события> :: <НПИ> ␣ <идентификатор случайного события> | <НПИ> ␣ <идентификатор нормального события>

<идентификатор случайного события> :: = / <идентификатор>

<идентификатор нормального события> :: = <идентификатор>

<аргумент> :: = <значение> | <адрес операнда>

* См.настоящий сборник, с. 40.

Примечания.

I. Форма записи аргументов определяется конкретным типом директивы.

2. Конструкции <строка> , <идентификатор> , <значение> , <адрес операнда> , а также нижерассматриваемый <сегмент> определены при описании языка макроассемблера (см.с. 40).

Семантика системы директив

В настоящей работе будем пользоваться определением семантики языка [5], согласно которому семантика может быть представлена путем описания машины (ИНТЕРПРЕТАТОРА); последняя, получив описание процесса на данном языке, реагирует действительным выполнением этого процесса. ИНТЕРПРЕТАТОР предусмотрен для запоминания иерархической структуры программы НИС и возбуждается только во время реализации (интерпретации) директив агрегатов .

<Процессу> соответствует программа, собираемая при помощи средств макроассемблера из нескольких сегментов; во время выполнения она целиком хранится в поле оперативной памяти некоторого элемента иерархии.

Элемент определяется при помощи конструкции <НПИ>.Первая цифра в ней указывает номер приоритетного уровня, считая сверху иерархии, а вторая цифра - номер псевдопараллельно осуществляемого процесса внутри уровня. Индекс А в <НПИ> служит для обозначения <основного процесса> , а индекс Б - для обозначения <процесса реакции>.

<Процесс> характеризуется двумя параметрами машинных ресурсов, которые пользователь указывает при помощи нижерассмотренной директивы РЕСРС.

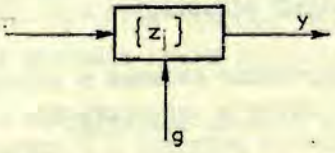
Предусмотрено, что в начале работы перед вызовом <процессов> в оперативную память ЭЦВМ элемент верхнего уровня иерархии (рис.I) при помощи директив РЕСРС заказывает ресурсы всех нижерасположенных элементов. Ввод с библиотеки и запуск этого основного элемента осуществляется при помощи последовательности директив РЕСРС, ВЫЗВ, СТАРТ, сообщаемых через терминал ИНТЕРПРЕТАТОРУ. В каждом элементе одновременно может быть обслужен не более чем один процесс.

⟨Основному процессу⟩ соответствует программный модуль, описывающий главную задачу конкретного элемента. Этот модуль может быть возбужден по директиве СТАРТ через свой единственный вход, который обязательно находится в самом начале ⟨процесса⟩.

Для реакции на случайные события в иерархии элементов основной процесс имеет возможность при помощи директив ПЛАН заказать ⟨процессы реакции⟩. В случае возникновения случайного события, объявленного в некотором элементе при помощи директивы СОБ, ИНТЕРПРЕТАТОР проверяет список запланированных процессов реакций во всей иерархии и, обнаружив совпадение имен и кодов событий, возбуждает соответствующий ⟨процесс реакции⟩. Последний осуществляется с приоритетом того ⟨основного процесса⟩, который его запланировал. Реализация ⟨основного процесса⟩ элемента задерживается до момента окончания ⟨процесса реакции⟩, в котором гасится также имя соответствующего случайного события. Если ⟨процесс реакции⟩ на возникшее случайное событие не запланирован, ИНТЕРПРЕТАТОР генерирует признак аварийного события, который вместе с ⟨НПИ⟩ объявленного события сообщается оператору НИС. Последний имеет возможность в режиме диалога указать необходимый процесс реакции, если соответствующая программа находится в памяти ЭВМ, и восстановить таким образом работу системы. Пользователю системы директив предоставлена возможность выбрать, какие из имеющихся событий в НИС именовать "случайными".

Конструкция ⟨агрегат⟩ соответствует специальному виду неавтономного кусочно-непрерывного агрегата, предложенного Н.П.Бусленко для описания специфических обрывающихся случайных процессов [3]. Пусть в ЭВМ работает программа ⟨агрегат⟩ (рис.2), на вход которой в момент t_0 поданы рабочие аргументы и сигнал пуска или только сигнал пуска. Агрегат некоторое время осуществляет последовательность внутренних состояний $\{z_j\}$, независимую от процессов в остальных элементах. Эта последовательность состояний может быть описана в любом независимом от ИНТЕРПРЕТАТОРА языке программирования (например в макроассемблере) и до момента t_0 транслирована в коды ЭВМ. По синтаксическому

определению <агрегат> заканчивается <директивой>, предназначенной для интерпретации.



Р и с. 2. Входы и выходы кусочно-непрерывного агрегата.

По сигналу прерывания на управляющем входе g , а также при обнаружении <директивы> в собственной программе агрегат переходит в одно из основных состояний, в котором обязательно вырабатывается информация на выходе y . Последняя может быть временно запомнена в ИНТЕРПРЕТАТОРЕ или же немедленно передана им в другие элементы иерархии. Длительность интервала прерывания программы агрегата контролируется ИНТЕРПРЕТАТОРОМ. Она не может быть больше значения t_{max}, определяемого пользователем в процессе описания иерархии директивой РЕСРС.

Конструкция <директива> служит для обеспечения взаимодействия между псевдопараллельно работающими процессами иерархии. Получив директиву , в зависимости от ее содержания ИНТЕРПРЕТАТОР немедленно возбуждает следующий <агрегат> <процесса> или задерживает это событие до реализации определенных условий в остальных элементах иерархии.

Для описания директив будем пользоваться обозначением α_i для i-го аргумента директивы. Могут быть применены следующие директивы:

I. РЕСРС $\sqcup$ <НПИ> $\sqcup \alpha_1 \sqcup \alpha_2$

α_1 – объем памяти в листах;

α_2 – максимально допустимое время прерывания <агрегатов> элемента в миллисекундах.

Директива служит для сообщения ИНТЕРПРЕТАТОРУ структуры осуществленной программы <НИС> (см. рис. I) и резервирует ресурс защищенной памяти элемента, обслуживаемого под соответствующим НПИ . Описание сруктуры должно начинаться с директивы, которая заказывает элемент организации с наимень-

шим номером уровня и заканчивается заказом элементов нижнего уровня. Сигналом окончания генерации организация для ИНТЕРПРЕТАТОРА является любая другая директива, следующая за директивой РЕСРС.

2. ↑ ВЫЗВ ␣ ⟨НПИ⟩α_1

α_1 – название процесса в библиотеке системы.

По директиве в заранее заказанное поле памяти элемента вводится из библиотеки системы соответствующая программа процесса. В ИНТЕРПРЕТАТОРЕ гасятся все имена событий, коды событий, идентификаторы массивов информации и запланированные при помощи директивы ПЛАН процессы реакции данного элемента.

3. ↑ ВЫД ␣ ⟨НПИ⟩ ␣ α_1 ␣ α_2 ␣ α_3,

α_1 – идентификатор массива данных;

α_2 – начальный математический адрес массива данных;

α_3 – объем памяти в байтах.

Директива сообщает ИНТЕРПРЕТАТОРУ о готовности массива данных к выдаче другому элементу иерархии согласно названию ⟨НПИ⟩. ИНТЕРПРЕТАТОР квалифицирует это как наступление события с идентификатором ЗВИ и кодом "I".

4. ↑ УНИН ␣ α_1

α_2 – идентификатор массива данных.

Директива запрещает доступ к данным, помеченным соответствующим идентификатором. Идентификатор события РВИ приобретает код "О".

5. ↑ ПРИЕМ ␣ ⟨НПИ⟩ ␣ α_1 ␣ α_2 ␣ α_3,

$\alpha_1, \alpha_2, \alpha_3$ имеют тот же смысл, что и в директиве ВЫД. При помощи директивы элемент просит выдать ему информацию с элемента, номер которого указан в ⟨НПИ⟩. Вырабатывается имя события с идентификатором ЗВИ и кодом "I". До получения данных процесс останавливается. Когда массив данных получен, ИНТЕРПРЕТАТОР вырабатывает и запоминает идентификатор события ЗВИ с кодом "О".

6. ↑ СТАРТ ␣ ⟨НПИ⟩ ␣ α_1 ␣ α_2 , ... , ␣ α_n ,

$\alpha_1, \ldots, \alpha_n$ – любые аргументы.

По директиве возбуждается программа соответствующего процесса и отмечается событие с идентификатором КОН и кодом "О".

7. ↑ ПЛАН ␣ ⟨имя события⟩ ␣ α_1 ␣ α_2 ␣ $_3$, ... , α_n ,

α_1 – код события ("О" или "I");

α_2 – математический адрес начала программы процесса реакции;

$\alpha_3, \ldots, \alpha_n$ – любые аргументы <процесса реакции>.

По директиве ИНТЕРПРЕТАТОРУ дается указание прервать <основной процесс> того элемента, который выдал директиву, и возбудить <процесс реакции> в момент, когда в системе будет объявлено событие с заданным именем и кодом. До наступления такого события продолжается реализация <основного процесса>.

8. ↑ УНПР ␣ <имя события> ␣ α_1,

α_1 – код события.

Директива отменяет запланированный процесс реакции в элементе, в котором она объявлена.

9. ↑ ПОВТ ␣ <НПИ> ␣ $\alpha_1, \ldots,$ ␣ α_n,

$\alpha_1, \ldots, \alpha_n$ – аргументы.

Директива дает указание повторить программу элемента с последней в ИНТЕРПРЕТАТОРЕ отмеченной "точки возврата", которая заранее объявлена при помощи директивы СОБ.

10. ↑ ВВТТ ␣ α_1 ␣ α_2,

α_1 – начальный математический адрес для ввода данных;

α_2 – объем памяти в байтах.

Директива служит для ввода данных с терминала в поле памяти того элемента, который ее объявил.

11. ↑ ВЫВТТ ␣ α_1 ␣ α_2 ␣ $\alpha_3, \ldots,$ ␣ α_n,

α_1 – начальный математический адрес выводимых данных;

α_2 – объем массива в байтах;

$\alpha_3, \ldots, \alpha_n$ – специальные аргументы.

Директива служит для вывода данных пользователю НИС через терминал.

12. ↑ НАЧТ ␣ α_1,

α_1 – начальный код времени.

По директиве ИНТЕРПРЕТАТОР начинает счет непрерывного времени.

13. ↑ ОПРТ ␣ α_1,

α_1 – математический адрес для записи кода времени.

По директиве опрашивается счетчик непрерывного времени и код записывается в соответствующий адрес.

14. ↑ ЖДАТЬ ␣ α_1,

α_1 – код времени в миллисекундах.

Процесс, который выдал директиву, приостанавливается на указанное время.

15. ↑ СОБ ␣ ⟨имя события⟩ ␣ α_1 ,

α_1 – код события.

По директиве ИНТЕРПРЕТАТОРУ сообщается событие элемента, ⟨НПИ⟩ которого указан в ⟨имени события⟩ . Если это событие помечено ⟨идентификатором нормального события⟩ , после запоминания его имени и кода ИНТЕРПРЕТАТОР в работу соответствующего элемента не вмешивается. Если в директиве указан ⟨идентификатор случайного события⟩ и код "I", основной процесс соответствующего элемента временно приостанавливается. Проверяется наличие запланированного процесса реакции на данное событие, и, если такое имеется, возбуждается его программа. По окончании процесса реакции восстанавливается прерванный основной процесс , а имя соответствующего случайного события отмечается кодом "0".

Окончание процесса реакции или основного процесса должно быть задано директивой СОБ, употребляя идентификатор события КОН с кодом "I".

При отсутствии запланированного процесса реакции , соответствующего случайному событию, этот факт через терминал сообщается пользователю НИС. Работа элемента в этом случае может быть восстановлена по директиве ПЛАН, которая сообщается через терминал.

Кроме вышерассмотренных идентификаторов РВИ, ЗВИ, КОН, в ИНТЕРПРЕТАТОРЕ употребляется идентификатор ТОЧК, служащий для обозначения точки возврата, с которой по директиве ПОВТ в случае сбоя системы повторяется программа основного процесса .

16. ↑ ОПРС ␣ ⟨имя события⟩ ␣ α_1 ,

α_1 – адрес для засылки кода события.

По этой директиве происходит опрос и засылка кода события в указанный адрес элемента.

17. ↑ КОГДА ␣ ⟨имя события⟩ ␣ α_1 ␣ ⟨имя события⟩ ␣ α_n ,

$\alpha_1, \ldots, \alpha_n$ – коды событий.

Процесс элемента, выдавшего эту директиву, приостанавливается до момента, пока коды всех указанных событий придут в соответствие с заданными значениями.

Авторы отдают себе отчет в том, что рассмотренная система директив в основном позволяет описывать задачи координации ИИС только в категориях, которые в работе [3] названы "средствами для управления в малом". Однако для создания быстродействующих ИИС, работающих в реальном масштабе времени, крайне важно выяснить средства координации иерархических мультипрограммных структур программного обеспечения ЭВМ. В связи с этим отметим основные принципы координации, условия реализации которых предусмотрены в настоящей системе директив:

1. Принцип координации процесса с упреждающими действиями осуществляется при помощи <процессов реакции> и директивы ПЛАН. Последняя обычно употребляется в координирующем элементе перед возбуждением координируемого процесса директивой СТАРТ;

2. Принцип повторения программы в случае сбоя в системе, начиная с места в <процессе>, которое заранее помечено согласно директиве

↑ СОБ ␣ <НПИ> ␣ ТОЧВ ␣ 1,

Повторение осуществляется по директиве ПОВТ;

3. Принцип асинхронного управления событиями, осуществляемый при помощи директив КОГДА или ОПРС в ожидающем элементе и директивы СОБ в ожидаемом элементе;

4. Принцип передачи собственного ресурса машинного времени подчиненным уровням организации может быть реализован в конкретном элементе при помощи директив КОГДА или ↑ СОБ ␣ <НПИ> ␣ КОН ␣ 1.

ЛИТЕРАТУРА

1. Темов В.Л. Принципы языка высокого уровня для управления экспериментом.-В кн.: Системы автоматизации научных исследований. Рига, "Зинатне", 1973, с.120-123.

2. Месарович М., Мако Д., Такахара И. Теория иерархических многоуровневых систем. М., "Мир", 1973. 343 с.

3. Бусленко Н.П., Калашников В.В., Коваленко И.Н. Лекции по теории сложных систем. М., "Советское радио", 1973. 438 с.

4. Тале И.А. Системный анализ и оценка целесообразности спектрально-кинетических люминесцентных методов исследований твердого тела. - Уч.зап. ЛГУ им.П.Стучки, т.160. Кибернетизация научного эксперимента, вып.3. Рига, 1972, с.11-25.

5. Дийкстра Э.В. Создание машинно-независимых языков программирования.-В кн.: Современное программирование, вып.1. М., "Советское радио", 1966, с.227-247.

УДК 65.01

Ю.Я.Кузьмин

ОЦЕНКА СТЕПЕНИ НЕОПТИМАЛЬНОСТИ РЕШЕНИЯ ПРИ СИСТЕМНОМ ПОДХОДЕ

В современной литературе появилось значительное количество работ, посвященных системному подходу при решении различных научно-технических и организационных задач. В данной работе сделана попытка рассмотреть слабо освещенную в литературе проблематику оценки точности решения задач при использовании методики системного подхода.

Системный подход является одним из наиболее комплексных подходов при решении современных научно-технических и организационных проблем [1–4]. Традиционная схема решения задачи в духе системного подхода состоит в следующем.

На этапе постановки задачи определяются основные цели, которые должны быть достигнуты, выявляются факторы (F), влияющие на принятые решения по выбору оптимальной для данной задачи альтернативы. На стадии генерации альтернатив (A_j) предлагаются различные средства достижения основной цели. Следующий этап связан с выбором оптимальной альтернативы из всех предложенных. Здесь определяется сама процедура выбора, взвешиваются факторы, учитываемые процедурой выбора, производится оценка каждой из альтернатив по всему перечню факторов и, наконец, на основе принятой процедуры выбора определяется оптимальная в данных условиях альтернатива.

Каждая из перечисленных выше процедур содержит ряд моментов, вносящих свой вклад в неоптимальность конечного результата.

Перечень выявленных факторов может быть неполным за счет того, что отбрасываются менее значительные факторы; кроме того, ряд факторов может быть просто неизвестен на стадии решения проблемы. Совокупность неучтенных факторов, естественно, вносит свой вклад в неоптимальность решения тем большую, чем больше эта совокупность. К этому же ведет и фиксация процедуры взвешивания факторов и оценки альтернатив.

Неоптимальность решения непосредственным образом зависит от общего количества альтернатив, среди которых ищется оптимальная. Всегда имеется неуверенность в полноте набора предложенных альтернатив, среда которых может отсутствовать действительно оптимальная альтернатива.

Наконец, неоптимальность решения зависит от критерия выбора альтернативы. Чаще всего в качестве критерия выбора принимается некоторая функция, например:

$$R_j = \sum_i \rho_i \alpha_{ij} \qquad (I)$$

где ρ_i - веса i -факторов, α_{ij} - коэффициенты соответствия j - альтернативы i -фактору. Однако в этом случае при оценке согласно (I) может быть отдано предпочтение альтернативе, набравшей абсолютное большинство хороших оценок по ряду факторов, но получившей худшую оценку по обязательному для альтернативы фактору в сравнении с альтернативой, получившей средние оценки. Это может случиться, например, при определении ρ_i , исходя из процедуры ранжирования.

Рассмотрим ряд затронутых вопросов на конкретном примере решения задачи определения целесообразности применения ЭВУМ "Днепр-I" для управления одним теплофизическим экспериментом.*

Требуется оценить три альтернативы управления экспериментом:

А1 - с помощью ЭВУМ "Днепр-I",

А2 - локальными автоматами (т.е. специализированными для каждой функции приборами),

А3 - ручное управление.

После обсуждения было выяснено, что при оценках необходимо учитывать следующие факторы:

Ф1. Простота эксплуатации аппаратуры.

Ф2. Дешевизна эксперимента.

Ф3. Безопасность эксперимента.

Ф4. Простота сбора информации и контроля.

Ф5. Качество эксперимента.

Ф6. Возможность развития аппаратуры.

Ф7. Простота управления экспериментом.

Ф8. Простота диалога с аппаратурой.

Ф9. Быстрота всей разработки.

---

* См. настоящий сборник, с. III.

Для того, чтобы принять решение, необходимо прежде всего выбрать способ оценки альтернатив и факторов, которым и будем пользоваться.

Существуют два способа оценок в системном анализе: абсолютный и относительный. В первом случае, когда возможно проводить инженерно-технические расчеты, осуществляется моделирование альтернативы и определяются связи между параметрами, а также вычисляются сами параметры для каждой альтернативы. Процедура выбора при этом сводится к определению альтернативы с наилучшими параметрами. В качестве примера такой ситуации можно привести выбор приборов с минимальной погрешностью измерения и максимальным быстродействием. Однако далеко не всегда возможен инженерно-технический расчет, как, например, в рассматриваемой выше задаче и, кроме того, зачастую совокупность параметров, по которым производится выбор, не может быть определена из одной модели и, следовательно, эти параметры не связаны общими соотношениями. В подобном случае обычно пользуются экспертными оценками. Назначается коллектив экспертов, задача которого состоит в определении степени важности каждого из факторов и коэффициентов соответствия альтернатив факторам. При экспертных оценках наиболее простым является применение процедуры ранжирования факторов по важности и альтернатив по степени соответствия факторам. При этом вес i-фактора может быть определен, например, следующим образом:

$$\rho_i = \frac{2(N-i+1)}{N(N+1)} \qquad (2)$$

Здесь: N – общее число факторов, i – ранжированный номер фактора ($i = 1$ для наиболее важного фактора).

Анализ приведенного выше примера дал следующие веса факторов: 0,133; 0,089; 0,200; 0,044; 0,178; 0,022; 0,111; 0,067; 0,156.

Допустим, что при учете числа факторов мы ошиблись в $k = \frac{M}{N}$ раз, где M – предполагаемое общее число факторов. Какова будет ошибка в ρ ? Легко показать, что

$$\Delta\rho_i = \rho_i(N) - \rho_i(M \geq N) = \frac{2}{kN}\left[(1-k) + \frac{k(kN+1)-(N+1)i}{(kN+1)(N+1)}\right]$$

Это составит для приведенного примера (для к = 2) соответственно: -0,046; -0,014; -0,094; 0,018; -0,078; 0,034; -0,030; 0,006; -0,062.

Критерий выбора в этом случае будет:

$$R_j = \frac{2}{N}\sum_{i=1}^{N} \alpha_{ij} - \frac{2}{N(N+1)}\sum_{i=1}^{N} i\alpha_{ij} \qquad (3)$$

Здесь: α_{ij} - ранжированный показатель соответствия; j - альтернативы i - фактору (α_{ij} = I для наилучшего соответствия).

Как видим, такая процедура взвешивания приводит к простым функциям. В этот же класс попадают и другие процедуры взвешивания, приводящие к линейным весовым функциям. Однако подобные процедуры не охватывают случай существенно неоднородных факторов и альтернатив, который характерен для реальных ситуаций. Имеются в виду, например, такие множества факторов, которые содержат равноценные факторы, или же смежные по важности факторы имеют существенно разные показатели (допустим, i -фактор важнее, чем $(i+1)$-, ..., $(i+n)$ -факторы, вместе взятые). В подобных случаях вместо процедуры ранжирования применяются оценки в некоторой многобалльной системе. Назначаться могут относительные и абсолютные (в данной системе факторов и альтернатив) оценки.

Относительные оценки назначаются следующим образом. Имеющиеся факторы ранжируются по важности, затем наиболее важному i -фактору присваивается максимальная оценка (В), последующим факторам - меньше (b_i). Альтернативы ранжируются в пределах каждого фактора, затем альтернативе, наиболее соответствующей фактору, присваивается максимальная оценка (А), остальные получают меньшие (α_{ij}), с учетом разницы в степени соответствия.

Критерий выбора в этом случае будет:

$$R_j = \left(A - \frac{\sum_{i=1}^{n} b_i \alpha_{ij}}{\sum_{i=1}^{n} b_i} \right. \qquad (4)$$

Иногда удобнее использовать нормированный критерий, позволяющий сравнивать альтернативы с учетом "худшей" и "лучшей" в данной ситуации. Легко видеть, что значения этих граничных альтернатив, согласно (I), будут:

$$R_{min} = a = \min\{\alpha_{ij}\},$$
$$R_{max} = A = \max\{\alpha_{ij}\}. \qquad (5)$$

В свою очередь для оценок по (4):

$$R_{min} = 0,$$
$$R_{max} = A - a. \qquad (6)$$

С учетом (5) и (6) нормированными оценками для (I) и (4) являются соответственно:

$$r_j = \frac{R_j - a}{(A - a)} \qquad (7)$$

$$r'_j = \frac{R_j}{(A - a)} \qquad (8)$$

Эти оценки позволяют судить, насколько близка данная альтернатива к "лучшей" в смысле полного удовлетворения заданным факторам. Основные источники неоптимальности здесь заключаются в задании A и a.

Другой источник неоптимальности, как уже отмечалось выше, заключается в ограничении числа факторов, влияющих на выбор альтернативы. Можно попытаться найти граничные значения оценок, если допустить что помимо известных факторов могут действовать (M - N) неизвестных. По (I):

$$R_j(M) = \sum_{i=1}^{M} \rho_i \alpha_{ij} = \sum_{i=1}^{N} \rho_i(M)\alpha_{ij} + \sum_{i=N+1}^{M} \rho_i(M)\alpha_{ij}, \qquad (9)$$

Нижняя оценка будет:

$$R'_j(M) = \sum_{i=1}^{N} \rho_i(M)\alpha_{ij} + a \sum_{i=N+1}^{M} \rho_i(M) \qquad (10)$$

а верхняя

$$R''_j(M) = \sum_{i=1}^{N} \rho_i(M)\alpha_{ij} + A \sum_{i=N+1}^{M} \rho_i(M) \qquad (11)$$

Здесь (M) означает, что оценки берутся для M -факторов. Для нормированного критерия (7)

$$\frac{R'_j(M) - a}{(A - a)} \leqslant r'_j \leqslant \frac{R''_j(M) - a}{(A - a)} \qquad (12)$$

и, соответственно, для (8):

$$r''_{j\,min} \leq r''_j \leq r''_{j\,max}$$

где $r''_{j\,min} = \dfrac{A - \sum_{i=1}^{N} \rho_i(M)\alpha_{ij} + a\sum_{i=N+1}^{M} \rho_i(M)}{(A-a)}$

$$r''_{j\,max} = \frac{A - \sum_{i=1}^{N} \rho_i(M)\cdot\alpha_{ij} + A\sum_{i=N+1}^{M} \rho_i(M)}{(A-a)}$$

В заключение приведем результаты применения описанных процедур для нахождения лучшей альтернативы рассмотренного ранее примера.

Таблица I

n	$\rho(9)$	$\rho(18)$	$\rho'''(9)$
I	0,I33	0,088	0,I26
2	0,089	0,076	0,090
3	0,200	0,I05	0,I80
4	0,044	0,065	0,054
5	0,I78	0,099	0,I7I
6	0,022	0,058	0,045
7	0,III	0,082	0,II7
8	0,067	0,070	0,072
9	0,I56	0,094	0,I44

Обозначения в таблице следующие:

ρ (9), ρ (I8) - веса, соответствующие номерам ранжированных 9 известных факторов и 9 дополнительных неизвестных, но менее важных факторов;

$\rho'''(9)$ - веса факторов согласно экспертной оценке.

Таблица 2

n	A'_1	A''_1	A'_2	A''_2	A'_3	A''_3
I	2	6	3	3	I	I0
2	3	2	2	6	I	I0
3	I	I0	2	6	3	3
4	I	I0	2	6	3	4
5	I	I0	2	7	3	2
6	I	I0	3	4	2	7
7	I	I0	2	8	3	6
8	I	I0	2	7	3	4
9	2	5	3	3	I	I0
R(9)	I,47	2,02	2,3I	4,04	2,22	4,00
R'(18)	I,34	I,43	I,98	3,06	I,90	2,73
R''(18)	I,86	—	2,50	—	2,42	—
R'''(9)	I,45	I,95	2,3I	4,07	2,23	4,00
r (9)	0,23	0,20	0,66	0,40	0,6I	0,40
r'(18)	0,I7	0,I4	0,49	0,3I	0,45	0,27
r''(18)	0,43	0,4I	0,75	0,60	0,7I	0,54
r'''(9)	0,22	0,I9	0,66	0,4I	0,62	0,40
$\bar{r}$	0,25		0,53		0,50	
$\pm\Delta$	0,09		0,I3		0,II	

Здесь приняты обозначения:

A' – коэффициенты соответствия альтернативы факторам согласно ее ранжированному номеру;

A'' – коэффициенты соответствия альтернативы факторам в соответствии с методикой экспертных оценок (I0-балльная система);

$R(9)$ – общая оценка альтернативы по 9 известным ранжированным факторам;

$R'(I8)$, $R''(I8)$ – граничные оценки альтернативы по 9 известным и 9 предполагаемым ранжированным факторам;

$R'''(9)$ – общая оценка альтернативы по 9 факторам с весами ρ''';

$r(9)$, $r'(I8)$, $r''(I8)$, $r'''(9)$ – нормированные оценки, соответствующие $R(9)$, $R'(I8)$, $R''(I8)$, $R'''(9)$.

$\bar{r}, \pm\Delta$ – усредненные оценки и отклонения.

Анализ данных таблицы I показывает, что первая альтернатива предпочтительней двух других для сравнительно широких вариаций исходных данных. Различимость второй и третьей альтернатив неубедительна. Видно, что выбор способа оценок существенно сказывается на значении R и r. Ранжирование в данном случае дает значения более высокие, чем в случае экспертных оценок.

Отметим, что оптимальность решения в соответствии с системным подходом зависит как от достоверности исходных данных, так и от способа их использования для принятия решения. В каждом конкретном случае необходимо оценивать величину неоптимальности решения.

ЛИТЕРАТУРА

I. Мейстер Д., Рабидо Дж. Инженерно-психологическая оценка при разработке систем управления. М., "Сов.радио", I970. 342 с.

2. Оптнер С.Л. Системный анализ для решения деловых и промышленных проблем. М., "Сов.радио", I97I. 215 с.

3. Диксон Дж. Проектирование систем: изобретательство, анализ и принятие решений. М., "Мир", I969. 440 с.

4. Лопухин М.М. PATTERN – метод планирования и прогнозирования научных работ. М., "Сов.радио", I97I. I82 с.

УДК 681.3.06:007.5

Х.Р.Краузе-Крузе

ЯЗЫК МАКРОАССЕМБЛЕРА ЭЦВМ "ДНЕПР-21"

Рассматриваются два уровня языка макроассемблера – основной уровень, близкий к машинному языку, и макроуровень, служащий для упрощения записи часто встречающихся последовательностей строк; дается описание основного уровня и способов построения макроуровня.

Язык макроассемблера (МА) является языком программирования, открытым для пользователя, т.е. пользователь может развивать его в нужном ему направлении. Поэтому в языке МА различаются два уровня:

– основной уровень, который встроен в транслятор и является общим для всех пользователей; ниже дается его описание;

– макроуровень, который каждый пользователь строит для себя; способы его построения рассмотрим ниже.

Для определения языка используются средства, указанные в статье (см.с. 47).

Программа на языке МА состоит из строк. Для записи строк разрешается использовать <основные знаки> . * Строки подразделяются на основные строки и макростроки. Макростока является более короткой записью часто встречающейся последовательности строк (в том числе макростроек).

Последовательность строк, транслируемая независимо от других строк, называется текстом сегмента, а результат трансляции – сегментом . Программа может состоять из одного или несколько сегментов. Сегменты в программу объединяются при помощи собирающей программы. ** Синтаксис текста программы описывает следующая формула:

$$\langle \text{текст сегмента} \rangle ::= \{\{\langle \text{заказ памяти} \rangle\}_n \mid \langle \text{пусто} \rangle\} \{\{\langle \text{основная строка} \rangle \backslash \langle \text{заказ памяти} \rangle\} \mid \langle \text{макростока} \rangle\}_n$$

---

* См.настоящий сборник, с. 47

** См.настоящий сборник, с. 50.

Основной уровень языка

⟨основная строка⟩ :: = ⟨команда⟩,|⟨константа⟩,| ⟨заказ памяти⟩,| ⟨описание глобальных адресов⟩,| ⟨описание внутренних адресов⟩,|⟨описание используемых сегментов⟩,| ⟨описание символического адреса⟩,| ⟨описание макрокоманды⟩,| ⟨комментарий⟩ | ⟨блок⟩.

Главным типом строк основного уровня являются команды. Каждой команде языка МА соответствует одна машинная команда. По своей структуре МА-команды ничем не отличаются от машинных команд.

⟨команда⟩ :: = ⟨метка⟩ ⟨коп⟩ $\{\sqcup$ ⟨операнд⟩$\}_n$|⟨коп⟩ $\{\sqcup$ ⟨операнд⟩$\}_n$

⟨метка⟩ :: = ⟨идентификатор⟩:

⟨идентификатор⟩ :: = ⟨буква⟩ | ⟨буква⟩$\{$⟨буква⟩|⟨цифра⟩$\}_6$

⟨коп⟩ :: = ⟨код операции⟩

Коды операций мнемонические; эти коды, а также соответствие между ними и численными кодами операций задаются таблицей (см. с. 45).

Аргументы операции могут быть заданы непосредственно (значения) или при помощи адресов первого или второго рангов.

⟨операнд⟩ :: = ⟨значение⟩ | ⟨адрес операнда⟩

⟨адрес операнда⟩ :: = ⟨адрес первого ранга⟩ | ⟨адрес второго ранга⟩ | ⟨адрес второго ранга с индексацией⟩

⟨адрес первого ранга⟩ :: = ⟨адрес⟩

⟨адрес второго ранга⟩ :: = (⟨адрес⟩)

⟨адрес второго ранга с индексацией⟩ :: = (⟨адрес⟩) И

⟨адрес⟩ :: = ⟨численный адрес⟩ | ⟨символический адрес⟩

⟨численный адрес⟩ :: = $\{\langle C_8\rangle\}_6 . \langle C_4\rangle$

⟨символический адрес⟩ :: = ⟨идентификатор⟩$\{$⟨целое⟩|⟨пусто⟩$\}$

Семантика численных адресов дана в работе [1].

Установление соответствия между символическими и численными адресами будет рассмотрено ниже. Если за идентификатором следует целое, то это целое прибавляется к численному адресу, соответствующему этому идентификатору.

В командах, имеющих операндами d-константы, вместо последних употребляются соответствующие символические адреса[I]

⟨значение⟩::=⟨целое⟩|⟨число⟩|⟨непосредственно заданный адрес⟩|⟨код⟩|⟨посимвольное значение⟩|⟨номер сигнала прерывания⟩

⟨целое⟩::=⟨8-целое⟩|⟨10-целое⟩

⟨8-целое⟩::=.{+|-}{⟨C_8⟩}$_n$

⟨10-целое⟩::={+|-}{⟨цифра⟩}$_n$

⟨число⟩::=⟨8-число с фиксированной запятой⟩|⟨10-число с фиксированной запятой⟩|⟨8-число с плавающей запятой⟩|⟨10-число с плавающей запятой⟩

⟨8-число с фиксированной запятой⟩::={+|-}·{⟨C_8⟩}$_n$

⟨10-число с фиксированной запятой⟩::={+|-}{⟨цифра⟩}$_n$

⟨8-число с плавающей запятой⟩::=.{+|-}{⟨C_8⟩}$_3$ {+|-}{⟨C_8⟩}$_n$

⟨10-число с плавающей запятой⟩::={+|-}{⟨цифра⟩}$_2$ {+|-}{⟨цифра⟩}$_n$

Целые (числа), не содержащие незначащих нулей, занимают наименьшее возможное количество символов. Каждый незначащий нуль удлиняет целое (число) на один символ.

⟨непосредственно заданный адрес⟩::=:⟨адрес операнда⟩

⟨код⟩::=*{{⟨двоичное содержимое символа⟩|⟨восьмеричное содержимое символа⟩}{M|H}}$_n$

⟨двоичное содержимое символа⟩::=⟨C_2⟩⟨C_2⟩⟨C_2⟩⟨C_2⟩⟨C_2⟩⟨C_2⟩⟨C_2⟩⟨C_2⟩

⟨восьмеричное содержимое символа⟩::=⟨C_4⟩⟨C_8⟩⟨C_8⟩

буква M или H после содержимого символа обозначает наличие или отсутствие маркера

⟨посимвольное значение⟩::={⟨основной знак⟩\␣,}$_n$

В посимвольном значении маркер будет проставлен только в последнем символе.

⟨номер сигнала прерывания⟩::=≠-⟨C_8⟩⟨C_8⟩⟨C_8⟩⟨C_8⟩⟨C_8⟩/⟨C_4⟩⟨C_8⟩⟨C_8⟩|≠+⟨C_8⟩⟨C_8⟩⟨C_8⟩⟨C_8⟩⟨C_8⟩

Первый способ записи предусмотрен для номеров сигналов прерывания второго рода [I].

⟨константа⟩::={⟨метка⟩|⟨пусто⟩}К{␣⟨значение⟩}$_n$

Константы служат для записи значений вне команд.

⟨заказ памяти⟩::={Р|Р.}␣⟨целое⟩{␣⟨идентификатор⟩}$_n$

Заказ памяти служит для присвоения символических адресов данным в рабочем поле. В нем резервируется несколько зон одинаковой длины, определяемой целым. Первому символу каждой зоны присваивается символический адрес – идентификатор. Резервирование начинается с того места в рабочем поле, в котором оно было закончено по предыдущему заказу памяти. Если заказ памяти начинается с Р., то резервирование начинается обязательно с начала ячейки.

⟨описание глобальных адресов⟩ :: = ГЛ $\{\sqcup$⟨идентификатор⟩$\}_n$

В описании перечисляются символические адреса – метки, определенные в других сегментах.

⟨описание внутренних адресов⟩ :: = ВН $\{\sqcup$⟨идентификатор⟩$\}_n$

В описании перечисляются те метки, которые разрешается использовать в других сегментах.

⟨описание используемых сегментов⟩ :: = ИС $\{\sqcup$⟨имя сегмента⟩$\}_n$

В описании перечисляются имена тех сегментов, метки которых использует сегмент.

⟨описание символического адреса⟩ :: = Э$\sqcup$Э⟨идентификатор⟩ $\sqcup$⟨адрес⟩

Использование адреса и идентификатора после такого описания равноправно.

⟨описание макростроки⟩ :: = МК $\sqcup$ ⟨код макростроки⟩, ⟨программа развертки⟩ КМ

Описание макростроки определяет процедуру переработки макростроки в последовательность строк (развертку макростроки). Понятие программы развертки будет рассмотрено при описании макроуровня языка.

⟨комментарий⟩ :: = $'\{$⟨основной знак⟩$\backslash'\}_n'$

Комментарий облегчает чтение напечатанной программы и не влияет на результат трансляции.

⟨блок⟩ :: = $\{$⟨метка⟩ | ⟨пусто⟩$\}$ БЛОК, $\{$ ⟨заказ памяти⟩ | ⟨пусто⟩$\}_n$ $\{\{$⟨основная строка⟩$\backslash$⟨заказ памяти⟩/⟨описание внутренних адресов⟩$\}$ | ⟨макростока⟩$\}_n$ КОНЕЦ

Символические адреса в рабочем поле и метки, определенные во внутреннем блоке недоступны внешнему блоку или сегменту. Объявлять метки блока внутренними глобальными адресами запрещается.

Макроуровень языка

Новые типы макростроk вводятся при помощи описаний макростроk. Рассмотрим требования к программам развертки.

Программа развертки – это подпрограмма транслятора, написанная пользователем. Аргументами подпрограммы являются адрес массива адресов аргументов макростроки и адрес фиксатора результата развертки макростроки. Перед выходом из программы развертки нужно засылать в сумматор отрицательное число, если аргументы проверяются и найдена ошибка, или нестрицательное число в противном случае.

Массив адресов аргументов состоит из трехсимвольных адресов аргументов макростроки. Аргумент макростроки – это группа последовательно размещенных знаков, начинающаяся за кодом макростроки или предыдущим аргументом и кончающаяся пробелом или запятой. Далее,

$$\langle \text{код макростроки} \rangle ::= \{\langle \text{основной знак} \rangle \backslash \sqcup, :\}_6$$

При пересылках аргументов следует иметь в виду, что особые знаки окаймлены маркерами.

Если макростроka имеет собственные идентификаторы, не зависящие от аргументов, то они записываются по особому правилу:

$$\langle \text{идентификатор в макростроке} \rangle ::= \langle \text{буква} \rangle \{\langle \text{пусто} \rangle \mid \{\langle \text{буква} \rangle \mid \langle \text{цифра} \rangle\}_5\}$$

Так как для пересылки в формируемую строку задан не сам фиксатор, а его адрес, следует позаботиться об изменении этого фиксатора, чтобы он всегда указывал на свободное место в конце формируемой строки.

Написание программ развертки можно облегчить созданием специальных типов макростроk.

Таблица

Коды операций

мнем. код	маш. код	мнем. код	маш. код	мнем. код	маш. код	мнем. код	маш. код
+Д1	300	-Ц5	364	/Д7	356	ФД3	253
+Д2	301	-Ц6	365	/Д8	357	+П1	204
+Д3	302	-Ц7	366	НРМ1	255	+П2	205
+Д4	303	-Ц8	367	НРМ2	257	+П3	206
+Ц1	320	-М1	334	СДА1	220	+П4	207
+Ц2	321	-М2	336	СД..2	221	+Т1	374
+Ц3	322	*Д1	310	СДА^	222	+Т2	375
+Ц4	323	*Д2	311	СДАМ1	224	+Т3	376
+Д5	340	*Д3	312	СДАМ2	225	+Т4	377
+Д6	341	*Д4	313	СДАМ3	226	.Л1	210
+Д7	342	*Ц1	330	ПД1	275	.Л2	211
+Д8	343	*Ц2	331	ПД2	276	.Л3	212
+Ц5	360	*Ц3	332	ПД3	277	.Л4	213
+Ц6	361	.Ц4	333	ПД5	270	+Л1	200
+Ц7	362	.Д5	350	ПД6	271	+Л2	201
+Ц8	363	.Д6	351	ПД7	272	+Л3	202
-Д1	304	.Д7	352	ПД8	274	+Л4	203
-Д2	305	.Д8	353	ПЦ1	265	СДЛ1	230
-Д3	306	.Ц5	370	ПЦ2	266	СДЛ2	231
-Д4	307	.Ц6	371	ПЦ3	267	СДЛ3	232
-Ц1	324	.Ц7	372	ПЦ5	260	ССЕ	100
-Ц2	325	.Ц8	373	ПЦ6	261	ССГ	101
-Ц3	326	/Д1	314	ПЦ7	262	ССЛ	102
-Ц4	327	/Д2	315	ПЦ8	264	ПСЕ	104
-Д5	344	/Д3	316	ПЗН1	214	ПСГ	105
-Д6	345	/Д4	317	ПЗН2	216	ПСЛ	106
-Д7	346	/Д5	354	ФД1	251	ПСЛП	046
-Д8	347	/Д6	355	ФД2	252	ФСЛП	107
ФСЛ2	117	УПП	036	ОД	007	ЧТРАК	053
МН	144	УПНП	037	СТОП	007	4ТРАК	053

мнем. код	маш. код	мнем. код	маш. код	мнем. код	маш. код	мнем. код	маш. код
МНГ	I45	БП	02I	ПРБ	060	ЧТРМ	05I
МЕ	I54	БПА	043	ОРБ	064	4ТРМ	05I
МЕС	I74	НОП	000	6ПШ	023	ЧТР6	050
МЕГ	I55	УПУО	022	БПША	063	4ТРБ	050
СС	I27	УПНЗ	020	ВП	075	ЧТРН	052
ССМ	I67	ИНДI	055	ПАДР	057	4ТРН	052
УПРБ	033	ИНД2	056	ПАДП	044	УО	006
УПМ	032	КЦI	026	ПАДПМ	045	ЕП	0I2
УПМР	030	КЦ2	027	ПРМ	06I	НП	0I3
УПБ	03I	РПЗ	002	ОРМ	065	ЗАЖ	070
УПР	034	РПЗ	003	С	062	ГАС	07I
УПНР	035	РПП	0I0	ЗПРМ	04I	ДМСК	072
УПМЕ	024	РПШ	0II	ЗПРБ	040	МСК	073
УПМН	025	РП	004	ЗПРН	042	КОНЕЦ	077

ЛИТЕРАТУРА

I. Управляющая система "Днепр-2" комплекса "Днепр-2I". Математическое обеспечение, т.I. Киев, I970. 243 с.

УДК 681.3.06:007.5

Х.Р.Краузе-Крузе

СИСТЕМА ПРОГРАММИРОВАНИЯ МАКРОАССЕМБЛЕРА ЭЦВМ "ДНЕПР-21"

Рассматривается набор приказов управления работой системы программирования макроассемблера, служащих для ввода, редактирования и трансляции программ на языке макроассемблера, а также для ввода, вывода и хранения информационных массивов любого вида.

Система программирования макроассемблера (МА) состоит из языка макроассемблера и ряда программ, при помощи которых программа, записанная на языке МА (МА-текст или текст), преобразуется в программу на машинном языке. Описание языка МА дано в статье "Язык макроассемблера ЭЦВМ "Днепр-21". * Программы МА-системы - макрогенератор, транслятор, собирающая программа и сервисные программы размещаются на системной магнитной ленте (СМЛ). Пользователь может иметь индивидуальную магнитную ленту (ИМЛ).

Управление работой системы происходит при помощи приказов, вводимых с перфоленты. Для записи форм приказов использованы формулы Бэкуса. Применены также следующие конструкции, сокращающие запись этих формул:

- конструкция вида $\{\langle A_1\rangle|\langle A_2\rangle|\ldots|\langle A_n\rangle\}$ означает, что вместо нее можно подставлять любую из металингвистических переменных $\langle A_i\rangle (1\leqslant i\leqslant n)$;
- если металингвистическая переменная $\langle A\rangle$ определена следующим образом:

$$\langle A\rangle ::= \langle A_1\rangle|\langle A_2\rangle|\ldots|\langle A_n\rangle;$$

то конструкция $\{\langle A\rangle\backslash\langle A_{i1}\rangle\langle A_{i2}\rangle\ldots\langle A_{ik}\rangle$ означает, что вместо нее можно подставлять любое из $A_j (1\leqslant j\leqslant n; j\neq i_1, i_2,\ldots, i_k)$;
- если в упомянутых конструкциях после закрывающей фигурной скобки следует индекс - число к , то эту конструк-

* См. настоящий сборник, с. 40.

цию можно повторить от одного до K раз; если индексом является буква n , то число повторений произвольное.

Такие же конструкции использованы и при определении синтаксиса языка МА. Для записи приказов и текстов используется набор знаков по ГОСТ 10859-64.

⟨знак⟩ :: = ⟨служебный знак⟩ | ⟨основной знак⟩
⟨служебный знак⟩ :: = x | ┐ | ≡ | !
⟨основной знак⟩ :: = ⟨особый знак⟩ | ⟨буква⟩ | ⟨цифра⟩ |

¯ | ≤ | ≥ | ∨ | ∧ | ⊃ | - | % | ◊ | | | - | _ | „ | ъ | ° | ' | ← | → | ? | ↓ | ± | ∇

⟨особый знак⟩ :: = + | - | / | , | . | ␣ | ₁₀ | ↑ | (|) | x | = | ; | [|] | * | ' | ' | ≠ | < | > | :
⟨буква⟩ :: = А | Б | В | Г | Д | Е | Ж | З | И | Й | К | Л | М | Н | О | П | Р | С | Т | У | Ф | Х | Ц | Ч | Щ |

Щ | Ы | Ь | Э | Ю | Я | D | F | G | I | J | L | N | Q | R | S | U | V | W | Z

⟨цифра⟩ :: = ⟨C_8⟩ | 8 | 9
⟨C_8⟩ :: = ⟨C_4⟩ | 4 | 5 | 6 | 7
⟨C_4⟩ :: = ⟨C_2⟩ | 2 | 3
⟨C_2⟩ :: = 0 | 1

Служебные знаки не могут быть использованы для записи текстов.

x означает "граница зоны".

Для исправления ошибок, замеченных при перфорации, служит знак ┐ , означающий отказ от знака, предшествующего ему, а также от остальных предшествующих знаков до ближайшей запятой или до начала приказа. Знаки ≡ и ! используются в качестве разграничителей в приказе СЕГМЕНТ.

Рассмотрим различные типы приказов.

I. Приказ ПРОГРАММА; форма записи:

x ПРОГРАММА␣ ⟨имя программы⟩ ␣ ⟨имя ИМЛ⟩ x;
⟨имя программы⟩ :: = $\{$⟨основной знак⟩ \ ␣, $\}_8$
⟨имя ИМЛ⟩ :: = $\{$⟨основной знак⟩ \ ␣, $\}_8$

При выполнении этого приказа запоминается имя программы и считывается каталог ИМЛ с заданным именем. Если приказ ПРОГРАММА применяется несколько раз, то задавать имя ИМЛ обязательно только при первом применении. Имя программы используется при распечатке текстов и при сборке.

2. Приказ КОНЕЦ; форма записи:

ϫ КОНЕЦ ϫ

При выполнении этого приказа происходит отказ от ИМЛ и имени программы.

3. Приказ СЕГМЕНТ; форма записи:

ϫ СЕГМЕНТ␣ ⟨имя сегмента⟩ ⟨адрес сегмента⟩ ,⟨указание о формировании транслируемого текста⟩⟨режимы вывода и трансляции⟩ϫ

Приказ СЕГМЕНТ позволяет:

- присвоить имя транслированному сегменту,
- формировать транслируемый текст,
- печатать сформированный текст, а также вывести его на перфоленту или ИМЛ,
- транслировать текст,
- вывести результат трансляции на перфоленту или ИМЛ.

⟨имя сегмента⟩ ::= {⟨основной знак⟩\␣,}$_7$

⟨адрес сегмента⟩ ::= ⟨математический адрес⟩ | ⟨пусто⟩

Метапонятие ⟨математический адрес⟩ определен в статье (см. с. 40); это начальный адрес программы, сборка которой начнется с данного сегмента; если адрес отсутствует, то он полагается равным 0.0.

⟨указание о формировании транслируемого текста⟩ ::= {⟨элемент формируемого текста⟩}$_n$

⟨элемент формируемого текста⟩ ::= ⟨имя сегмента⟩, | ⟨имя сегмента⟩{␣ ⟨область⟩}$_n$, | ≡{⟨основной знак⟩}$_n$ ≡

⟨область⟩ ::= ⟨номер строки⟩ | ⟨номер строки⟩-⟨номер строки⟩

Формирование предназначено в основном для исправления текстов; можно собрать текст также из частей текстов различных сегментов. Элементы формируемого текста размещаются один за другим в порядке их определения. Элемент текста - это или непосредственно заданный текст, или часть текста какого-либо сегмента. Непосредственно заданный текст окаймляется знаками ≡ . Для определения части текста какого-либо сегмента должны быть заданы имя сегмента и область. Область - это или номер строки, или последовательность строк, которая определяется номерами первой и последней строки. Строки по-

лучают номера при распечатке текстов.

<номер строки> :: = $\{$<цифра>$\}_4$

Часть приказа, следующая за восклицательным знаком, определяет обработку сформированного текста.

<режимы вывода и трансляции> :: = $\{$<вывод текста>$\}_n$<режим трансляции>, $\{$<вывод транслированного сегмента>$\}_n$! <пусто>

<вывод текста> :: = А|М|П <пусто>

<режим трансляции> :: Т|Р| <пусто>

<вывод транслированного сегмента> :: = <пусто>

Выводы кодируются буквами:

А - печать,

М - запись на ИМЛ,

П - вывод на перфоленту.

Режим трансляции задается буквами:

Т - трансляция,

Р - трансляция с последующей распечаткой параллельно исходного текста и результата трансляции в числовом коде. Если режимы вывода и трансляции не задаются, то по умолчанию они равны АМ,Т,М.

Вывод транслированного сегмента на перфоленту возможен, если сегмент является программой, т.е. в нем не определены глобальные и внутренние адреса.*

4. Приказ СБОРКА; форма записи:

х СБОРКА ␣ <имя сегмента> ␣ <запись> х

Приказ служит для образования программы из сегментов. В приказе задается имя сегмента, с которого начинается сборка. Адрес этого сегмента (см. приказ СЕГМЕНТ) является начальным адресом собираемой программы. Адреса других сегментов не учитываются. Вывод кодируется буквой:

М - запись на ИМЛ,

П - вывод на перфоленту.

При записи программы на ИМЛ название ее берется из приказа ПРОГРАММА.

5. Приказ СЕРВИС; форма записи:

х СЕРВИС х$\{$х$\{$ <приказ сервиса>,$\}_n$ х$\}_n$

---

* См. настоящий сборник, с. 43.

Во время работы приказа СЕРВИС теряется имя программы и состояние каталога ИМЛ; они должны быть восстановлены приказом ПРОГРАММА.

Для группы сервиса важно понятие поля. Поле – это область памяти, его содержимое может быть выведено на внешний носитель приказами сервиса. Заполняется поле также при помощи приказов сервиса.

Ниже приводится перечень приказов сервиса:

СМЛ ⟨имя МЛ⟩ – запись начальной структуры СМЛ на МЛ с данным именем;
⟨имя МЛ⟩::=$\{$⟨основной знак⟩\␣,$\}_8$

ИМЛ ␣⟨имя МЛ⟩ – запись начальной структуры ИМЛ на МЛ с данным именем;

КОП ␣ $\{$И|С$\}$␣⟨имя МЛ⟩ – копирование библиотеки ИМЛ (СМЛ) на МЛ с данным именем и печать каталога этой МЛ. На МЛ до копирования должна быть записана соответствующая начальная структура;

ПКАТ – печать каталога ИМЛ;

СГ – очистка поля;

ИМЯ ␣ ⟨имя массива⟩ – присвоение имени содержимому поля для записи на ИМЛ;
⟨имя массива⟩::=$\{\equiv|$⟨пусто⟩$\}$ $\{$⟨основной знак⟩\␣,$\}_7$ Если имя массива начинается со знака ≡ , то массив является текстом того сегмента, имя которого следует за ≡ ;

ЗП ␣ ⟨имя массива⟩ – запись содержимого поля с данным именем в библиотеку на ИМЛ;

ВЫВС ␣ {⟨пусто⟩ | АЦПУ} – вывод содержимого поля на перфоленту или печать в посимвольном режиме[I];

ГАС ␣ ⟨имя массива⟩ – из библиотеки удаляется массив с данным именем.

Следующие три приказа служат для заполнения поля. Каждый приказ вводит один массив с ИМЛ или перфоленты. Можно ввести в поле несколько массивов, которые размещаются один за другим в порядке поступления.

ВМЛМ ␣ ⟨имя массива⟩ – ввод массива с ИМЛ;

ВВС – ввод зоны перфоленты в посимвольном режиме;

ВЫХОД – последний выполняемый приказ сервиса.

5. Приказ СТОП; форма записи:

x СТОП x

Приказ служит для окончания работы с системой.
Система запускается директивой диспетчера ДД-3:

ВЗ ␣ МЛ ␣ {⟨знак⟩\x}$_3$ ␣ МА ␣ СМЛ ␣ =/

ЛИТЕРАТУРА

I. Управляющая система "Днепр-2" комплекса "Днепр-2I". Математическое обеспечение, ч.I. Киев, 1970. 243 с.

2. Управляющая система "Днепр-2I". Операционная система ДД-3. Описание и инструкция по пользованию. Киев, 1973. 132 с.

УДК 62-50:681.3.06

А.А.Бернуп-Бернхоф, К.А.Гужа

ОРГАНИЗАЦИЯ И СТРУКТУРА ПЕРИФЕРИЙНОГО ПРОЦЕССОРА НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ СИСТЕМЫ

Излагаются особенности организации периферийного процессора (ПП), работающего в мультипрограммном режиме обслуживания физических каналов научно-исследовательской системы. Приводятся данные, характеризующие процесс обмена информацией между ПП и центральной ЭЦВМ. Рассматриваются дальнейшие возможности внедрения ПП в измерительный центр, предусмотренный для автоматизации исследований физических свойств твердых тел.

В практике проектирования вычислительных систем (см. [1; с.336-342]), а также научно-исследовательских систем (НИС) для управления физическим экспериментом [2 и 3] весьма часто предусматривается совместное применение универсальной ЭЦВМ и процессоров специального назначения. Такое решение, конечно, позволяет лучше использовать вычислительные мощности ЭЦВМ для одновременного обслуживания нескольких пользователей. Следует также отметить, что, разрабатывая проект измерительного центра для исследования физических свойств твердых тел [4], еще перед выбором его основной ЭЦВМ целесообразно на основе небольшого процессора практически изучать особенности сбора измерительной информации и управления измерительно-воздействующей аппаратурой (ИВА) НИС. Чтобы обеспечить эти возможности в ЛГУ им.П.Стучки был разработан программируемый периферийный процессор "Физпульт", который имеет местную оперативную память для хранения программ и данных. Его система команд [5] была специализирована для работы с ИВА эксперимента.

Опыт четырехлетней эксплуатации ПП в условиях управления экспериментом позволил сделать ряд выводов как по

конструкции самого процессора, так и по программному обеспечению НИС. Настоящая работа посвящается обобщению полученной информации, а также изложению тех структурных особенностей, которые были обеспечены ПП в результате его реконструкции и подключения к центральной ЭЦВМ "Днепр-21".

Кроме вышеупомянутой цели познавательного характера, процессор проектировался с учетом следующих применений:

- управление ИВА эксперимента в реальном масштабе времени и прием результатов измерений на ПП;
- организация процесса обмена информацией между ИВА и ЭЦВМ, удаленной на расстояние около 100 метров от места реализации эксперимента;
- обеспечение обмена информацией между человеком и НИС.

Рассмотрим средства реализации вышеперечисленных функций в процессоре "Физпульт". В ходе разработки программного обеспечения НИС выяснилось, что модули ее программ достигают высокую степень автономности и универсальности, если они ориентированы на обслуживание конкретных физических каналов. Физическим каналом в настоящей работе названа группа агрегатов измерительно-воздействующей аппаратуры НИС, служащая для управления или измерения конкретного физического параметра объекта исследований.

Анализ алгоритмов физических методов исследований [6] показывает, что в большинстве случаев необходимо обеспечить параллельную работу нескольких физических каналов НИС и, кроме того, канала вывода визуальной графической информации для осведомления пользователя системы. Это означает, что ПП должен работать в режиме обслуживания нескольких параллельных процессов.

Возможности мультипрограммной работы ПП после его реконструкции были обеспечены программой ДИСПЕТЧЕР, возбуждаемой в следующих случаях:

I. Внутренний таймер прерывает программу ПП, что означает необходимость переключиться с режима обслуживания текущего физического канала на выполнение программы следующего по порядку канала НИС.

2. Таймер возбуждает систему прерывания программ ПП, указывая, что ожидание на начало сеанса обмена информацией с ЭЦВМ задерживается недопустимо долго.

3. Центральная ЭЦВМ вырабатывает сигнал прерывания программы ПП, запрашивая выводить результаты измерений для обработки в ее память.

4. Программа любого физического канала требует выполнить одну из следующих процедур:

- ввод текущего реализуемого программного модуля i-го физического канала в соответствующую область памяти ПП;
- сообщение в ЭЦВМ об аварийной ситуации программы i-го физического канала;
- присвоение идентификатора результату измерения и его запоминание в общем для всех каналов массиве данных, предусмотренном в ЭЦВМ для вывода;
- приостановка программы i-го канала до реализации события, помеченного в программе i-го канала к-им идентификатором;
- сообщение в ДИСПЕТЧЕР о реализации к-ого события в программе i-го канала.

Обмен информацией с центральной ЭЦВМ и выполнение некоторых вспомогательных операций программы ДИСПЕТЧЕР обеспечены при помощи устройства комплексирования процессоров, которым дополнен ПП. Это устройство позволяет выполнять следующие команды:

- запрет прерывания программы ПП;
- прерывание программы ЭУВМ с приоритетом i-го физического канала и ввод программы в память ПП по сигналу разрешения центральной машины;
- прерывание программы ЭЦВМ и ввод данных в ее память из ПП по сигналу разрешения машины;
- запрос прерывания программы ПП через заданное время Δt.

Кроме реакции на перечисленные команды, устройство комплексирования служит для возбуждения программы ДИСПЕТЧЕР по сигналам прерывания, поступающим с ЭЦВМ или тайме-

ра, находящегося в ПП.

Среди структурных особенностей ПП, управляющего в мультипрограммном режиме ИВА эксперимента, важно предусмотреть три разных режима счета времени:

- реализация задержки заданной величины между двумя операциями в программе любого канала НИС;
- выдача прерывающих сигналов через запрограммированные интервалы времени для переключения ПП на обслуживание следующего канала НИС;
- непрерывный счет времени с начала эксперимента и возможность программного опроса соответствующего счетчика в моментах сбора измерительной информации.

В связи с ограниченными возможностями системы прерывания программ в ПП "Физпульт" эти процессы были организованы при помощи трех одновременно работающих счетчиков.

При разработке программ, служащих для управления ИВА в реальном масштабе времени, существенно знать максимальные задержки, которые могут возникать между отдельными операциями этих программ в связи с работой ПП. Для программы конкретного канала такими в основном являются интервал времени, необходимый для обмена информации между ЭЦВМ и ПП и сумма временных квантов, необходимая для обслуживания остальных каналов НИС.

Величина этих задержек не должна понижать информационные возможности НИС, определяющиеся быстродействием ее ИВА. Поэтому устройство комплексирования ПП разработано с учетом требования, чтобы величина максимальной задержки не превышала времени одного измерения. Практически время реализации макрокоманды обмена информацией между ПП и ЭЦВМ определяется выражением:

$$T_1 = 10 + t_0 \cdot n,$$

где n – число пересылаемых ячеек памяти ПП, а $t_0 = 0{,}15$ мсек, а время прерывания программы канала с целью обслуживания остальных каналов НИС определяется

$$T_2 = (7 + S)_K,$$

где K – число каналов НИС, S – время обслуживания одного

канала (мсек).

Как структурную особенность ПП специально следует рассмотреть применение в составе ПП оперативной памяти. Если память используется в качестве запоминающего устройства программ, ее объем определяется возможностям достаточно быстрой координации работы ПП при помощи программ центральной ЭЦВМ. В этом случае реальны две альтернативы:

I) ПП обеспечивается памятью, в которой в начале эксперимента возможно записать все необходимые рабочие подпрограммы. Координация процессов ПП в этом случае осуществляется при помощи рабочих аргументов подпрограмм, которые поступают с ЭЦВМ через ее мультиплексорный канал. Этот же канал используется для ввода результатов измерений в центральную ЭЦВМ. Для экспериментов, соответствующих физическим методам исследований [6], в этом случае может потребоваться объем 3-4 К;

2) центральная ЭЦВМ высылает в память ПП только те подпрограммы, которые необходимы для реализации конкретного этапа эксперимента. Таким образом достаточно, если на программу каждого физического канала выделено около 200 ячеек памяти, объем всего запоминающего устройства ПП равен 0,5-I,5 К. ПП должен быть подключен к памяти ЭЦВМ через быстрый селекторный канал или канал непосредственного доступа.

С учетом того, что вторая альтернатива предусматривает меньше затрат на аппаратуру, она была применена при организации НИС на основе ПП "Физпульт".

Анализируя возможности построения ПП, следует рассмотреть еще одно конструктивное решение, согласно которому ПП разрабатывается без местного запоминающего устройства для хранения программ эксперимента [3]. Работа такого ПП осуществляется по командам, которые он из оперативной памяти ЭЦВМ вызывает самостоятельно через мультиплексорный канал. Этот же канал используется для записи результатов измерений. Такая конструкция ПП имеет ряд преимуществ. Общий объем аппаратуры ПП уменьшается приблизительно на половину. По предварительным оценкам экономия затрат на программирование

равна 20%. При необходимости ПП может быть обеспечен сверхбыстрой буферной памятью для накопления измерительной информации.

Из возможностей ПП "Физпульт", которые не могут быть обеспечены процессором, не имеющим местного запоминающего устройства, следует указать реализацию программ тестов ИВА без помощи ЭЦВМ, а также вывод графической информации на экран электронно-лучевой трубки для осведомления пользователя НИС. В то же время можно отметить, что ПП типа [3,5] только частично позволяют решить задачу связи человека с системой. Они могут быть использованы в качестве инженерного пульта управления каналов НИС в режиме их отладки и ремонта. ПП не предусмотрены для организации системы светового пера, а также обеспечения режима диалога оператора с НИС через телетайп или пишущую машинку. Эта задача решается непосредственно при помощи ЭЦВМ.

Оценивая результаты внедрения ПП "Физпульт" и разработки программного обеспечения эксперимента согласно методике, изложенной в настоящем сборнике (см.с. 12), отметим следующее.

Объем аппаратуры современных мини-ЭЦВМ превышает объем аппаратуры ПП. Это означает, что применение минимашин для непосредственного управления ИВА физического эксперимента типа [6] бесспорно оправдается, когда автоматизации подвергается одиночная НИС, а не ее комплекс. В настоящее время выяснена реальная возможность применения ПП в комплексе с ЭЦВМ для управления одиночным экспериментом физики твердого тела и одновременного решения нескольких фоновых задач вычислительного характера.

Более эффективное использование вычислительной техники измерительного центра может быть достигнуто путем применения ЭЦВМ в режиме одновременного обслуживания нескольких НИС, в которых установлены ПП типа [3,5]. Однако опыт применения двухпроцессорной структуры (ЭЦВМ и ПП) для автоматизации одиночной НИС, работающей по алгоритмам физических методов исследований [6], показывает, что ее действие связано с весьма сложными процессами обработки информации. Это

подтверждает необходимость на этапе разработки системы разделения времени ЭЦВМ пользоваться аппаратом вероятностного моделирования для выяснения временных характеристик, с которыми могут быть обслужены процессы НИС, имеющие разные приоритеты [7].

ЛИТЕРАТУРА

1. Дроздов Е.А., Пятибратов А.П. Основы построения и функционирования вычислительных систем. М., "Энергия", 1973. 112 с.

2. Виноградов В.И. Структура и организация дискретных информационных систем для автоматизации экспериментов. Л., препринт ФТИ-380, 1972. 38 с.

3. Бернуп-Бернхоф А.А. Выбор варианта связи ЭЦВМ с системами экспериментирования. - "Автоматика и вычислительная техника", 1973, №2, с.64-70.

4. Бернуп-Бернхоф А.А. Формализация цели проектирования научно-исследовательской системы. - Уч.зап.ЛГУ им.П.Стучки, т.160. Кибернетизация научного эксперимента, вып.3. Рига, 1972, с.37-56.

5. Бернуп-Бернхоф А.А. "Физпульт" - машина, управляющая научным экспериментом. - Кибернетизация научного эксперимента. Рига, 1968, с.181-201.

6. Тале И.А. Системный анализ и оценка целесообразности автоматизации спектрально-кинетических люминесцентных методов исследования твердого тела. - Уч.зап.ЛГУ им.П.Стучки, т.160. Кибернетизация научного эксперимента, вып.3. Рига, 1972, с.11-25.

7. Бернуп-Бернхоф А.А., Краузе-Крузе Х.Р., Гутанс Я.Я. Моделирование процессов обработки информации вычислительно-управляющего комплекса.-Уч.зап.ЛГУ им.П.Стучки, т.196. Кибернетизация научного эксперимента, вып.3. Рига, 1973, с. 50-69.

УДК 681.3.06:007.5

И.А.Гука, Г.Г.Бетун

КОНТРОЛЬ РАБОТОСПОСОБНОСТИ НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ СИСТЕМЫ

Рассматриваются организация периодического и профилактического контроля измерительно-воздействующей аппаратуры научно-исследовательской системы, осуществляемого при помощи управляющей вычислительной машины, а также принципы, положенные в основу осуществления такого контроля; приводятся примеры организации контроля работоспособности канала в целом и отдельных агрегатов.

В исследованиях физических свойств твердого тела применяются научно-исследовательские системы (НИС) [I]. Последние строятся на основе измерительно-воздействующей аппаратуры (ИВА), связанной с управляющей вычислительной машиной (УВМ).

Для успешного проведения физического эксперимента необходима подготовка НИС к эксперименту, которая включает организацию контроля УВМ и ИВА.

Рассмотрим организацию контроля ИВА, который осуществляется при помощи УВМ и состоит из:

- периодического контроля на основе контрольных и диагностических тестов,
- профилактического контроля.

Периодический контроль

Периодическим контролем можно убедиться в работоспособности ИВА и в случае неисправности получить информацию для ее локализации. Предъявим следующие требования к контролю:

- максимальное быстродействие процесса контроля,
- максимальная автоматизация операций контроля,
- минимальное привлечение дополнительных аппаратурных средств.

Контроль целесообразно производить по каналам НИС (каналом НИС будем называть группу агрегатов ИВА, служащую для измерения конкретного физического параметра исследуемого объекта или осуществления воздействия на объект).

Далее рассмотрим контроль научно-исследовательской системы, которая разработана в ЛГУ. Для осуществления контроля необходим измерительный прибор, позволяющий с достаточной точностью преобразовать аналоговые величины в цифровые коды. В качестве такого прибора используется имеющийся в системе аналого-цифровой преобразователь (АЦП); сначала он должен быть проверен. Затем можно начинать проверку отдельных каналов. Тест канала контролирует работоспособность канала и в случае аварии в нем выдает информацию, по которой с определенной вероятностью можно отыскать неисправный агрегат. С целью более точного определения места неисправности необходимо перейти к тесту детализированной проверки отдельного агрегата.

Тест канала помогает физику-экспериментатору оперативно контролировать ИВА непосредственно перед экспериментом. Процесс контроля в этом случае должен происходить с максимальным быстродействием и при высокой степени автоматизации.

Тест детализированной проверки отдельного агрегата нужен инженеру, обслуживающему НИС, и должен по возможности точнее локализовать неисправность, т.е.отыскать место неисправности с точностью до сменного блока или отдельного узла.

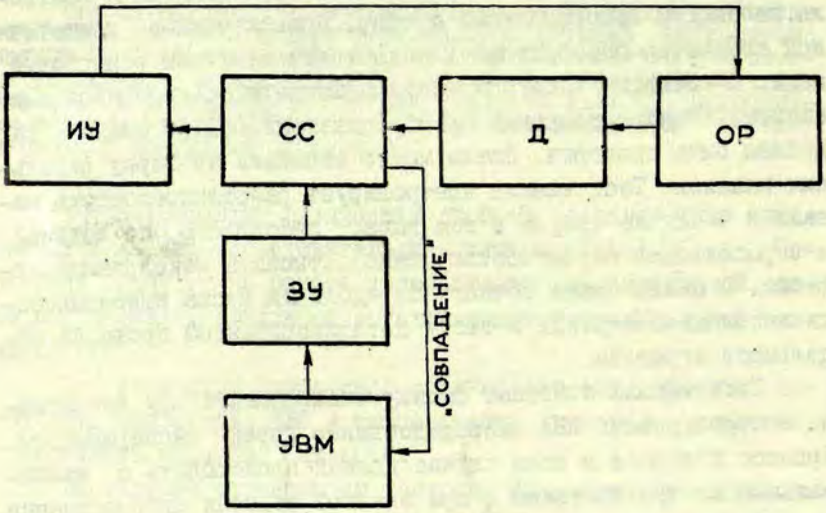
Таким образом можно выделить основные принципы организации контроля:

- специализация теста для проверки отдельного канала,
- применение детализированной проверки отдельных агрегатов в случае обнаружения неисправности в канале.

Какие особенности появляются при осуществлении этих принципов, покажет анализ примеров теста канала управления температурой и теста детализированной проверки автомата управления монохроматором.

Тест канала управления температурой

Рассмотрим функционирование канала управления температурой [2] (см.схему на рис.I).

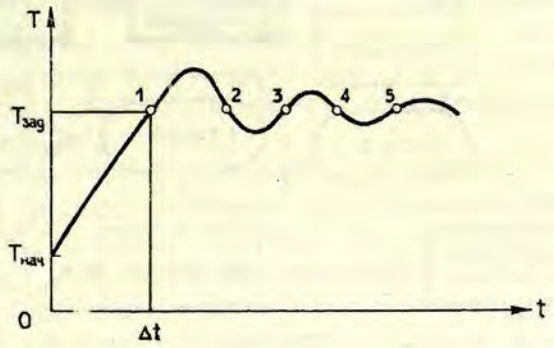


Р и с. I. Функциональная схема канала управления температурой.

ИУ – исполнительное устройство; СС – схема сравнения; Д – датчик; ОР – объект регулирования; ЗУ – задающее устройство.

Температура объекта регулирования (держатель кристалла) воспринимается датчиком. В задающем устройстве вырабатывается сигнал, соответствующий необходимому значению температуры объекта . Схема сравнения выполняет функцию сравнения показания датчика с сигналом задающего устройства и вырабатывает выходной сигнал, который подается на исполнительное устройство. Исполнительное устройство в зависимости от значения и величины сигнала схемы сравнения реализует приток или отток тепла от объекта.

После выдачи кода от УВМ на задающее устройство, соответствующего определенной температуре, последняя в объекте регулирования устанавливается по следующей зависимости (рис.2):

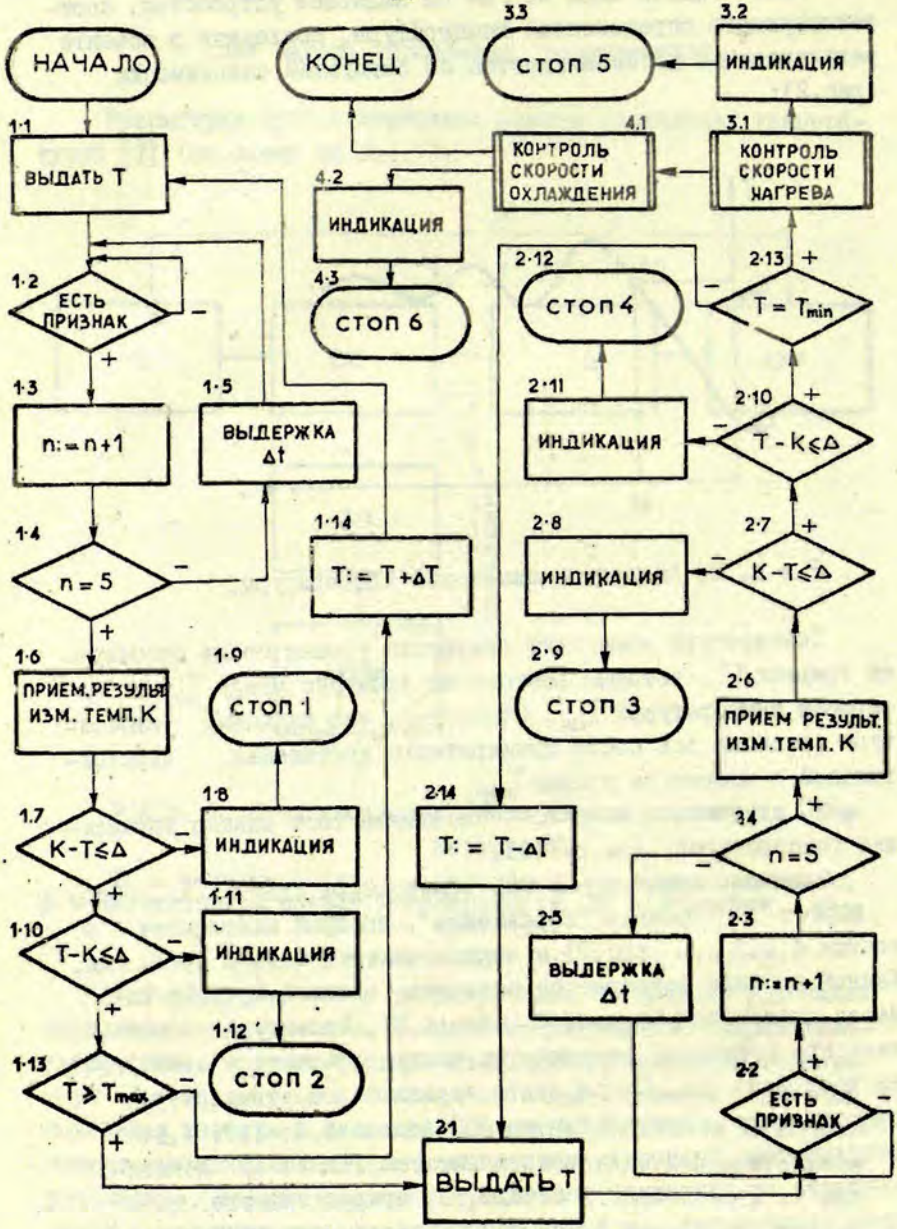


Р и с. 2. Диаграмма изменения температуры.

Температура достигает заданного уровня после промежутка времени Δt, который зависит от разности между $T_{зад}$ и начальной температурой $T_{нач}$. Считается, что заданная температура установилась после пятикратного достижения действительной температуры уровня $T_{зад}$.

С учетом сказанного составлен тест канала управления температурой (см.рис.3).

Задавая определенный код температуры (блок I.I), проверяем признак "совпадение", который появляется в точках I,2,3 ... (рис.3) и вырабатывается схемой сравнения. Данный признак проверяется пять раз (блоки I.2; I.3; I.4) через промежуток времени τ (блок I.5). Последнее обстоятельство исключает возможность чтения признака в одной точке несколько раз. После этого заданный код температуры сравнивается с полученным на АЦП (блоки I.6; I.7; I.IO) результатом. Разность, которая получается из-за неидентичности градуировки рабочего и измерительного датчика (термопары), а также всякого рода случайные помехи, не оказывающие существенного влияния на работу



Р и с. 3. Алгоритм теста канала управления температурой.

канала управления температурой, учитываются при помощи допуска Δ . Далее по программе температура увеличивается с шагом ΔT до значения T_{max} (блоки I.I3; I.I4).

Аналогично работает тест в режиме охлаждения.

Несоответствие заданного кода температуры результату измерения свидетельствует о неисправностях в температурном канале; в этом случае тест останавливается (блоки I.9; I.I2). УВМ предварительно сообщает причину останова теста (блоки I.8; I.II).

Кроме точности установки температуры, необходимо проверить скорость установки заданной температуры (блоки 3.4, ..., 4.I). Для этой цели при температуре объекта, равной T_{min} , УВМ выдает код температуры T_{max} и через промежуток времени Δt , после которого должна установиться заданная температура, принимается код с АЦП и сравнивается с заданным.

При недопустимом расхождении установившейся температуры с величиной T_{max} УВМ индицирует код действительной температуры образца и тест останавливается (блок 4.2).

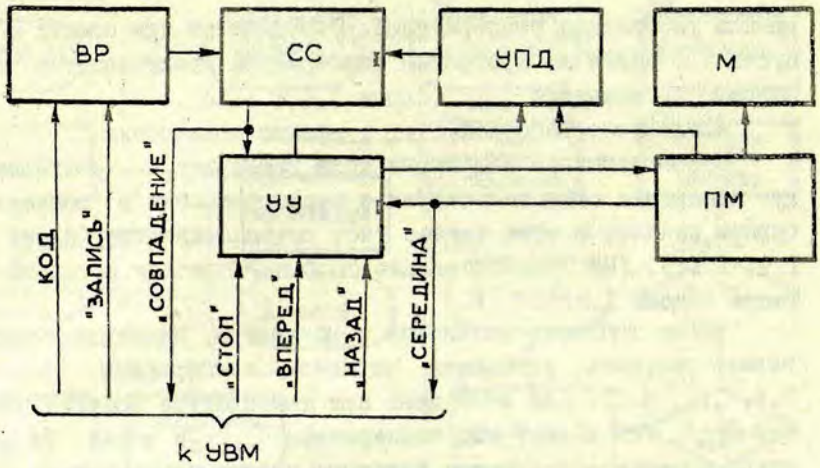
Таким же образом проверяется скорость охлаждения. Данный тест является типичным примером организации контроля работоспособности канала в целом.

Тест проверки автомата управления монохроматором

Функционирование автомата управления монохроматором (АУМ) можно представить по упрощенной схеме, представленной на рис.4 [3].

От УВМ на входной регистр подается код, соответствующий определенному положению лимба монохроматора. Импульсом "запись" данный код записывается на входной регистр, импульсом "вперед" ("назад") происходит запуск привода монохроматора. После установки монохроматора в заданное положение схема сравнения вырабатывает признак "совпадение". Импульс "стоп" служит для остановки двигателя. Признак "середина" вырабатывается специальной схемой при среднем положении лимба монохроматора.

В монохроматоре предусмотрен реверс в крайних положениях лимба.



Р и с. 4. Функциональная схема автомата управления монохроматором.

ВР – входной регистр; СС – схема сравнения; УПД – указатель положения двигателя; М – монохроматор; ПМ – привод монохроматора; УУ – узел управления.

Следовательно, тест проверки АУМ должен состоять из следующих частей:

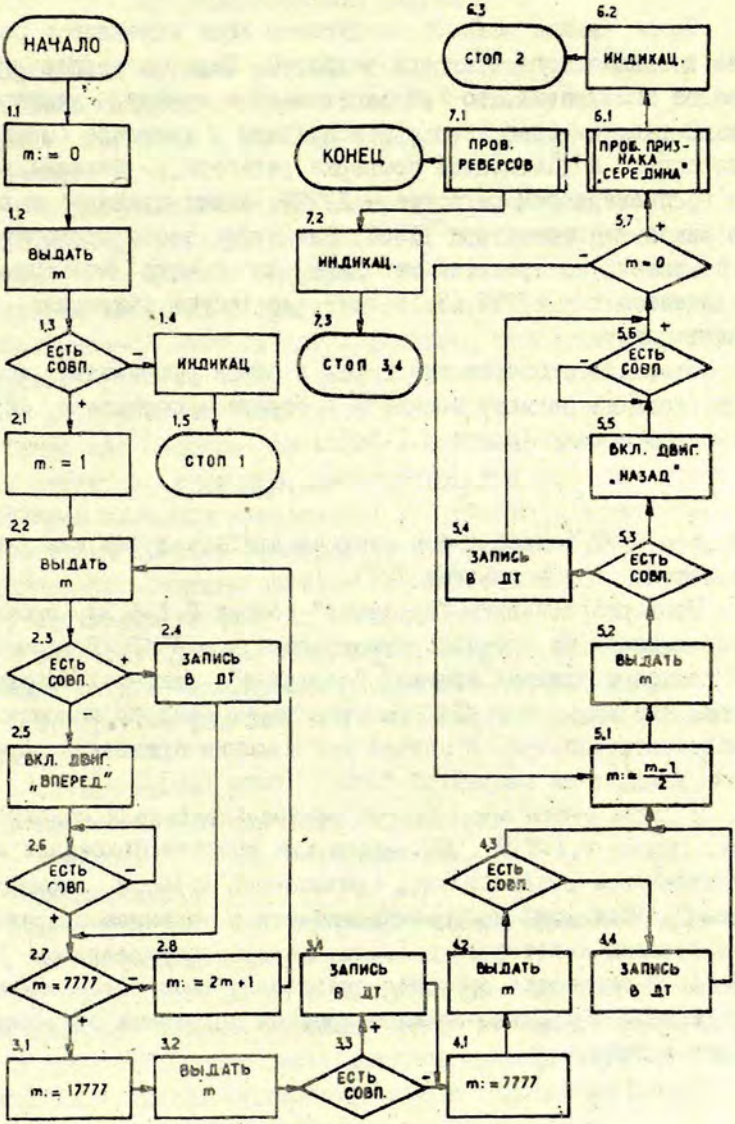
1) проверка схемы сравнения;
2) проверка входного регистра;
3) проверка признака "середина";
4) проверка реверсов двигателя.

Монохроматор уставляется в начальное положение. Затем тестом (рис.5) проверяется схема сравнения, для чего на регистр монохроматора выдается код "0" и без запуска привода опрашивается признак "совпадение" (блоки I.I ÷I.2). Признак должен быть, так как данный код соответствует начальному положению.

В случае отсутствия признака программа индицирует сообщение о неисправности и выходит на аварийный стоп (блоки I.4; I.5).

Проверяемый регистр содержит I3 разрядов. На него для проверки выдается серия кодов m . С целью точного определения неисправного разряда коды формируются УВМ по закону:

$$m_{n+1} = 2m_n + 1; \quad n = 0, 1, 2, \ldots$$



Р и с. 5. Алгоритм теста проверки АУМ.
ДТ – диагностическая таблица.

После выдачи каждого следующего кода изменяется состояние только одного триггера регистра. Если состояние триггера не изменяется, то УпМ записывает в ячейку диагностической таблицы сообщение о неисправном триггере (блок 2.4) и переходит к дальнейшей проверке регистра. Максимальный код на проверяемом регистре – I7777, а код крайнего положения лимба монохроматора II300. Для того, чтобы проверить I3-й разряд по принятой методике, на регистр монохроматора выдается код I7777 (3.I; 3.2), но запуск двигателя не производится.

Затем для проверки триггеров в обоих устойчивых состояниях коды на регистр выдаются в обратном порядке и формируются по закону (блоки 4.I–5.8):

$$m_{n+1} = \frac{m_n - 1}{2}\ ; \qquad n = 1, 2, 3, \ldots$$

Код m = 7777 выдается без запуска двигателя, так как лимб уже находится в положении 7777.

Проверка признака "середина" (блоки 6.I–6.3) производится выдачей на входной регистр кода m = 4540. В этом случае должен появиться признак "середина", свидетельствующий о том, что коду m = 4540 соответствует среднее положение лимба монохроматора. В случае непоявления признака программа выходит на аварийный "стоп" (блок 6.3).

В конце теста проверяются реверсы двигателя монохроматора. (блоки 7.I–7.3). Для этого при среднем положении лимба монохроматора выдается код, превышающий 4540, и команда "назад". Если лимб не устанавливается в заданном положении по истечении некоторого времени, которое определяется минимальной скоростью вращения двигателя, тест останавливается.

Реверс в противоположном крайнем положении проверяется аналогично.

Профилактический контроль

Выше была описана система контроля измерительных и воздействующих каналов с целью проверки правильности работы отдельных агрегатов и каналов в целом, а также локализации неисправного узла.

Цель профилактического контроля - выявление узлов, в которых имеются ненадежные элементы. Последнее достигается специально заданными режимами. Тяжелые условия создаются: отклонением питающих напряжений от номинальных значений; изменением рабочей частоты или повышением температуры окружающей среды. При этом выбор способа зависит от конкретного типа схем и устройств. Различного рода комбинационные устройства, такие как цифровые автоматы и их компоненты (логические схемы, регистры, счетчики, дешифраторы и т.д.) проверяются отклонением питающих напряжений; устройства с временным и температурным дрейфом проверяются повышением температуры; программно управляемые оптико-механические элементы проверяются увеличением частоты их коммутации.

В качестве примера рассмотрим контроль программно управляемых оптико-механических элементов каналов [4].

К программно управляемым оптико-механическим элементам относятся:

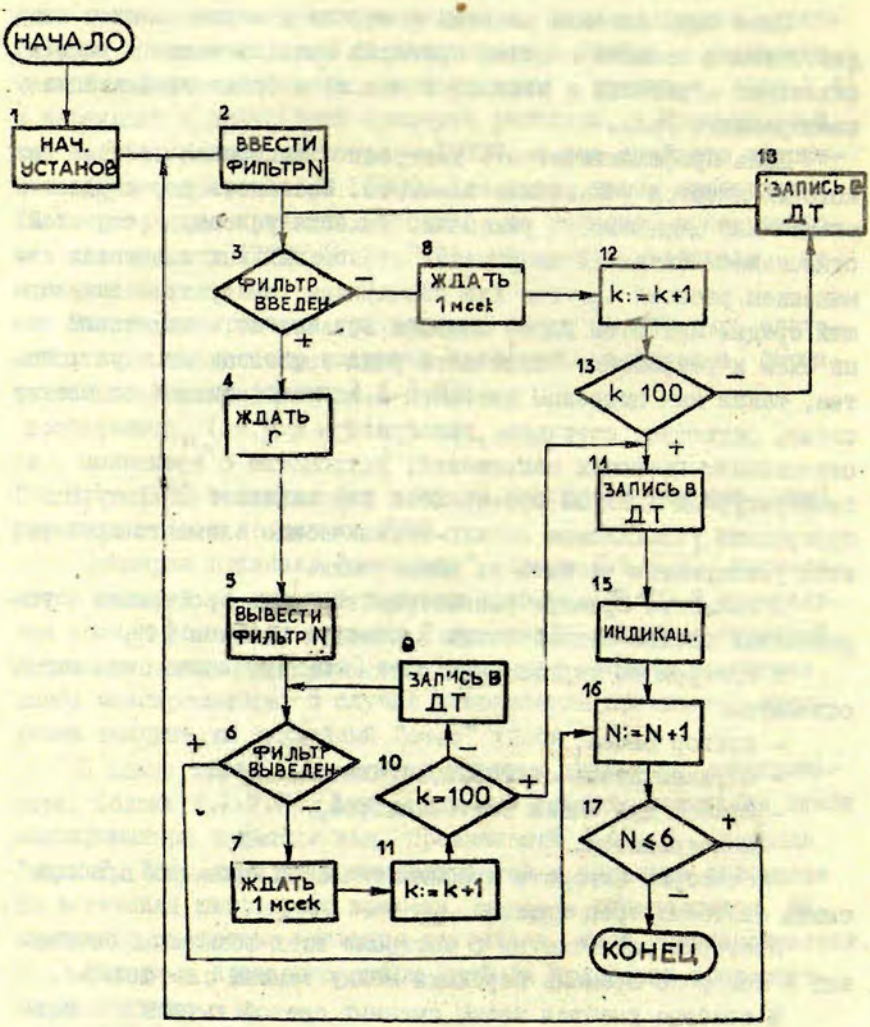
- затвор света,
- переключатель светового пучка,
- прибор для смены светофильтров,
- монохроматор.

Рассмотрим алгоритм профилактической проверки прибора смены светофильтров (рис.6).

Проверка заключается в контроле всех возможных состояний и контроле времени перехода между такими состояниями.

В приборе имеется шесть сменных светофильтров с контактной индикацией, сигнализирующей о том, какие фильтры находятся в оптическом канале.

Согласно алгоритму, попеременно вводятся и выводятся все шесть фильтров. Время переходного режима фиксируется и записывается в диагностическую таблицу (блоки 3-8-12-13-18-



Р и с. 6. Алгоритм профилактической проверки прибора смены светофильтров.
ДТ – диагностическая таблица.

-3) или (блоки 6-7-II-I0-9-6). В соответствующем введенному (выведенному) светофильтру адресе диагностической таблицы остается последнее показание счетчика (блок I2 или II). Если признак конечного положения фильтра не появляется недопустимо долго, т.е. если K - число в счетчике (блок I2 или II) - больше заданного числа A (A = I00), сообщение о неисправности выводится на регистр индикации УВМ (блок I5) и записывается в диагностическую таблицу. УВМ переходит к проверке остальных фильтров. После проверки всех фильтров программа останавливается.

Л И Т Е Р А Т У Р А

I. Кибернетизация научного эксперимента, вып.I. Рига, I968.

Глава 2. Научно-исследовательская система НИКС-В, с.53-76.

2. Зариньш М.Я., Плаудис А.Э. Терморегулятор малоинерционного азотного криостата. - Уч.зап.ЛГУ им.П.Стучки, т.I70. Кибернетизация научного эксперимента, вып.4. Рига, I974, с. I56-I77.

3. Янсонс Я.Л., Страумен Я.Я., Тарденак Э.Э. Преобразователь код-угол для управления монохроматором. - Уч.зап.ЛГУ им.П.Стучки, т.I70. Кибернетизация научного эксперимента, вып.4. Рига, I972, с.20I-2I5.

4. Страумен Я.Я., Пранч Е.А., Аберс А.Я., Лиепа Г.М., Тройцис А.Л. Элементы оптико-механических каналов. - Уч. зап. ЛГУ им.П.Стучки, т.I70. Кибернетизация научного эксперимента, вып.4. Рига, I972, с.I78-200.

УДК 681.3.01:007.5

Ю.Я.Кузьмин, И.А.Кельман, С.В.Гвоздев

МИНИМИЗАЦИЯ ВРЕМЕННЫХ ПОТЕРЬ В УНИВЕРСАЛЬНОМ ПРОГРАММНОМ ОБЕСПЕЧЕНИИ ЭКСПЕРИМЕНТА

Предлагается ряд альтернатив для минимизации временных потерь, возникающих при использовании универсальных программных модулей эксперимента с циклическим характером обслуживания процессов.

Имеется несколько причин временных потерь в универсальном программном обеспечении эксперимента (УПОЭ). Основная причина состоит в том, что универсальность модуля порождает "лишние" команды по отношению к обслуживаемым процессам системы [*].

Однако эти "лишние" команды позволяют применять модули в различных ситуациях и для разного числа процессов, т.е. именно они и делают модуль универсальным.

Другая причина состоит в том, что не учитывается готовность обслуживаемых процессов, в результате чего процесс обслуживается "вхолостую". Однако эта причина не является принципиальной и может быть легко устранена, например, механизмом селекции процессов.

Третья причина связана с запаздыванием обслуживания процессов из-за медленнодействия устройств.

Четвертая причина заключается в том, что не учитывается приоритетность отдельных процессов. Этот вопрос также разрешим указанными выше средствами.

Следующая группа причин временных потерь в УПОЭ аналогична упомянутым, но относится уже к комплексу модулей.

В случае научно-исследовательской системы "GUNDEGA-2" [1,2 [**]] комплекс модулей был объединен в жесткую последова-

---

[*] Под обслуживаемым процессом будем понимать сбор, управление, контроль научно-исследовательской системы, вычисление в ходе эксперимента ряда параметров, связанных с обработкой измерительных результатов, связь оператора с этой системой и т.п.

[**] См. настоящий сборник, с. III.

тельность, так называемое кольцо групповых программ. Это непринципиально, если цикл эксперимента, зависящий от переходных процессов в объекте исследования и аппаратуры, больше, чем время обхода кольца. В противном случае такое УПОЭ становится непригодным. Для УПОЭ – "GUNDEGA-2" цикл составляет I сек, но может быть сокращен до 400 мсек за счет уменьшения числа усреднений параметров.

Рассмотрим основные варианты минимизации временных потерь в УПОЭ.

I. Простейший способ минимизации временных потерь состоит в переходе на более быстродействующую технику. Например, с переходом от ЭВУМ "Днепр-I" на мини-ЭВМ "М-6000" ожидается уменьшение временных потерь более чем в 5 раз.

Эффективным методом ускорения работы УПОЭ является использование технических возможностей современных ЭВУМ, таких как канал прямого доступа в память, системы прерывания.

2. Другой путь состоит в повышении качества универсальных модулей, т.е. в уменьшении количества команд и применении различных программистких приемов, ускоряющих работу модуля.

3. Действенный способ уменьшения временных потерь внутри модуля состоит в селекции обслуживаемых процессов или вычисляемых параметров в соответствии с некоторыми условиями, определяющими необходимость обслуживания процесса в данный момент.

При обсуждении различных механизмов селекции процессов или параметров будем пользоваться термином маска . Напомним, что процесс называется замаскированным (маска равна I), если он не подлежит обслуживанию.

Теперь рассмотрим некоторые варианты селекции процессов.

I. Использование системных часов. Пусть i-й процесс описывается функцией

$$F_i(M_i, \{N\}_i, t_i, \Delta t_i, x_i),$$

где M_i – маска,

$\{N\}_i$ – группа идентификаторов процессов, связанных с i-процессом,

t_i – время предыдущего обслуживания,

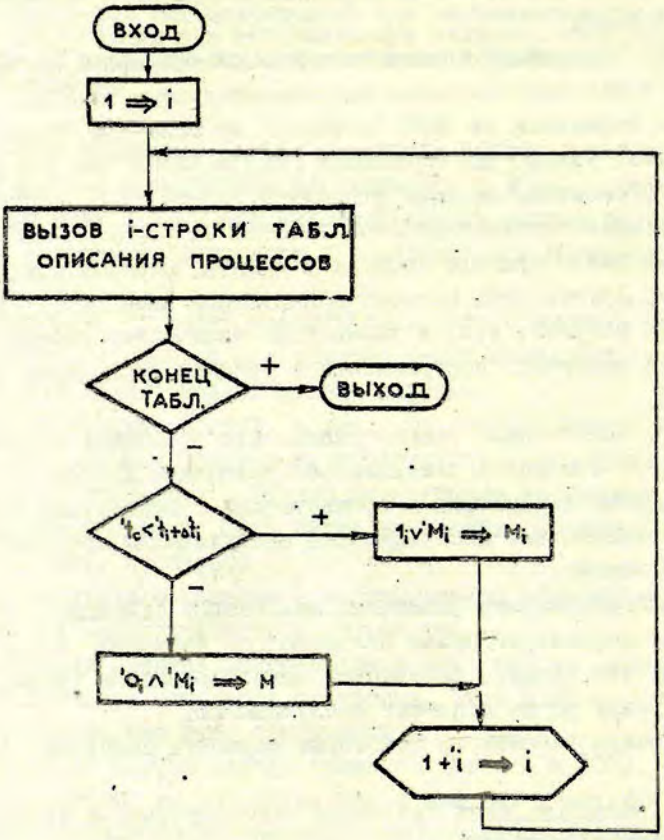
Δt_i – период обслуживания,

x_i – значение параметра процесса.

Процесс маскируется, если $t_c < t_i + \Delta t_i$,

где t_c - системное время в момент вычисления маски (см.рис. I).

Естественно, процессы с большим приоритетом обладают меньшим Δt.



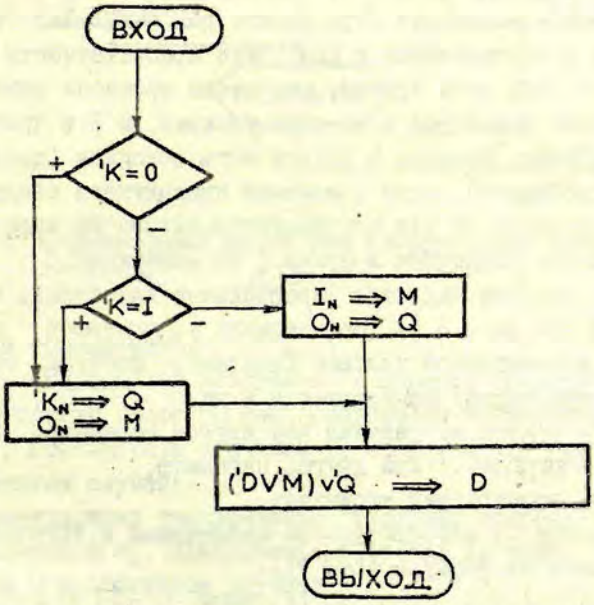
Р и с. I. Селекция процессов при использовании системных часов t_c.

2. Маскирование в режиме диалога. Использование диалога дает возможность принимать решение о необходимости обслуживания процесса в ходе проведения эксперимента, поэто-

му такой способ особенно удобен при осуществлении поискового эксперимента, а также при отладке программ эксперимента и отдельных каналов научно-исследовательской системы (НИС). Кроме того, этот метод позволяет значительно упростить программное обеспечение и сократить требуемую память ЭВМ, так как исключает необходимость хранения и обработки условий, определяющих моменты обслуживания процессов, что, как показывает опыт, существенно превышает затраты на реализацию режима диалога.

Подобный способ селекции процессов применялся в научно-исследовательской системе "GUNDEGA-2" для каналов управления; он может быть обобщен на все виды каналов.

На рис.2 приведена блок-схема вычисления маски и управляющего кода канала системы "GUNDEGA-2".



Р и с. 2. Механизм маскирования процесса каналов (управления - D).

Q - регистр диалогового управления каналами, M - маска, K_N - параметр для N - канала, Γ_N, O_N - включить и выключить N -канал, соответственно.

При поступлении от оператора сообщения вида "ВЫД N K+", в регистр оператора Q заносится управляющий код K . В процедуре принимают участие еще два регистра – регистр масок M и регистр выходного управляющего кода D .

Если значение M устанавливается равным I, то процесс считается закрытым для управления оператором и должен получать код управления программно.

3. Остановимся более подробно на методе селекции процессов с помощью логических условий, записанных в виде дизъюнктивных нормальных форм (ДНФ). Обстоятельный разбор именно этого способа объясняется тем, что, на наш взгляд, аппарат ДНФ достаточно универсален и может применяться для описания сложных условий функционирования элементов НИС и самого программного обеспечения.

Как указывалось ранее, селекция процессов используется в основном для решения двух задач: для выявления готовности процесса к обслуживанию и для учета приоритетности отдельных процессов, при этом примем, что маска процесса равна 0, если выполняются некоторые известные условия, и I в противном случае. Например, процесс N должен быть обслужен (должно быть выдано сообщение), если с момента предыдущего обслуживания прошло не менее 60 сек или поступил запрос от экспериментатора на выдачу сообщения и процесс не возбужден.

Эти условия являются управляющими для блока, вычисляющего маску (см.рис. 3,4). При задании управляющей информации каждое элементарное условие (например, поступил запрос на выдачу информации) записывается в виде $x_i - L_i Y_i$, где:

x_i – показание датчика или другой параметр;

Y_i – уставка или другой параметр;

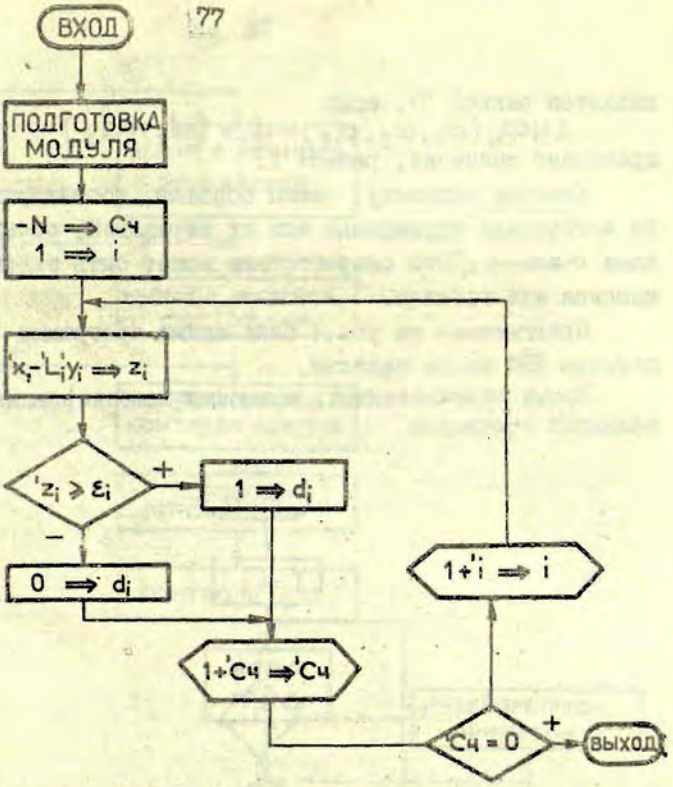
L_i – нормирующий множитель.

Переход от действительных переменных к логическим осуществляется по формуле (см.[2]):

$$d_i(x_i, Y_i, L_i, \varepsilon_i) = \begin{cases} I, & \text{если } x_i - L_i Y_i \geqslant \varepsilon_i, \\ 0, & \text{если } x_i - L_i Y_i < \varepsilon_i. \end{cases}$$

здесь ε_i – допустимая величина отклонения (см.рис.3).

Для задания сложных условий связи между логическими переменными записываются в виде ДНФ по следующему образцу.



Р и с. 3. Вычисление множества элементарных высказываний $\{d_i\}$.

Введем обозначения:

– элементарному высказыванию "с момента предыдущего обслуживания прошло более 60 сек" сопоставим логическую переменную d_1, принимающую значение I, если условие выполнено, и 0 в противном случае;

– элементарному высказыванию "поступил запрос" – логическую переменную d_2, принимающую значение I, если запрос поступил, и 0 в противном случае;

– элементарному высказыванию "процесс возбужден" – логическую переменную d_3, которая равна I, если процесс возбужден, и d_3, равную 0, в противном случае.

Тогда процесс подлежит обслуживанию (маска устанав-

ливается равной 0), если

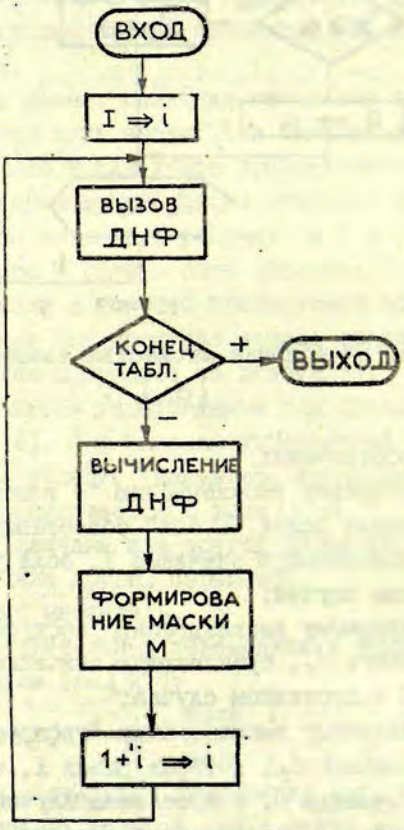
$$ДНФ_N(d_1, d_2, d_3) = d_1 \vee (d_2 \wedge \bar{d}_3)$$

принимает значение, равное I.

Каждому процессу, таким образом, соответствует несколько логических переменных или их отрицаний, соединенных знаками $\vee$ или $\wedge$. Это соответствие может быть задано в виде массива или таблицы.

Приведенная на рис.4 блок-схема программы вычисляет с помощью ДНФ маски каналов.

Кроме перечисленных, возможны комбинированные методы селекции процессов.



Р и с. 4. Селекция каналов при использовании ДНФ, описывающих условия обслуживания каналов.

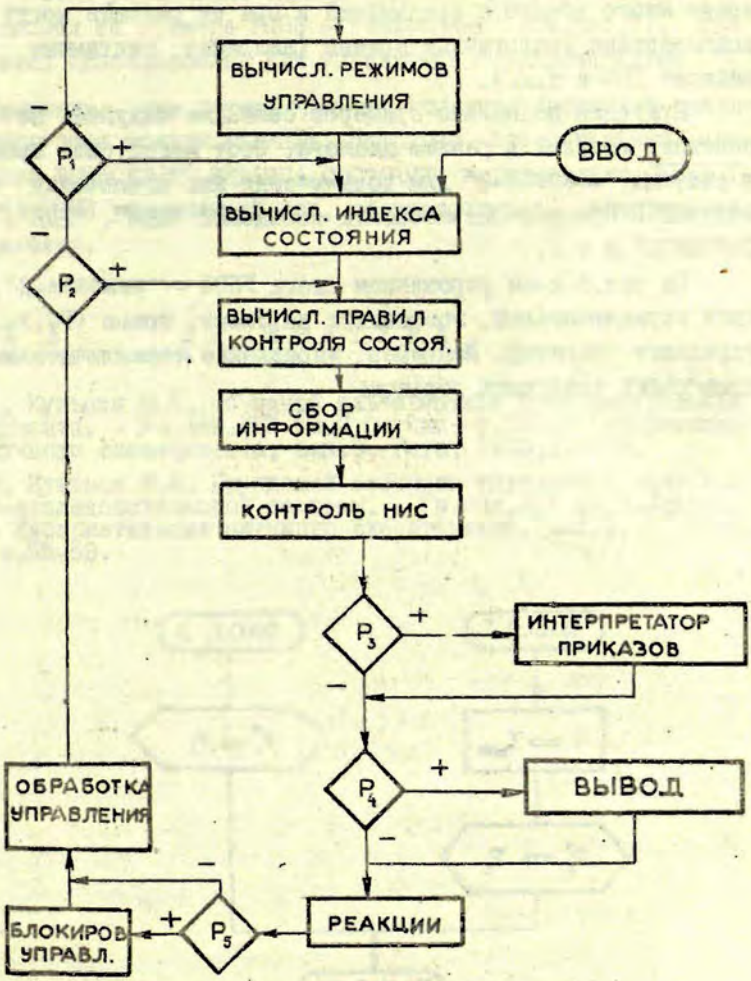


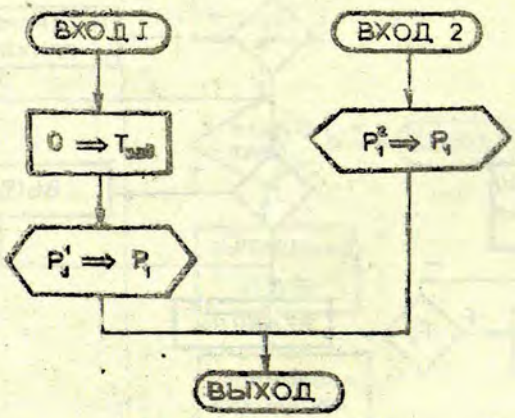
Рис. 5. УПОЭ – "GURDEGA-2".

Селекция модулей: P_I – НДИ (–), ИДИ (+); P_2 – есть аварии (+), нет аварий (–); P_3 – есть приказ оператора (+), нет приказа (–); P_4 – требуется вывод информации (+), не требуется (–); P_5 – есть аварии (+); нет аварий (–).

Далее рассмотрим варианты селекции модулей. Эта задача имеет много общего с предыдущей и при ее решении могут быть использованы аналогичные приемы (например, системные часы, аппарат ДНФ и т.д.).

Приведем несколько примеров селекции модулей. Начнем с селекции модулей в режиме диалога. Этот метод был применен в системе "GUNDEGA-2" для подключения или отключения ряда модулей с помощью операторских сообщений "ЖДИ", "ИДИ", "ОСТАНОВИСЬ" и т.д.

На рис.5 дана упрощенная схема УПОЭ - "GUNDEGA-2". Из пяти переключателей, показанных на схеме, тремя (P_I,P_3,P_4) управляет оператор. Например, управление переключателем P_I происходит следующим образом:



Р и с. 6. Управление переключателем P_I в УПОЭ - "GUNDEGA-2".

ВХОД I - режим ЖДИ, ВХОД 2 - режим ИДИ. P_I, P_2 - переходы на модуль ВЫЧИСЛЕНИЕ РЕЖИМОВ УПРАВЛЕНИЯ и модуль ВЫЧИСЛЕНИЕ ИНДЕКСА СОСТОЯНИЯ, соответственно

Метод управления одними модулями другими может быть тоже пояснен на примере УПОЭ – "GUNDEGA-2" (см.рис.5), где этот метод использовался для управления переключателями P_2 и P_5. Положения этих переключателей являются функциями выходной информации модуля РЕАКЦИЯ. Например, при серьезных авариях блок ВЫЧИСЛЕНИЯ ИНДЕКСА СОСТОЯНИЯ выдал модулю ОТРАБОТКА УПРАВЛЕНИЯ управляющий код, соответствующий прекращению эксперимента.

ЛИТЕРАТУРА

1. Кузьмин Ю.Я. Об одной альтернативе программирования эксперимента. – Уч.зап.ЛГУ им.П.Стучки, т.196. Кибернетизация научного эксперимента, вып.5. Рига, 1973, с.3-16.

2. Кузьмин Ю.Я. Групповой алгоритм управления каналами научно-исследовательской системы. – Уч.зап.ЛГУ им.П.Стучки, т.196. Кибернетизация научного эксперимента, вып.5. Рига, 1973, с.55-59.

УДК 681.3.01:007.5

С.В.Гвоздев

ВЫВОД ИНФОРМАЦИИ В УНИВЕРСАЛЬНОМ ПРОГРАММНОМ ОБЕСПЕЧЕНИИ ЭКСПЕРИМЕНТА

Обсуждается набор программных модулей, позволяющий решать задачи, возникающие при выводе информации в автоматизированном эксперименте.

Применение УВМ к задачам сбора измерительной информации и прямого цифрового управления предполагает переработку значительного количества информации. Лишь небольшая часть этой информации может быть представлена с помощью регистрирующей аппаратуры (табло, индикаторы, графопостроители). В то же время обслуживающий персонал в некоторых ситуациях должен иметь в распоряжении дополнительную информацию, как для возможности принятия оптимального решения в альтернативных ситуациях, так и для идентификации аварийной ситуации и локализации сбоев. Очевидно, предоставление в распоряжение оператора всей информации, имеющейся в памяти УВМ, не является наилучшим решением задачи, так как ручная обработка ее весьма трудоемкий процесс. Программное обеспечение эксперимента, следовательно, должно включать в себя некоторые модули, которые позволяют проводить селекцию информации и коммутацию внешних устройств вывода, а также наблюдение за интересующими параметрами во времени. Значение подобного рода программ возрастает в случае, когда эксперимент обслуживается в режиме диалога "УВМ – человек".

Подобного рода идеи могут быть реализованы, естественно, многими способами. Это зависит от различных факторов и в первую очередь от:

- архитектуры построения программного обеспечения;
- типа применяемой УВМ;
- сложности поставленной проблемы.

Надо сразу отметить, что мы исходим из предположения, что программное обеспечение построено на основе комплекса универсальных групповых программ [I]. Наиболее последовательно и полно этот принцип применен при построении системы MADAM [2]. В рамках этого подхода решение задачи вывода информации сводится весьма естественно к созданию некоторых универсальных групповых программных модулей, выполняющих определенные специализированные функции. В работах [3, 4] имеются описания тех функций, которые выполняют программные модули, решающие задачу селекции информации. Рассмотрим подробнее модуль ВЫВОД в программном обеспечении научно-исследовательской системы "GUNDEGA-2" *. Система имеет три канала вывода: одно печатающее устройство, два графопостроителя. При необходимости имеется возможность получить дополнительно до пяти каналов вывода информации на перфоленту. Чтобы большая часть переменной информации была доступна для вывода, все величины как принятые с датчиков и обработанные, так и вычисленные, располагаются в общем поле результатов в едином формате. Это позволяет упростить программу подготовки данных для вывода, избегая переводов из одного формата в другой.

Рассмотрим организацию вывода информации на печать. Запрос на печать информатор вводит в УВМ через клавиатуру электропишущей машинки (ЭПМ), используя четыре типа приказов:

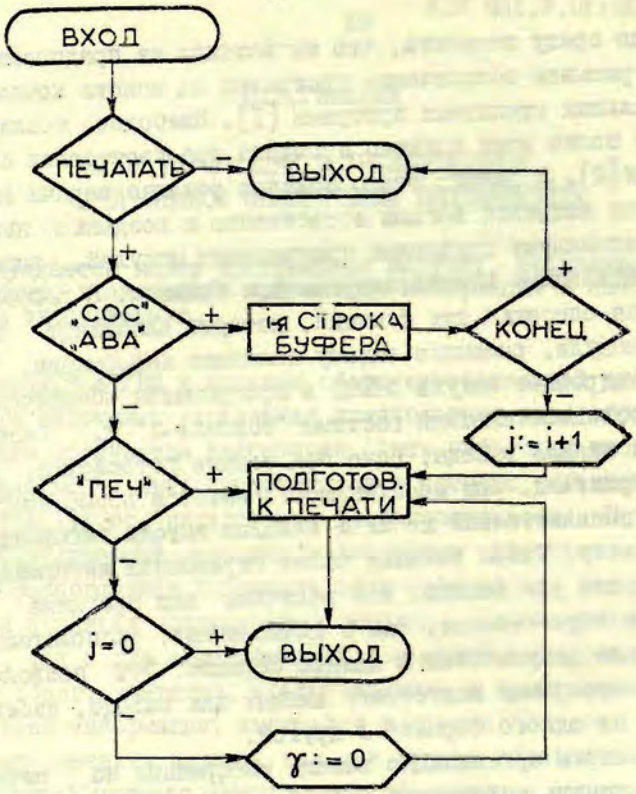
I. ПЕЧАТАЙ № +
2. ОПРОСИ № +
3. СОСТОЯНИЕ +
4. АВАРИИ +

Интерпретатор приказов сообщает программе подготовки данных информацию о виде вывода, которая в зависимости от типа сообщения производит поиск и подготовку данных. На рис.I приводится блок-схема модуля подготовки информации.

На схеме видно, что программа выбирает данные из буфера и стандартным образом подготавливает их для модуля ПЕЧАТЬ. Заполнение буферов происходит после интерпретации

---

* См.настоящий сборник, с.III.



Р и с. Блок-схема модуля подготовки данных для модуля ПЕЧАТЬ.

операторского приказа.

Программа ПЕЧАТЬ является по сути дела управляющей программой для ЭПМ, которая работает в реальном масштабе времени с УВМ.

Отрицательной чертой организации вывода в научно-исследовательской системе "GUNDEGA-2" является отсутствие возможностей для указания условий, в зависимости от которых производится выдача информации того или иного типа.

В системе MADAM эти трудности решаются использованием их программных модулей как

- циклическая выдача результата (ZYMP),
- анализ сбоя (STAP),
- логическая связь результатов (DTAB),
- выдача текстов (MWDR).

Остановимся вкратце на этих модулях. Программа циклической выдачи результатов инициируется приказом оператора. Для большей информативности выводимых данных имеется возможность объединять несколько параметров в одну группу, которая одновременно выводится на печать через заданный промежуток времени. Всего можно задать несколько подобных групп.

Таким образом оператор может проследить во времени состояние целиком некоторого функционального узла или взаимодействие нескольких элементов системы.

Программа анализа сбоев возбуждается при обнаружении серьезной аварийной ситуации. Сигнал аварии поступает из подпрограммы логической связи результатов. Входными величинами для этой программы являются состояния дискретных датчиков или значения логических проверок на поле аналоговых сигналов. Например, каждый аналоговый сигнал при нормальном функционировании системы находится в некотором допустимом интервале. Соответствующее логическое высказывание равно "да" или "нет" в зависимости от величины аналогового сигнала. Программа логической связи проверяет наличие в системе нежелательных отклонений и соответствующим образом реагирует. В частности, при неопасных или несистематических сбоях результаты анализа передаются программе выдаче текстов, которая информирует оператора о замеченных неисправностях. Следовательно, программа логической связи результатов выполняет диагностирующие функции. Необходимо отметить, что логические проверки аналоговых сигналов могут быть гораздо сложнее описанного случая, что допускает сообщения о состоянии системы во взаимосвязи со многими сигналами.

Если сбой, обнаруженный программой логической связи, оказался достаточно серьезным, то программа анализа сбоя подготавливает для выдачи информацию в протокол сбоя (НЕКРОЛОГ).

Для определения сбоя следует выяснить, как ведут себя в некотором промежутке времени определенные величины, относящиеся к данному устройству. Чтобы избежать трудоемкого разбора

протокола сбоя, все величины, важные для определения сбоя, необходимо отобразить по возможности нагляднее. В системе MADAM аналоговые и дискретные сигналы четко различаются. Это сделано, по-видимому, в целях экономии оперативной памяти, так как аналоговый сигнал занимает 24 разряда, а дискретный только один разряд. При разработке программного обеспечения системы "GUNDEGA-2" для увеличения надежности работы с дискретными каналами и для простоты доступа к информации было решено каждый дискретный сигнал отображать в одно слово как и аналоговый.

Протокол сбоя охватывает некоторый промежуток времени до и после сбоя (пред- и послеистория). Он содержит изменение во времени необходимых аналоговых сигналов и изменение дискретных сигналов, которые произошли в этом промежутке времени, а также состояние (фотографию) дискретных сигналов к началу выдачи протокола.

Рассмотренные программы обслуживает единая для всех программа выдачи текстов, которая сортирует информацию, поступающую на вывод по различным выводным каналам. Кроме того, программа осуществляет подготовку текстов для пояснения выводимой информации.

Если, кроме всего прочего, учесть еще и различные количественные характеристики рассмотренных программ (см.[4]), то станет ясным что система MADAM обладает мощным и развитым инструментом для наблюдения за процессом. В данной статье на основе опыта существующих разработок попытаемся выяснить состав и структуру программных модулей, обеспечивающих вывод информации.

Как уже отмечалось, мы будем исходить из предположения, что вся интересующая нас информация представлена в едином формате и расположена в общем поле результатов.Предположим, что с каждой единицей информации связано некоторое однозначно заданное число, которое мы будем называть номером канала. Следовательно, запрос на вывод осуществляется указанием номеров соответствующих каналов.

Для простоты выберем следующие модули вывода:

- модуль обработки дизъюнктивных нормальных форм (ДНФ) (ЛОГИКА);

- модуль циклического вывода (ЦИКЛ);
- модуль аварийных сообщений (АВАРИЯ);
- модуль управления внешними устройствами (ВЫВОД).

Модуль ЦИКЛ осуществляет подготовку информации для программы ВЫВОД и используется при необходимости циклического отображения состояния какого-либо узла системы. Условия, при которых активизируется модуль ЦИКЛ, задаются в виде ДНФ и обрабатываются модулем ЛОГИКА. Заполнение таблиц ДНФ происходит по указанию оператора в момент запроса на вывод.

Продемонстрируем построение операторских приказов и схему работы модулей ЛОГИКА, ЦИКЛ и ВЫВОД на примере.
Пусть нам нужно вывести на печать три канала с номерами 5,7 и 10 как одну группу с номером 4 на печать с циклом 15 сек.

1. Формирование группы.
   ГРУППА (4,5,7,10);

По этому приказу каналы с указанными номерами идентифицируются с группой под номером 4. Это указание формирует в программе ЦИКЛ строчку таблицы, позволяющую находить в поле результатов необходимую информацию.

2. Указание условий вывода.
   УСЛОВИЕ (4,15);

В данном случае программе ЛОГИКА дается указание о том,что с периодом в 15 секунд должна инициироваться группа с номером 4.

3. Пуск протокола.
   ПЕЧАТЬ (4);

По этому сообщению начинается отсчет времени для циклического вывода.

4. Конец протокола.
   СТОП (4);

5. Расформирование группы.
   КОНЕЦ (4);

Как видно из операторских приказов, в процессе работы мы можем изменять как состав группы, так и условие вывода.Если в приказе "УСЛОВИЕ" после номера канала стоит точка,то за этим должна следовать логическая функция, значение "истинно" которой служит указанием для протоколирования группы.

Например,

УСЛОВИЕ (4). (К3 > У3) и (К5 < К7)

Группа 4 инициируется, если значение канала 3 больше или равно соответствующей уставке, а значение канала 5 меньше показания канала 7.

Схема взаимодействия модулей при циклическом выводе следующая (в рассмотренном примере): по приказу "ПЕЧАТЬ" программа ЛОГИКА следит за тем, выполняются ли условия, заданные оператором. В положительном случае программе ЦИКЛ передается номер группы и управление. Модуль осуществляет подборку необходимой информации, редактирует ее и размещает в буфере. Кроме этого, выдается указание программе ВЫВОД, которая в свою очередь печатает подготовленную информацию и сообщает программе ЦИКЛ о конце операции с целью получения нового задания.

Наблюдение за аварийными ситуациями происходит при помощи программы АВАРИЯ. Функции ее такие же как и программы ЦИКЛ. Указание о возбуждении протокола сбоев программа получает от системного модуля КОНТРОЛЬ, который следит за изменением опрашиваемых величин. Кроме того, объем выводимой информации зависит от типа сбоя, от взаимосвязи с другими величинами и т.п.

Более подробная структура программ зависит от типа используемой УВМ. Кроме того, широта возможностей программ тесно связана с набором внешних выводимых устройств, следовательно, большее количество периферийных устройств (пишущие машинки, графопостроители, быстрая печать, перфораторы) позволяет более гибко использовать преимущество рассмотренной системы вывода.

ЛИТЕРАТУРА

1. Кузьмин Ю.Я. Об одной альтернативе программирования эксперимента. - Уч.зап.ЛГУ им.П.Стучки, Т.196. Кибернетизация научного эксперимента, вып.5. Рига, 1973, с.3-16.

2. Hirschberg G. Zielsetzung und Anwendung des Programmsystems MADAM. - "Siemens-Zeitschrift", 1971, 45, H.10, S.793-795.

3. Кузьмин Ю.Я., Гвоздев С.В., Банга А.Я., Котомин Е.Я. Особенности режима диалога в автоматизированном эксперименте. - Уч.зап.ЛГУ им.П.Стучки,т.196.Кибернетизация научного эксперимента, вып.5. Рига,1973, с.32-49.

4. Müller C., Zankl A. Prozeßüberwaschung mit dem Programmsystem MADAM. - "Siemens-Zeitschrift", 1971, 45, H.10, S.799-803.

УДК 681.3.01:007.51

Ю.Я.Кузьмин, Л.М.Кузьмина, Я.П.Цирулис

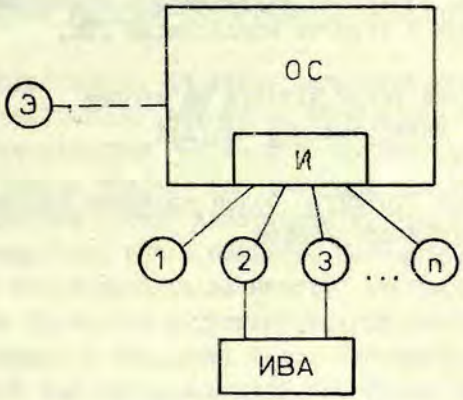
РЕЖИМ ДИАЛОГА В ПОИСКОВЫХ ЭКСПЕРИМЕНТАХ

Рассматриваются некоторые альтернативы разработки программного обеспечения экспериментов поискового характера, которым присуще формирование алгоритма в процессе экспериментирования и которые не требуют высокой скорости исполнения операций. Предлагается использовать режим диалога на уровне универсальных программных модулей. Режим диалога должен позволять в процессе диалога формировать и выполнять сложные фразы – последовательности простых директив.

Поисковый эксперимент является одним из важнейших элементов научного исследования, одновременно это и весьма трудный элемент. Основной особенностью такой разновидности эксперимента является изменчивость как аппаратуры, так и методики. Поисковый эксперимент часто кончается там, где начинается традиционное проектирование научно-исследовательской системы. Проблема автоматизация поискового эксперимента заключается в создании аппаратурного и соответствующего программного обеспечения, которое охватывает достаточный набор экспериментов и допускает его дальнейшее развитие без переделок. Программное обеспечение поискового эксперимента должно позволять отрабатывать алгоритм в ходе самого эксперимента. Отсюда ясно, что решение проблемы непосредственно зависит от развитости средств диалога "исследователь – система".

Режим диалога на уровне драйверов системы

Примером конкретной реализации достаточно простых средств диалога является разработанный фирмой Hewlett Packard язык ATS BASIC , расширенный до задач измерения и управления токовыми, частотными, релейными и другими каналами [1].



Р и с. I. Простая диалоговая система.

Э – экспериментатор; ОС – операционная система; И – интерпретатор; I +n – директивы; ИВА – измерительно-воздействующая аппаратура эксперимента.

Пример директивы в ATS BASIC (обращение к цифро-аналоговому преобразователю):

```
DCV   (N, V, C)
                 ------------ границы тока
              ---------------- диапазон напряжения
           ------------------- номер источника тока
        ---------------------- имя директивы
```

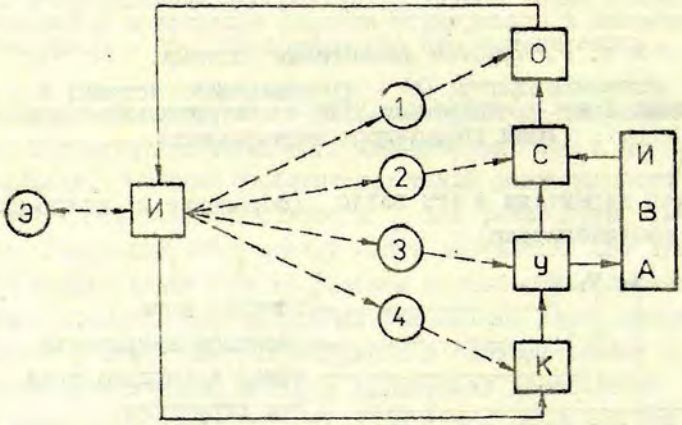
Рис.I. поясняет идею организации подобного программного обеспечения. Экспериментатор с телетайпа обращается к системе с помощью специальных директив, которые интерпретируются либо как драйверы аналого-цифровых преобразователей, релейных выходов и т.п., либо как специальные вычислительные действия.

Недостаток этого способа состоит в том, что уровень директив, которые может использовать экспериментатор, слишком низок и не позволяет описывать события эксперимента в компактной форме. Введение более сложных директив связано с написанием и отладкой подпрограмм, что не может быть выполнено в ходе поискового эксперимента. Другой недостаток заключается в том, что совмещение режима диалога с параллельным выпол-

нением функций сбора, контроля, управления и отработки возможно лишь для сложных и дорогих модификаций ЭВМ.

Простой режим диалога на уровне универсальных модулей

На рис.2 показана структура более сложного программного обеспечения поискового эксперимента.



Р и с. 2. Диалоговая система на основе универсальных программных модулей.

Модули: С – сбор; О – отработка; И – интерпретатор; К – контроль; У –управление; I + n – директивы управления модулями.

Здесь должно быть несколько заготовленных заранее программных модулей. Эти модули являются универсальными: в каждом из них сосредоточена одна из основных функций научно-исследовательской системы – сбор, обработка, контроль, управление и т.д. Для настройки модуля на конкретный состав системы в него должна быть заранее внесена информация специального вида. Это могут быть таблицы коэффициентов линеаризации и мас-

штабирования показаний датчиков и исполнительных механизмов, логические условия срабатывания исполнительных механизмов и т.п.

Естественно, за универсальность модулей приходится платить ухудшением некоторых параметров системы, например, повышением загрузки ЭВМ за счет выполнения дополнительных команд, однако реально для типичных экспериментов среднего быстродействия (поток информации на входе порядка 100 замеров в секунду) или очень больших, но медленных экспериментов (опрос сотен каналов за минуту) это ухудшение не принципиально, тем более, что в систему всегда можно ввести модуль селекции каналов и проводить опрос каналов и регулирование с оптимальными для каждого канала частотами в соответствии с временными и другими условиями. В то же время принцип универсальных модулей зачастую позволяет на порядок сократить затраты на программирование экспериментов, поскольку одни и те же модули могут легко и быстро перенастраиваться для решения разных задач.

Необходимый для поискового эксперимента режим диалога в подобной системе осуществляется не на уровне драйверов каналов ЭВМ, как обычно, а на уровне управления универсальными программными модулями. Поскольку степень интеграции универсальных модулей значительно выше, чем для драйверов каналов, то и набор директив, доступный экспериментатору, значительно ближе к описанию ситуаций эксперимента.

Известен набор универсальных программных модулей, получивший название MADAM, разработанный фирмой Siemens [2], а также набор простых модулей, описанный в статье [3].

Пример директив управления модулями [3] (обращение к модулю управления выходными каналами системы и к модулю контроля состояния системы):

ВЫДАТЬ В КАНАЛ N000 код температуры 1250°С +
СООБЩИТЬ -АВАРИЙНЫЕ КАНАЛЫ +

Эти, простые с точки зрения экспериментатора, директивы на самом деле эквивалентны нескольким драйверам системы.

Важно, что режим диалога при подобной организации программного обеспечения может быть совмещен с выполнением основных функций научно-исследовательской системы, независимо от

конкретной цели поискового эксперимента, поскольку модули постоянно находятся в действии, информируя экспериментатора о текущем состоянии и выполняя его директивы при непосредственном осуществлении поискового эксперимента.

Однако при такой организации диалога неудобно выполнение циклически повторяющихся последовательностей директив, их каждый раз приходится вводить вручную.

Режим диалога на уровне фраз

Рассмотренный (см. [3 ; с.50-54]) вариант простого расширения режима диалога на уровне универсальных программных модулей позволяет выполнять циклически повторяющиеся последовательности директив. Ниже будет рассмотрена соответствующая альтернатива построения программного обеспечения поисковых экспериментов на основе применения универсальных программных модулей и специального вида интерпретатора.

Прежде всего выделим набор универсальных программных модулей, позволяющих реализовать основные функции эксперимента - сбор измерительной информации, управление аппаратурой, первичную обработку результатов измерений, контроль состояния и др. Определим формат обращения к этим модулям в виде директив:

/ДИРЕКТИВА/ ::= /ИМЯ ДИРЕКТИВЫ//n_1, ...,n_k/ +

где /n_1, ...,n_k/ - параметры управления модулем, + - признак конца директивы.

Набор директив вместе с интерпретатором образует первый, низший уровень системы программирования. Это то, что должно быть введено в ЭВМ до начала поискового эксперимента. Сам поисковый эксперимент формируется в режиме диалога и соответствует комбинациям директив (фраз) часть из которых может быть введена заранее, при осуществлении предыдущих экспериментов.

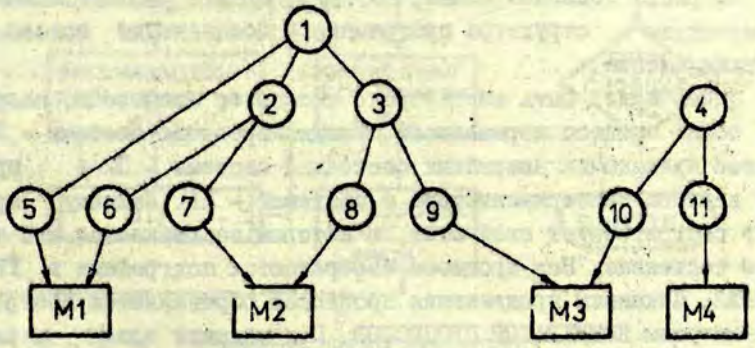
Определим сложные последовательности директив (фразы) следующим образом:

/ФРАЗА/ ::=/ДИРЕКТИВА/ | /ДИРЕКТИВА/ /ФРАЗА/ +

Очевидно, что фразы имеет смысл вводить для сокращения обращений к системе, поскольку они соответствуют определению сложного действия экспериментатора через простые, наиболее

часто повторяющиеся для экспериментов комбинации директив. Эти стереотипные последовательности, которые экспериментатор находит во время поискового эксперимента, он закрепляет в памяти ЭВМ путем присвоения им новых имен и указания переменных параметров простых директив (впрочем, простые директивы сами могут быть фразами). Так порождаются новые директивы, которые дополняют имеющиеся, но отличаются от них большей интегрированностью и тем, что они могут быть порождены или уничтожены в ходе поискового эксперимента. Совокупность новых директив образует ГРАФ ФРАЗ, узлами которого являются директивы, а ребра указывают на компоненты.

Можно сделать так, чтобы в системе формировался только один ГРАФ ФРАЗ, независимо от числа пользователей. Это позволит удачные методы , найденные в поисковых экспериментах одними авторами, использовать всеми пользователями системы. Введение фраз осуществляется специальным программным модулем ФОРМИРОВАТЕЛЬ ГРАФА, который начинает работать, если не обнаруживает подобной фразы в памяти ЭВМ, одновременно он заносит имя новой директивы в список имен всех директив.



Р и с.3. Пример ГРАФА ФРАЗ.
Цифрами обозначены директивы управления модулями.
MI – M4 – модули.

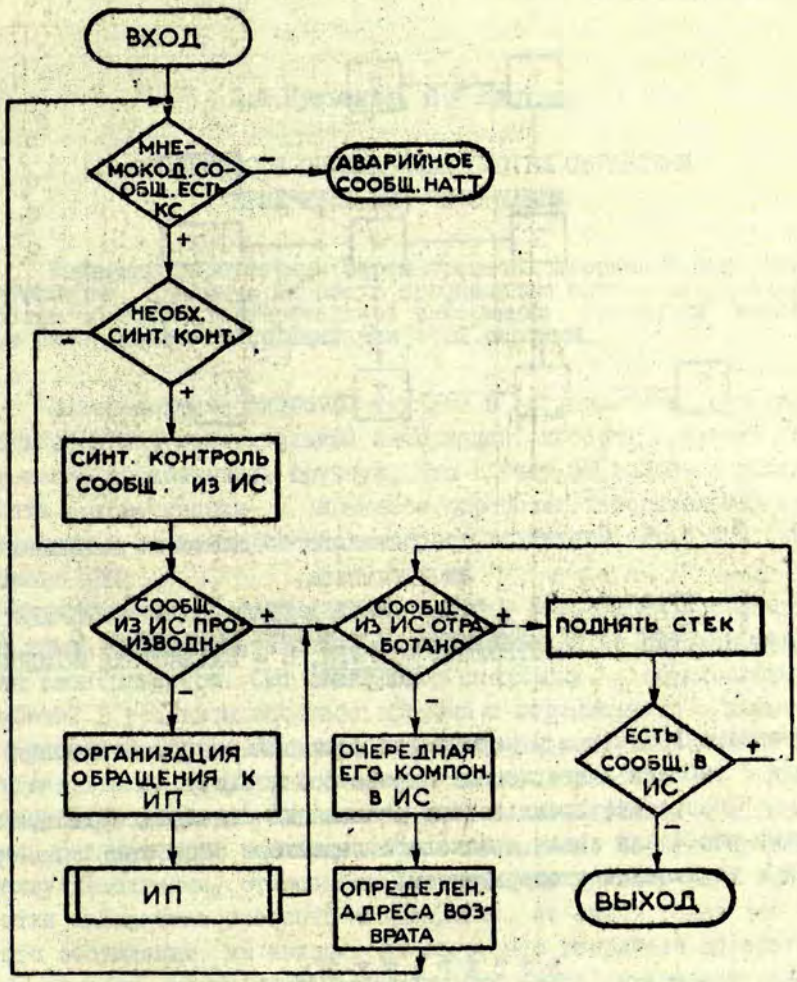
Отработка директивы, включенной в ГРАФ ФРАЗ, происходит следующим образом. Введенную экспериментатором директиву модуль ИНТЕРПРЕТАТОР сравнивает со списком наличных директив, опознав ее, он определяет адрес нижнего узла в ГРАФЕ ФРАЗ, с которого начинается отработка директивы. Если это была директива непосредственной связи с модулем (например, 5 на рис.3), то модулю передаются все параметры, необходимые для отработки директивы, и она выполняется. Если же это была сложная директива, то первой ее компоненте передается та часть информации, которая требуется для задания ее переменных параметров, а затем она также расшифровывается. Процесс расшифровки и заполнения переменных параметров идет "сверху вниз" и продолжается до тех пор, пока не будет достигнут самый нижний уровень, т.е. уровень тех директив, которые непосредственно связаны с модулями. После отработки последовательности простых директив (например, 6,7 на рис.3), определяющих более сложную (2 на рис.3), ИНТЕРПРЕТАТОР вызовет следующую составную директиву (3 на рис.3), определит ее переменные параметры и отработает соответствующие ей простые директивы. Этот процесс закончится тогда и только тогда, если будут отработаны все компоненты директивы, заданной экспериментатором.*

На рис.5 показана общая, соответствующая рассматриваемой альтернативе структура программного обеспечения поискового эксперимента.

Здесь может быть инициировано несколько процессов, например, общий процесс нормального функционирования системы – П3, процесс ликвидации аварийных состояний системы – П2 и процесс диалога экспериментатора с системой – П1. Каждому процессу соответствует свой стек, в котором запоминается его текущее состояние. Все процессы изображаются подграфами в ГРАФЕ ФРАЗ. Механизм продвижения процессов определяется программным модулем КОММУТАТОР ПРОЦЕССОВ. При задании новых директив участвует ФОРМИРОВАТЕЛЬ ГРАФА. На стадии генерации программного обеспечения системы участвует НАСТРОЙЩИК МОДУЛЕЙ, который из БИБЛИОТЕКИ ИСХОДНЫХ МОДУЛЕЙ выбирает нужные РАБОЧИЕ МОДУЛИ и настраивает их на систему (заполняет таблицы со-

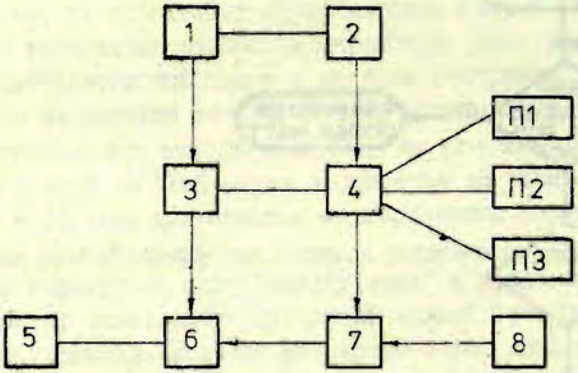
---

* Структура ИНТЕРПРЕТАТОРА приведена на рис.4.



Р и с. 4. Структура ИНТЕРПРЕТАТОРА.

ИП – исполнительная программа; ИС – исполнительный стек.



Р и с. 5. Структура программного обеспечения поискового эксперимента.

I - ГРАФ ФРАЗ, 2 - ФОРМИРОВАТЕЛЬ ГРАФА, 3 - ИНТЕРПРЕТАТОР, 4 - КОММУТАТОР ПРОЦЕССОВ, 5 - объект управления, 6 - РАБОЧИЕ МОДУЛИ, 7 - НАСТРОЙЩИК МОДУЛЕЙ, 8 - БИБЛИОТЕКА ИСХОДНЫХ МОДУЛЕЙ.

ответствия датчиков и параметров физическим адресам каналов, задат таблицы линеаризации параметров и т.п.).

В настоящее время начата реализация отдельных фрагментов схемы рис.4 для задач поискового характера обработки измерений и управления экспериментом.

ЛИТЕРАТУРА

I. HP ATS BASIC, printed USA, Hewlet Packard, I972.

2. "Siemens-Zeitschrift", I97I, 45, H.IO, S.793-799.

3. Кузьмин Ю.Я. Об одной альтернативе программирования эксперимента. - Уч.зап.ЛГУ им.П.Стучки, т.I96. Кибернетизация научного эксперимента, вып.5. Рига, I973, с.3-59.

УДК 681.3.01:007.51

Л.М.Кузьмина, Я.П.Цирулис

ОСОБЕННОСТИ ОРГАНИЗАЦИИ СИСТЕМ ОБРАБОТКИ ИЗМЕРИТЕЛЬНОЙ ИНФОРМАЦИИ

Выявлены характерные черты процесса вторичной обработки информации, влияющие на общую организацию автоматизированных систем обработки измерительной информации. Приведены некоторые соображения об организации этой системы.

Использование цифровой техники в эксперименте порождает большой поток измерительной информации, которую зачастую невозможно обрабатывать вручную. Это влечет за собой необходимость автоматизации процесса обработки измерительной информации и создания специализированных систем обработки на основе ЦВМ.

Первичная обработка измерительной информации проводится во время эксперимента с целью использования ее для управления экспериментом. Она имеет свою специфику, обусловленную работой в реальном масштабе времени с ограниченной памятью. В настоящей статье мы рассмотрим лишь задачи, возникающие при автоматизации вторичной (окончательной) обработки, во время которой из результатов эксперимента извлекается та информация, ради которой эксперимент был поставлен. Четкую границу между действиями, относящимися к тому или другому виду обработки информации, провести нельзя, но это не очень важно для нашего обсуждения: мы исходим из того, что вторичная обработка, как правило, может быть проведена уже после завершения самого эксперимента.

Во многих случаях вторичная обработка сводится к последовательному выполнению более или менее определенных и независимых друг от друга шагов или действий. В некоторых спектральных исследованиях это, например, может заключаться в

следующем: I) просматривается серия однотипных спектров, удаляются аномальные выбросы, свидетельствующие о явных ошибках, или проводятся другого вида коррекции; 2) вычитается фон; 3) исключается влияние аппаратной функции; 4) спектры суммируются (усредняются); 5) меняется масштаб; 6) результирующий спектр разлагается на отдельные полосы, например, гауссианы. Обычно алгоритмы, применяемые на каждом шаге, представляют собой некоторые вычислительные процедуры и могут быть запрограммированы на одном из имеющихся языков высокого уровня. Если методика эксперимента и обработки измерительной информации установившаяся, а характер обрабатываемой информации заранее известен, то из таких подпрограмм можно составить полный алгоритм для комплексной обработки экспериментальных данных. Все это требует определенной квалификации от составителя программ, и если физическая сущность обрабатываемой информации не играет заметной роли при выборе вычислительных методов, то программы редко пишутся самими экспериментаторами: последние выступают в качестве "заказчиков".

В более сложных ситуациях результаты, полученные на первых этапах обработки, могут существенно повлиять на дальнейшие действия. Так, например, в вышеуказанной последовательности даже после усреднения может потребоваться сглаживание результирующего спектра, а затем, в зависимости от его особенностей (число пиков, их разрешение и т.п.), – тот или другой алгоритм разложения. Кроме того, при изменении характера обрабатываемой информации приходится изменять имеющуюся программу, форматы данных или даже заново отрабатывать методику обработки, т.е. опробовать те или другие сочетания элементарных подпрограмм. Здесь для принятия решений необходимо активное участие экспериментатора в процессе обработки, поскольку для нахождения приемлемого пути важно учитывать физическую сущность обрабатываемой информации. Если, наоборот, элементарные подпрограммы соответствуют категориям, которыми мыслит экспериментатор, то отпадает необходимость привлекать к принятию решения программистов и математиков. Их задачи тогда могут быть сведены к пополнению набора этих подпрограмм.

Из сказанного понятно, что основная цель, к которой следует стремиться при создании системы для комплексной обработ-

ки измерительной информации – это обеспечение тесного контакта между пользователем-экспериментатором и ЦВМ. В частности, такая система должна быть ориентирована на человека, а не на машину. Другая особенность этой системы заключается в том, что ее "профессиональная ориентация" определяется исключительно набором элементарных подпрограмм обработки и допустимыми видами и форматами информации. Не изменяя ядро системы, ее можно настраивать на те или иные задачи.

Обзор ряда реализованных аналогичных систем, например, ПОФИ-I, ПОФИ-2, СОФИ, КОД, МАДАМ[I-6]*, отличающихся друг от друга некоторыми особенностями структуры и организации или обеспечением связи экспериментатора с машиной, подтверждает, что автоматизированные системы комплексной обработки измерительной информации должны удовлетворять по крайней мере следующим требованиям (порядок перечисления этих требований является в некоторой степени случайным; в зависимости от конкретных условий предпочтение можно отдать тем или другим из них):

I) универсальность: система не должна быть ориентирована на конкретный алгоритм обработки;

2) гибкость: система должна обеспечить доступ к библиотечным подпрограммам, дать возможность подключать индивидуальные подпрограммы, а также по-разному компоновать имеющиеся в распоряжении пользователя подпрограммы и по своему усмотрению выбрать вид представления информации;

3) стабильность: адаптация и расширение системы должно осуществляться без изменения ее структуры;

4) надежность: должна контролироваться работа системы, ЭВМ и в особенности пользователя;

5) простота обращения: должен иметься простой язык общения пользователя с системой;

6) удобство пользования: система должна обеспечить широкий сервис. Кроме того, желательно иметь возможность вызвать к работе ту или иную модификацию системы в зависимости от целей и наличия свободных ресурсов.

---

* Также частное сообщение Б.Д.Брока о системе управления и обработки данных системы "АНАКОНДА", применяемой в газхроматографии.

Перечисленные требования относятся к функционированию системы. Но они во многом определяют также внутреннюю организацию системы, ее структуру. Укажем на некоторые особенности этой организации:

1) язык общения ориентирован на работу с подпрограммами и менее приспособлен для описания отдельных арифметических, логических и других вычислений, т.е. для составления самих подпрограмм. Он содержит также средства для определения новых видов данных;

2) система работает в режиме диалога; это позволяет для расшифровки сообщений пользователя применять интерпретатор. Имеются средства для визуального представления информации;

3) сообщения пользователя имеют по возможности однообразную структуру, что позволяет стандартизовать их обработку;

4) система состоит из отдельных модулей, выполняющих строго определенные функции. В частности, четко разделены средства для собственно обработки экспериментальных данных и средства для обслуживания и развития системы;

5) имеется динамическое распределение памяти, причем система осуществляет это самостоятельно – без участия пользователя.

Итак, система обработки должна содержать следующие основные блоки:

1) блок, организующий обмен сообщений между пользователем и системой;

2) интерпретатор для обработки сообщений пользователя;

3) библиотека системы, содержащая модули системы и библиотечные подпрограммы;

4) каталог подпрограмм и их описаний;

5) индивидуальные библиотеки;

6) банк файлов измерительной информации;

7) генератор системы;

8) загрузчик и распределитель памяти.

При чем каждый из этих блоков может быть в большей или меньшей степени развитым. Это во многом зависит от структуры языка общения с системой.

ЛИТЕРАТУРА

1. Воробьева Н.Н., Нефедьева Л.С. Язык общения в системах приема и обработки физической информации. Препринт ОИЯИ 10-4595, Дубна. 1969.

2. Говорун Н.Н., Нефедьева Л.С.О проблемах математического обеспечения ЭВМ в задачах автоматизации обработки спектроскопической информации. - "Информационный бюллетень Совета по автоматизации научных исследований АН СССР", 1971, №3, с.13-18.

3. Воробьева Н.Н., Нефедьева Л.С. Общая организация математической обработки числовых масоивов в системе ПОФИ-2.- "Информационный бюллетень Совета по автоматизации научных исследований АН СССР", 1971, №3, с.19-23.

4. Hirschberg G. Zielsetzung nd Anwendung des Programmsystems MADAM. - "Siemens Zeitschrift", 1971, 45, H 10, S. 793-795.

5. Wendelin R. Aufbau und Funktion des Programmsystems MADAM. - "Siemens Zeitschrift", 1971, 45, H 10, S.796-799.

6. Mosedale T.W. Pedant. - "Software - pract. and expert.", 1973, 3, Nr.2, p.121-143.

УДК 62-50:007.51

Л.М.Кузьмина, А.Н.Назарова

ДИАЛОГОВАЯ СИСТЕМА ДЛЯ ОБРАБОТКИ СПЕКТРОВ - ВАРИАТОР

Рассмотрена простая система для обработки спектров, построенная по модульному принципу и ориентированная на режим диалога. Система допускает включение и отработку директив пользователя соответствующим последовательностям подпрограмм - производным подпрограммам.

В 1971 году в Проблемной лаборатории физики полупроводников был предложен простой алгоритм разложения спектров на элементарные полосы в системе ч е л о в е к - м а ш и н а [I]. Напомним, что процесс обработки спектров ведется при непрерывном взаимодействии экспериментатора с цифровой вычислительной машиной (ЦВМ), причем на долю первого приходятся операции качественного характера, а ЦВМ в то же время выполняет все вычислительные процедуры. Приведем некоторые данные по использованию предложенного алгоритма в течение трех лет. Было разложено около 150 полученных на спектрометре UR -20 спектров кристаллов типа $KBr + SO_4^{-2} + Cu$. Спектры разлагались на полосы, описываемые гауссовыми и лоренцевыми функциями. Каждый из таких спектров мог содержать до восьми полос. В среднем на анализ одного спектра, состоящего из пяти-шести полос тратилось 40-50 минут.

Эксплуатация программы, реализующей указанный алгоритм, выявила некоторые ее недостатки как методического, так организационного характера:

- для разложения одного и того же спектра на полосы описываемые разными функциями, экспериментатору приходилось вводить в УВМ соответствующую каждой функции программу и повторять цикл разложения сначала ;

- для разных пользователей программы, при небольших различиях в алгоритмах обработки, требовалось вмешательство программиста с целью изменить программу;

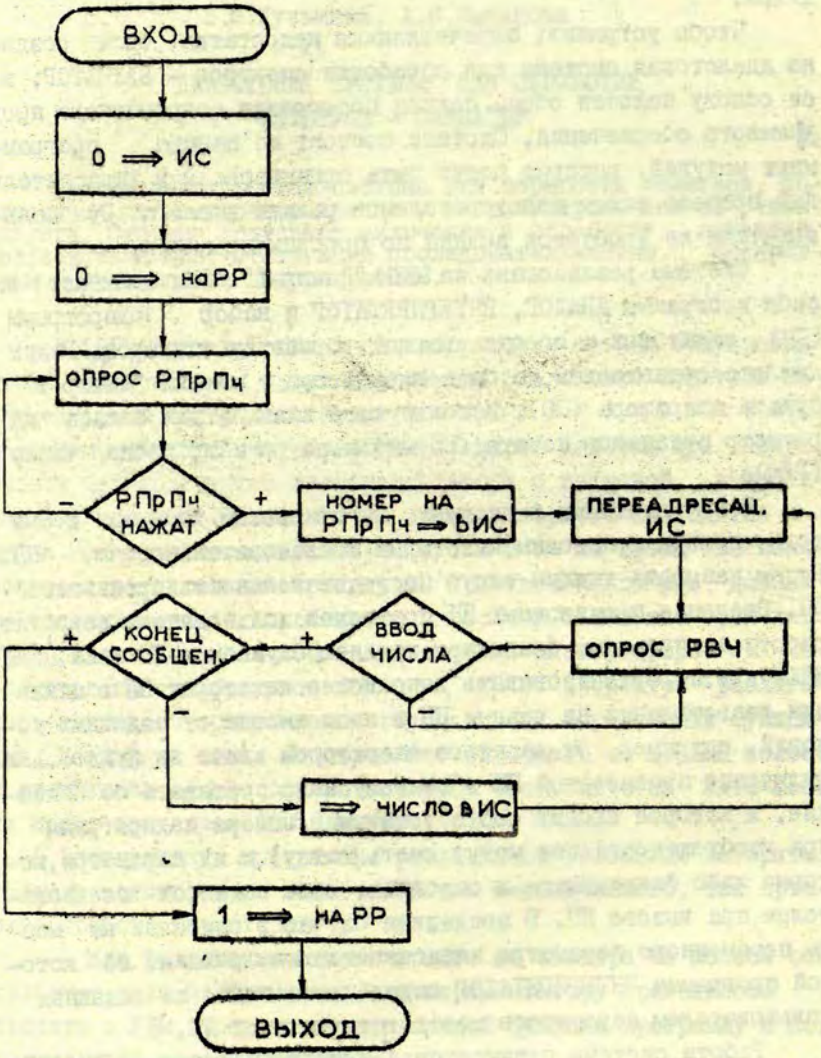
– расширение программы требовало ее значительных переделок.

Чтобы устранить перечисленные недостатки, была создана диалоговая система для обработки спектров – ВАРИАТОР; в ее основу положен общий подход построения современного программного обеспечения. Система состоит из набора программных модулей, которые могут быть организованы в вычислительный процесс самим пользователем в режиме диалога. От пользователя не требуется знаний по программированию.

Система реализована на ЭВУМ "Днепр-I". Она включает в себя программы ДИАЛОГ, ИНТЕРПРЕТАТОР и набор подпрограмм (ПП), служебных и обеспечивающих обработку спектров. Режим диалога организован на базе имеющегося у машины "Днепр-I" пульта оператора (ПО). Используются клавиши для записи на регистр признаков печати (РПрПч) и на регистр ввода числа (РВч):

Всем входящим в систему ПП присвоены номера; кроме того, номера присвоены некоторым последовательностям ПП. Будем называть каждую такую последовательность производной ПП. Введение производных ПП позволяет пользователю сократить работу за пультом, благодаря наличию служебных ПП типа ЦИКЛ, КЛЮЧ и т.п., организовывать выполнение некоторых ПП в цикле или разветвления на разные ПП в зависимости от заданных условий, например, от нажатого оператором ключа на пульте. Для включения производной ПП в систему надо составить ее описание, в которое входят соответствующие номера подпрограмм (при необходимости они могут иметь метку) и их параметры, которые либо фиксируются в описании, либо задаются пользователем при вызове ПП. В последнем случае в описании на месте переменного параметра записывается информация, по которой программа ИНТЕРПРЕТАТОР определяет, какие из заданных пользователем параметров соответствуют этой ПП.

Работа системы начинается с запуска программы ДИАЛОГ, которая в цикле опрашивает РПрПч (блок-схема на рис.I). Для обращения к ПП пользователь задает ее номер на РПрПч и переменные параметры – на РВч. ДИАЛОГ управляет также заданием режимов на регистр режимов (РР), по которому пользова-


ВХОД
0 ⟹ ИС
0 ⟹ на РР
ОПРОС РПр Пч
РПр Пч НАЖАТ
НОМЕР НА РПр Пч ⟹ ЬИС
ПЕРЕАДРЕСАЦ. ИС
КОНЕЦ СООБЩЕН.
ВВОД ЧИСЛА
ОПРОС РВЧ
⟹ ЧИСЛО В ИС
1 ⟹ НА РР
ВЫХОД

Р и с. I.

тель определяет, находится ли система в состоянии счета или ждет указаний пользователя. После ввода в машину признака конца сообщения ДИАЛОГ передает управление на ИНТЕРПРЕТАТОР. который начинает работу с вывода на печатающее устройство введенной информации, тем самим возлагая ответственность за ее правильность на самого пользователя. Далее осуществляется проверка наличия заданного номера в каталоге подпрограмм (КП). При отсутствии такового на печатающем устройстве выводится "НЕТ ПОДПРОГРАММЫ" и система ждет новых указаний пользователя. В КП записаны номера всех имеющихся в системе ПП признаки простых или производных ПП, указатели адресов ПП отработки, если они простые, или адресов, по которым находятся описания, если они производные. Все ПП, входящие в систему, написаны по единому стандарту. Их работа заканчивается формированием возврата на ИНТЕРПРЕТАТОР, переменным параметрам отводится строго определенное поле памяти для всех ПП. ИНТЕРПРЕТАТОР организует пересылку необходимых параметров в это поле и передачу управления на ПП отработки.

Выполнение производных подпрограмм ИНТЕРПРЕТАТОР организует по схеме, описанной в статье. [*] Для примера рассмотрим здесь одну производную ПП. Ее описание имеет вид:

I.	MI	ПП7 (p_1) – опрос пульта
2.		ПП9 (p_1, p_2, p_3) – вычисление нового спектра; полосы описываются лоренцевой функцией
3.		ПП10 – вывод на осциллоскоп
4.		ПП3 (p_4) – КЛЮЧ
5.		ПП4 (p_5) [**] – ЦИКЛ на метку MI
6.	Признак конца описания	

Для обращения к этой сложной ПП пользователю достаточно задать ее номер – I8. Далее ИНТЕРПРЕТАТОР организует выполнение следующих ПП: опрос пульта, с которого задаются параметры полос (ПП7); вычисление нового спектра, полосы которого описываются лоренцевой функцией, а параметры заданы с пульта (ПП9); вывод на осциллоскоп исходного спектра, вычис-

---

[*] См. настоящий сборник, с. 90.

[**] $p_1, \ldots, p_5$ – параметры ПП.

ленного спектра и разности между ними (ПП10); (ПП3) – КЛЮЧ осуществляет проверку, нажат или не нажат пультовый ключ, номер которого задан в виде переменного параметра этой ПП. Если ключ не нажат, управление передается следующей (ПП4), заданной в описании. Если ключ нажат, то одна ПП из описания пропускается и управление передается ПП, следующей за ней. В нашем примере описание кончилось и управление передается на ИНТЕРПРЕТАТОР. ПП4 (ЦИКЛ на метку М1) организует через ИНТЕРПРЕТАТОР передачу управления на ПП7. Таким образом в системе организуется циклический опрос пульта и последовательная подгонка параметров всех полос.

Предлагаемая диалоговая система для обработки спектров предоставляет пользователю следующие возможности:

1. В режиме диалога реализуются произвольные алгоритмы обработки из имеющихся в наборе ПП; изменение алгоритма в процессе работы не представляет никаких трудностей.

2. В системе имеются сложные ПП, позволяющие за одно обращение реализовывать целый комплекс модулей, выполнение которых организуется не обязательно последовательно, благодаря подпрограммам ЦИКЛ, КЛЮЧ и т.п.

3. Расширение системы осуществляется за счет добавления новых подпрограмм.

Отметим ряд особенностей предлагаемой методики, выявившихся в результате трехлетнего использования программы, чтобы предостеречь пользователей системы от мнения, что, не имея никаких сведений о спектре, его можно разложить. При решении задачи анализа спектров оператор обязательно должен иметь некоторое представление о структуре обрабатываемых спектров. Для однозначного разложения требуется, кроме того, знание числа полос в спектре, и по крайней мере, один параметр каждой полосы. На практике чаще всего приходится обрабатывать спектры, о которых нет детальных сведений. В таком случае спектр должен быть разрешен настолько, чтобы экспериментатор мог "на глаз" правильно определить число полос в нем. Но даже и тогда отсутствуют критерии, характеризующие качество разложения в смысле его достоверности, и здесь приходится рассчитывать на опыт и интуицию экспериментатора, а также на удачный выбор формы полос. Отме-

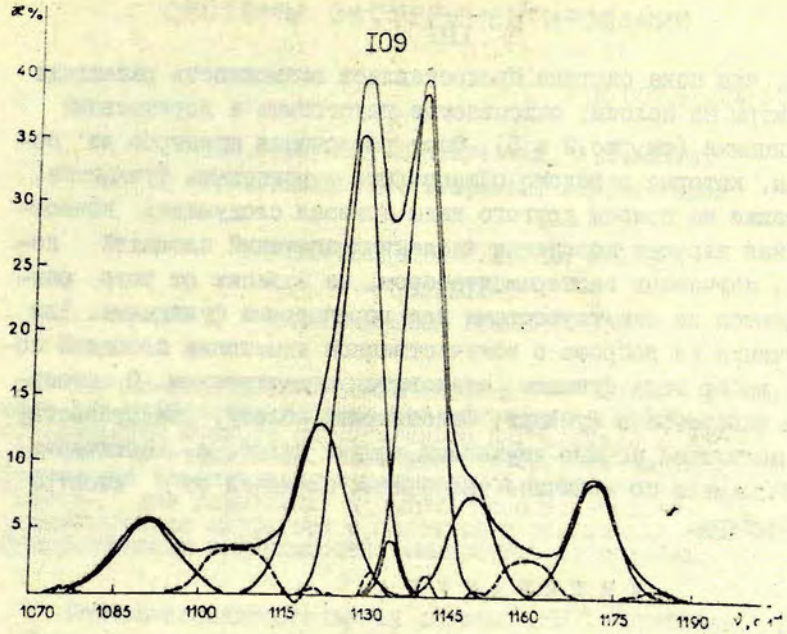

x %
40
35
30
25
20
15
10
5
1070
1085
1100
1115
1130
1145
1160
1175
1190
ν, см⁻¹

Рис. 2.

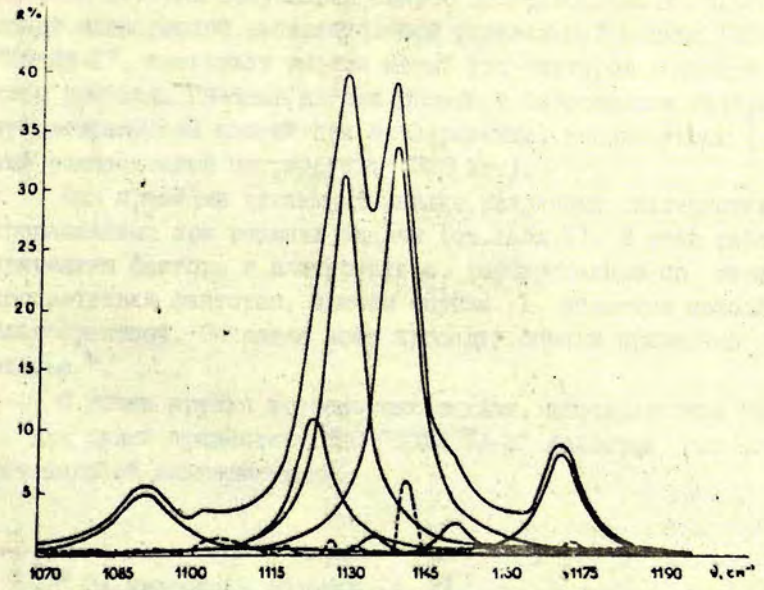

x %
40
35
30
25
20
15
10
5
1070
1085
1100
1115
1130
1145
1160
1175
1190
ν, см⁻¹

Рис. 3.

тим, что пока система предоставляет возможность разлагать спектры на полосы, описываемые гауссовыми и лоренцевыми функциями (см.рис.2 и 3). Опыт разложения спектров на полосы, которые заведомо описывались лоренцевыми функциями, а также на полосы другого вида показал следующее: качественная картина поведения численных значений площадей полос, изучаемых экспериментатором, не зависит от того, описываются ли они гауссовыми или лоренцевыми функциями. Для изучения же вопроса о количественном изменении площадей полос выбор вида функции становится существенным. О неверном выборе вида функции, описывающей полосу, свидетельствует появление, помимо изучаемых, ложных полос, с хаотически меняющимися по величине значениями площадей от спектра к спектру.

ЛИТЕРАТУРА

I. Закис Ю.Р., Кузьмин Ю.Я., Кузьмина Л.М., Москальонов А.В., Пуце Л.Р. Анализ спектров в системе "человек-машина". – "Ж.прикл.спектроскопии", 1972, 17, 6, с.1098-1101.

УДК 007.5.004:50

Я.Я.Страумен, Ю.Я.Кузьмин, Л.Я.Банга,
С.В.Гвоздев, М.Я.Зариньш, А.А.Паспарне

НАУЧНО-ИССЛЕДОВАТЕЛЬСКАЯ СИСТЕМА "GUNDEGA-2" ДЛЯ ПОИСКОВЫХ ЭКСПЕРИМЕНТОВ

Научно-исследовательская система "GUNDEGA-2" предназначена для проведения сложных теплофизических испытаний с использованием мощного радиационного нагревателя и камеры охлаждения со свойствами, близкими к свойствам черного тела. Применение электронной вычислительной управляющей машины "Днепр-I" для управления испытаниями обеспечивает удобное взаимодействие оператора с системой и реализацию различных функциональных зависимостей температуры от времени.

Научно-исследовательская система (НИС) "GUNDEGA-2" разрабатывалась для целей автоматизации теплофизических испытаний образцов большой поверхности. Уже на первых стадиях разработки системы обсуждался вопрос целесообразности использования электронной вычислительной управляющей машины (ЭВУМ) "Днепр-I", поскольку имелся целый ряд факторов в пользу такого решения. Главные из них связаны с оперативным контролем и предотвращением аварий при эксперименте, использующем мощный радиационный нагреватель (320 квт).

Был проведен системный анализ различных альтернатив, предложенных при решении задачи (см.табл.I). В этой таблице приведены факторы и альтернативы, ранжированные по степени соответствия факторам, причем оценка 1 является наиболее благоприятной. Описание всех процедур оценки приведено в статье *.

С точки зрения приведенных оценок, использование "Днепр-I" для целей применения НИС "GUNDEGA-2" является наиболее оптимальной альтернативой.

---

* См.настоящий сборник, с. 32.

Таблица I

Факторы (ранжирование)	Альтернативы				
	идеальная	"Днепр-I"	локальн. автоматы	ручн. упр.	пессимистич.
I. Безопасность эксперимента	I	I	2	3	3
2. Качество эксперимента	I	I	2	3	3
3. Быстрота разработки	I	2	3	I	3
4. Простота эксплуатации	I	2	3	I	3
5. Простота управления экспериментом	I	I	2	3	3
6. Целевизна эксперимента	I	3	2	I	3
7. Простота диалога с аппаратурой	I	I	2	3	3
8. Простота сбора информации, контроля	I	I	2	3	3
9. Возможность развития аппаратуры	I	I	3	2	3
ИТОГО, %	I00	25	53	50	0,0
$\pm\Delta$	—	9	I3	II	—

Технические средства НИС "GUNDEGA-2"

НИС "GUNDEGA-2" выполнена по модульному принципу, т.е. отдельные ее узлы могут иметь самостоятельное применение, либо входить в качестве элементов в другие системы управления экспериментом. Опасения о возможности непосредственного цифрового управления значительной мощностью (до 320 квт) не оправдались. Как показал опыт, правильно организованные каналы управления и подбор программных режимов позволят избежать помех, которые могут возникнуть при регули-

рованни большой мощностью.

Ниже рассмотрим основные характеристики системы и ее технический состав.

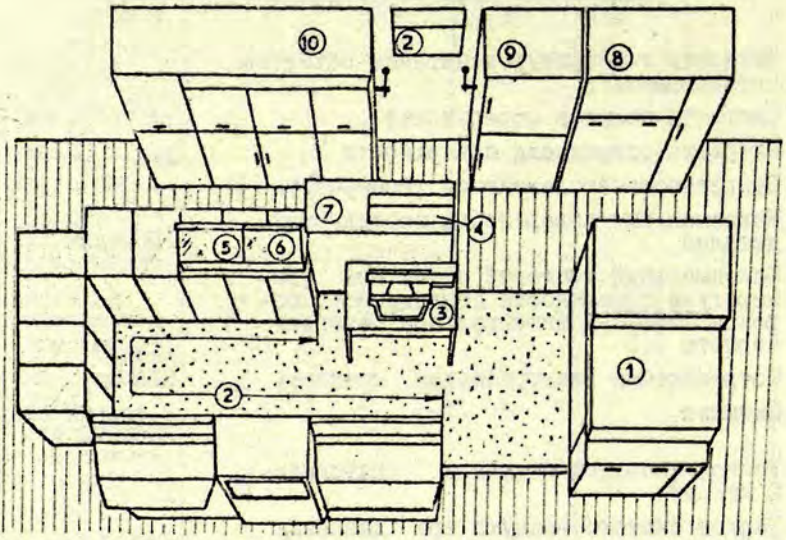
Технические характеристики НИС "GUNDEGA-2"

Интервал температур испытания объектов исследования	I00-I500°C
Скорость нагрева поверхности	до 20°C/сек
Скорость охлаждения поверхности	до I0°C/сек
Погрешность отслеживания температуры	~ 3%
Максимальная поверхность исследуемых деталей	I000 $см^2$
Максимальный тепловой поток при температуре поверхности исследуемого образца I500°C и интегральной степени черноты 0,3	I00 $вт/см^2$
Потребляемая электрическая мощность	320 квт
Питание	3-фазная сеть 380/220 в
Расход холодной воды при давлении 2 $кг/см^2$	I5 $м^3$/час
Расход сжатого воздуха при давлении 5-6 атм	200 $м^3$/час
Требуемая производительность отсосной вентиляции	600 $м^3$/час

Датчики измерения и контроля физических величин и каналы управления системой, имеющие непосредственную связь с ЭВУМ (количество)

Термопары	7
Двухпозиционные датчики потока жидкости и газа	8
Датчики давления, непрерывные	2
Датчики давления, двухпозиционные	8
Датчики положения, двухпозиционные	5
Каналы тиристорного регулятора температуры	3
Каналы управления вентилями газа и воды	I3

Система выполнена в напольном варианте и занимает площадь 60 м^2. Внешний вид системы показан на рис.I.

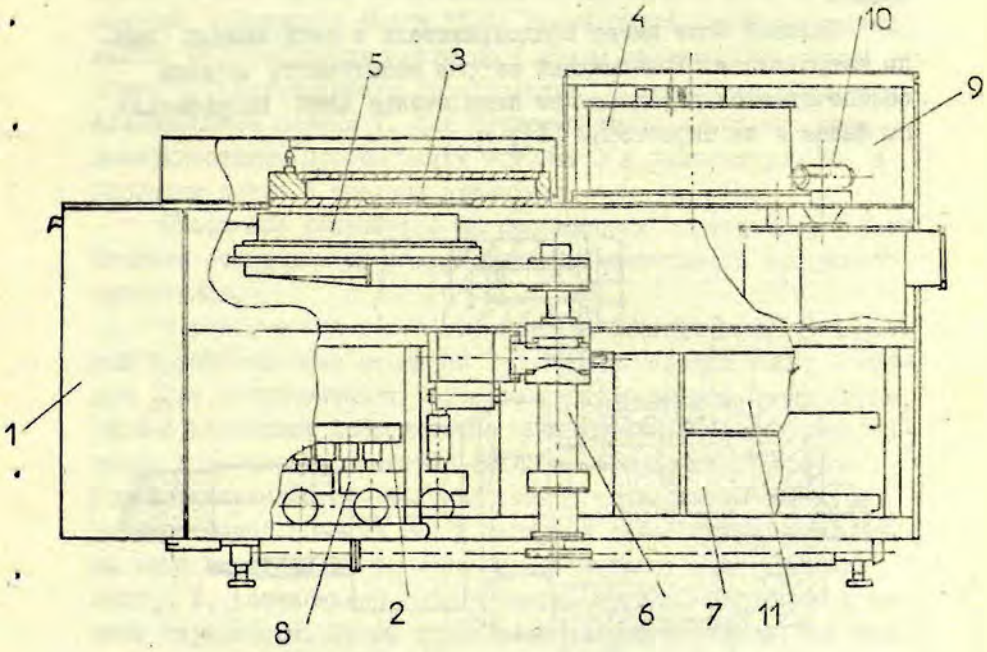


Р и с. I. Общий вид НИС "GUNDEGA-2".

I - испытательный стенд, 2 - ЭВУМ "Днепр-I", 3 - электрическая печатающая машинка (ЭПМ); 4 - табло, 5 и 6 - двухкоординатные самопишущие потенциометры ПДС-021М , 7 - кросс-шкаф, 8 - шкаф магистралей, 9 - силовой автомат, I0 - распределительный шкаф.

Ниже будут рассмотрены назначение и краткое описание основных технических средств системы.

И с п ы т а т е л ь н ы й с т е н д (рис.2) позволяет исследовать образцы с плоской поверхностью площадью до I000 см^2. Для обеспечения теплового режима испытания в стенде имеется соответствующая основная подсистема и вспомогательные устройства. В подсистему теплового режима входят блоки I,2,3,4,5,6 и 7, а к вспомогательным устройствам относятся блоки 8,9,I0 и II.



Р и с. 2. Испытательный стенд.

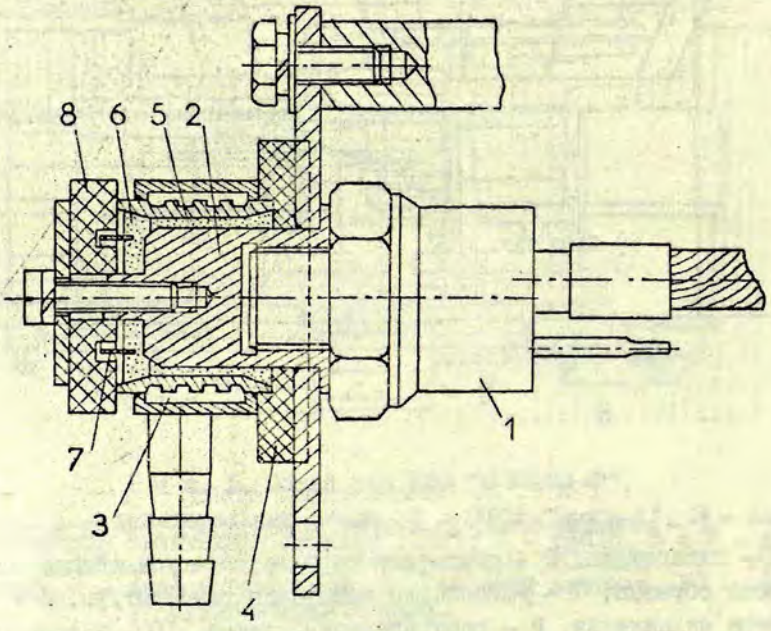
I – силовой блок, 2 – блок управления, 3 – нагреватель, 4 – охладитель, 5 – держатель образца, 6 – устройство поворота образца, 7 – устройство измерения температуры, 8 – камера охлаждения, 9 – распределитель газов, I0 – устройство отсоса газов, II – блок электронных ключей.

Поскольку испытательный стенд является сложной подсистемой, приведем краткое описание наиболее важных его элементов.

С и л о в о й б л о к представляет собой охлаждаемую водой панель (см.рис.3) с девятью охлаждаемыми водой мощными тиристорами марки ВКДУС-I00 для питания радиационного нагревателя (кварцевые лампы). Управление мощностью осуществляется включением тиристоров на заданное число полуволн питающего тока [I]. Такой способ регулирования большой мощнос-

ти позволил избежать проблем, связанных с помехами, и применить ЭВУМ "Днепр-I" для непосредственного цифрового управления.

Силовой блок имеет предохранители в цепи каждой лампы нагревателя. Равномерный нагрев испытуемого образца обеспечивается чередованием подключения ламп нагревателя по фазам и по тиристорам [I].



Р и с. 3. Схема крепления охлаждаемого водой тиристора ВКДУС-I00.

I - тиристор, 2 - втулка, 3 - водяной коллектор, 4 - изоляционная пластинка, 5 и 6 - изоляторы, 7 - изоляционное кольцо, 8 - крепящий диск.

Тиристор I завинчивается во втулку 2, которая установлена в водяном коллекторе 3 посредством изоляционной пластинки 4. Втулка 2, водяной коллектор 3 и пластинка 4 скреплены между собой эпоксидным клеем; изолятор 5 пред-

ставляет собой кольцевую щель, заполненную спрессованным порошком окиси магния с небольшой добавкой разведенной на ацетоне эпоксидной смолы ЭД-5. После сушки запрессованной массы изготовляется изолятор 6 путем заливки эпоксидным клеем соответствующего пространства. В слое клея находится изоляционное кольцо 7. Для предотвращения электрического поверхностного пробоя между втулкой 2 и коллектором 3 в крепящем диске 8 сделана выточка.

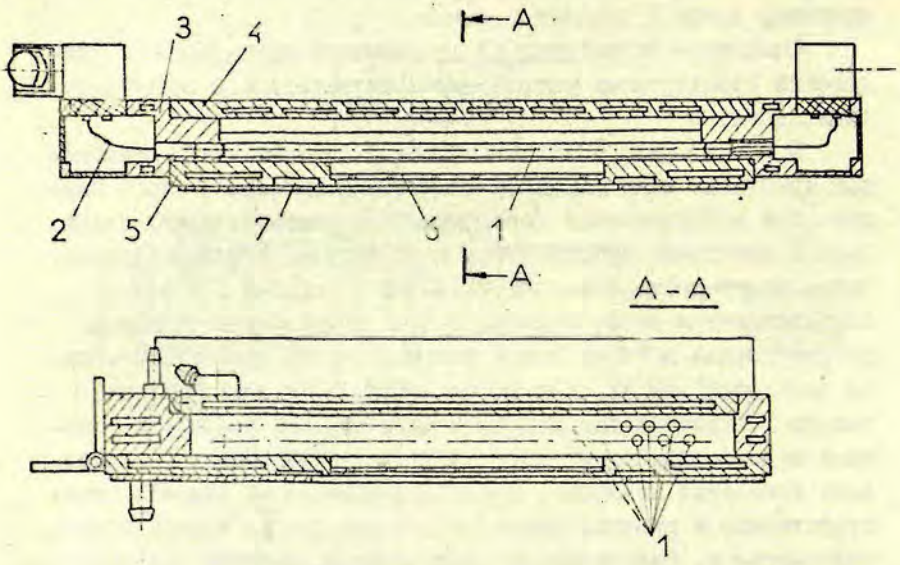
Описанное устройство по способности охлаждения равноценно стандартному устройству принудительного воздушного охлаждения.

Нагреватель (рис.4) состоит из 63 кварцевых трубчатых ламп марки КИ 220х2000 мощностью 5 квт каждая (при использовании форсированного режима). Лампы двумя рядами заполняют пространство нагревателя, образуя радиационную поверхность размером 42х42 см с отдачей 100 вт/см^2. Для достижения столь высокой отдачи тепла лампы сближены до расстояния в 14 мм между центрами нитей накаливания. Концы ламп выведены из рамы нагревателя, а их электрические выводы 2, кончающиеся гибким медным жгутом, соединены с шинами тиристоров. Жгуты прикреплены к молибденовым выводам ламп никелевым желобкам, приваренным точечной сваркой непосредственно к выводам ламп. Такая конструкция обладает компактностью и, как показала эксплуатация, высокой надежностью электрических контактов. Рама нагревателя 3, верхний 4 и нижний 5 отражатели охлаждаются водой. В нижнем отражателе закреплено защитное тонкое кварцевое стекло 6, а снизу отражателя находится отсасывающий коллектор 7; все это необходимо для защиты нагревателя от загрязнения при горении испытуемого образца.

Корпусы кварцевых ламп охлаждаются комбинированным способом – теплопроводностью газа в нагревателе и излучением на покрытие отражателей.

Охладитель стенда (рис.2) должен обеспечить охлаждение испытуемого образца до температуры 600°К со скоростью охлаждения этого образца в окружении абсолютно черного тела с температурой 77°К. Перед испытанием нагреваемая поверхность (если образец прозрачен для излучения нагрева-

теля) покрывается поглощающим слоем. Покрытие изменяет первоначальный интегральный коэффициент черноты образца, и охлаждение не будет соответствовать заданному режиму при из-

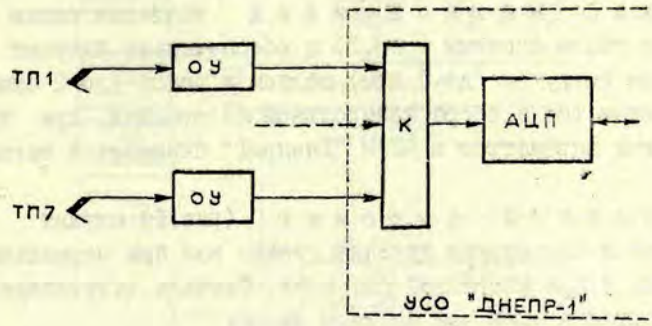


Р и с. 4. Нагреватель испытуемого образца.

I - кварцевые лампы, 2 - вывод лампы, 3 - рама нагревателя, 4 и 5 отражатели, 6 - защитное кварцевое стекло, 7 - отсасывающий коллектор.

лучении на абсолютно черное тело. Если для охлаждения использовать обдув, то возникает опасность порчи слоя и, кроме того, трудно обеспечить равномерность охлаждения по поверхности образца. Поэтому с целью управления процессом охлаждения разработано устройство, позволяющее многократно изменять степень черноты поверхности охладителя, поглощающей излучение образца. Степень черноты регулируется программно.

Устройство измерения температуры (рис.2). Измерение и контроль температуры в НИС "GUNDEGA-2" производится по семи каналам. Каждый канал (рис.5) состоит из термопары, операционного усилителя на основе интегральной схемы "Регата" и аналого-цифрового преобразователя (АЦП), входящего в комплект ЭВУМ "Днепр-I".



Р и с. 5. Температурные каналы.
ТП-ТП7 - термопары, ОУ - операционный усилитель, К - коммутатор каналов ЭВУМ "Днепр-I".

Применение схемы "Регата" показало ее высокую надежность и хорошие усилительные качества. Относительная погрешность измерения составляет не более I%.

Опрос каждой из термопар происходит с частотой I6 раз в секунду. Показания усредняются, линеаризуются и приводятся к шкале температур Цельсия.

Блок электронных ключей (рис. 2). Управление воздушными и водяными потоками, а также механическими приводами в стенде осуществляется 2I тиристорным ключом, каждый из которых включает или выключает электромагнит соответствующего вентиля в момент перехода тока питания через нуль. Это свело помехи в каналах управления к минимуму. Включение всех тиристоров системы осуществляется с помощью блока релейных выходов, входящего в комплект ЭВУМ "Днепр-I".

Управление испытательным стендом заключается в выхо-

де на определенные безаварийные состояния системы: готовности, нагрева, охлаждения, а также переходные состояния. В состоянии нагрева, например, функционирует нагреватель, образец находится в камере нагрева, охлаждаются определенные элементы испытательного стенда, отсасываются газы горящего образца и т.п. Машина контролирует этот процесс и при выходе параметров процесса за уставки немедленно сообщает об этом оператору, а сама пытается ликвидировать аварию.

Шкаф магистралей является одним из основных узлов системы (рис.I) и обеспечивает питание стенда сжатым воздухом (до 6 атм), гелием и водой (до 2 атм), в нем имеются точки сбора контрольной информации, при выходе за уставки параметров в ЭВУМ "Днепр-I" передается сигнал аварии.

Силовой автомат (рис.I) служит для включения и выключения питания стенда как при нормальной работе, так и при аварийной ситуации. Сигналы отсутствия питания подаются в ЭВУМ как сигналы аварии.

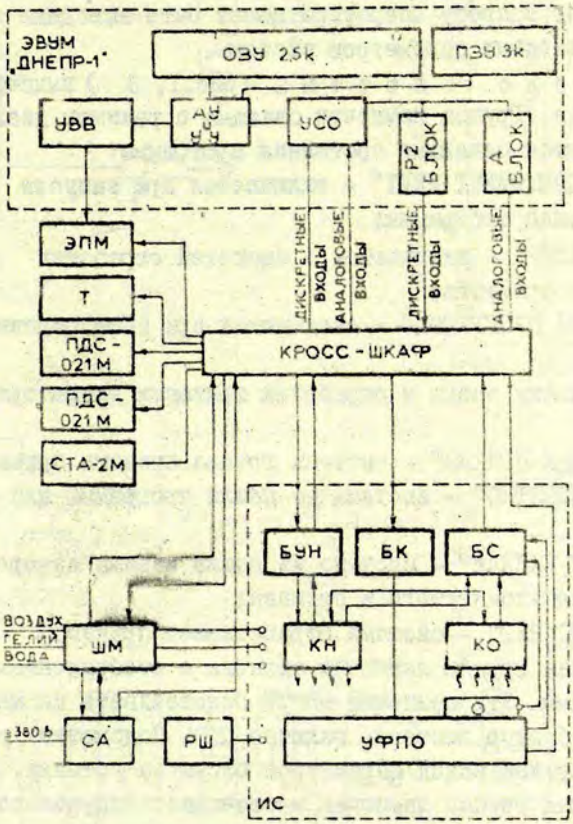
Распределительный шкаф (рис.I) предусмотрен для разводки трех фаз 380 и 320 квт по 3 группам тиристоров нагревателя.

Кросс-шкаф (рис.I) является местом централизации всех каналов НИС "GUNDEGA-2" и других нестандартных для ЭВУМ элементов, с одной стороны, и всех каналов ЭВУМ, с другой стороны; он необходим для проверки всех элементов НИС, а также для удобства сборки системы и ее отладки.

Электронная вычислительная управляющая машина (рис.I) "Днепр-I" средней комплектации содержит, кроме основных блоков (УСО, УАУ, УВВ, ПЗУ, 5 блоков ОЗУ по 512 ячеек каждый), блок релейных выходов и блок аналоговых выходов.

В блоке УВВ дополнительно смонтированы дешифратор и контактная группа, что позволяет, используя систему прерывания, вводить цифро-буквенную информацию с клавиатуры ЭПМ "Soemtron" в ОЗУ "Днепр-I".

Двухкоординатные самопишущие потенциометры ПДС-021М подключены (2 шт.) к четырем каналам А-блока и двум каналам Р-блока. На эти



Р и с. 6. Блок-схема НИС "GUNDEGA-2".

ОЗУ – оперативное запоминающее устройство; ПЗУ – пассивное запоминающее устройство; УВВ – устройство ввода-вывода; АУ и УУ – устройство арифметики и управления; УСО – устройство связи с объектом; Р2 блок – блок релейных выходов; А блок – блок аналоговых выходов; ЭПМ – печатающее устройство; Т – табло; ПДС-0,2IМ – самопишущее устройство; СТА-2М – перфоратор; БУН – блок управления нагревателем; БК – блок ключей; БС – блок согласования; КН – камера нагрева; КО – камера охлаждения; О – объект; УФПО – устройство фиксации и перемещения объекта; ШМ – шкаф магистралей; СА – силовой автомат; РШ – распределительный шкаф; ИС – испытательный стенд.

приборы по запросу оператора может быть выведена любая пара из всех входных параметров системы.

Т а б л о с и с т е м ы (рис.I, 3) выдает два вида информации. Группа лампочек связана с режимом диалога. Сигнализируются основные состояния программы:

"ЭКСПЕРИМЕНТ ИДЕТ" - включается при запуске основной регулирующей программы;

"ТАЙМЕР" - включается с частотой отработки основного цикла эксперимента;

"СБОЙ ПРОГРАММЫ" - включается при сбое основных модулей.

Динамику ввода и отработки приказов оператора отображают:

"ВВОДИ ПРИКАЗ" - система готова принять приказ;

"НЕПОНЯТНО" - система не нашла программы для отработки приказа;

"НЕТ КАНАЛА" - система не нашла канала измерения или управления упомянутого в приказе;

"ВЫПОЛНЯЮ" - система отрабатывает приказ.

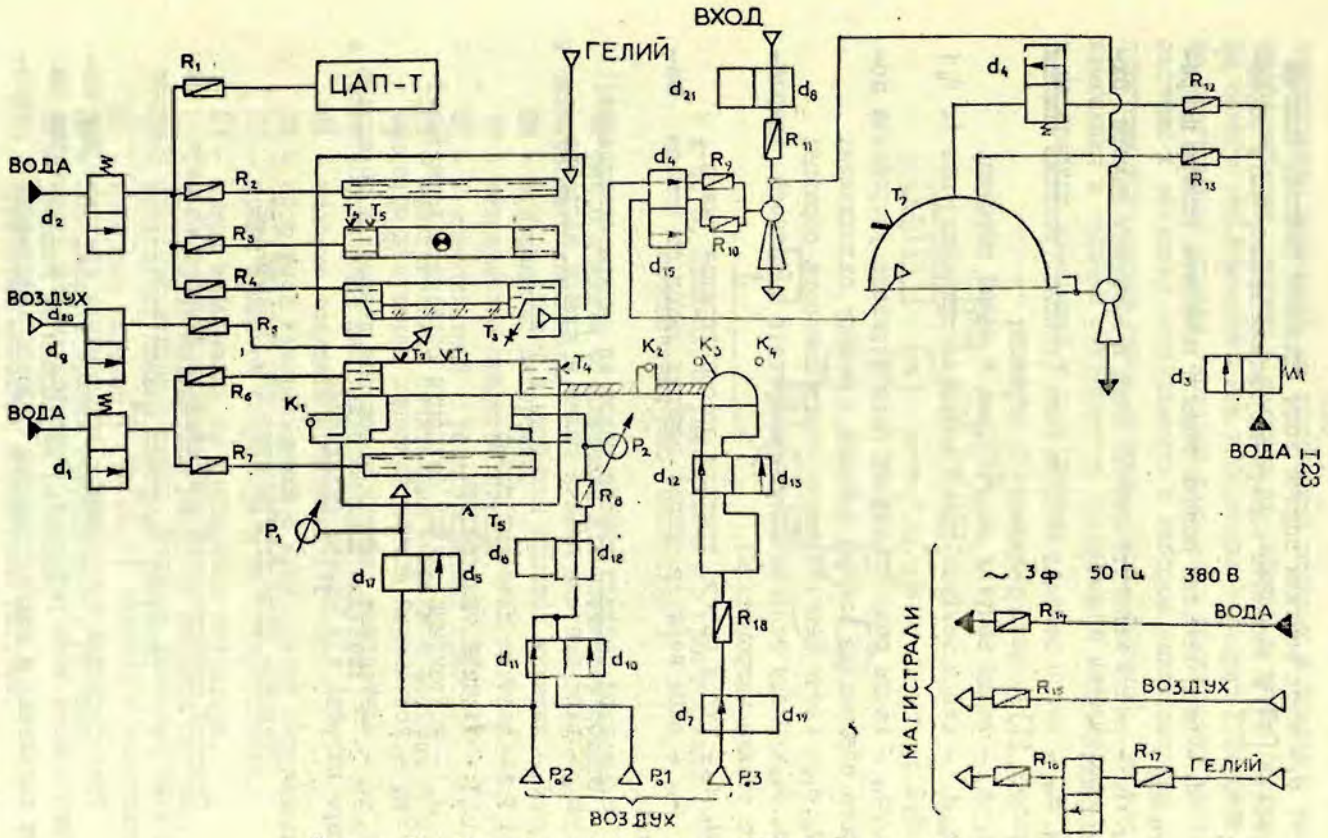
Другая группа лампочек связана с отображением состояния системы. Эти лампочки могут подсвечивать на мнемосхеме любой из блоков системы, включая ЭВУМ. Подсветка происходит, если обнаружен выход параметров блока за уставки.

Третья группа лампочек высвечивает текущее состояние всех выходных каналов системы, кроме регулятора мощности.

Структура НИС "GUNDEGA-2"

На рис.6 показана структурная блок-схема НИС "GUNDEGA-2". Местом сопряжения экспериментального оборудования ЭВУМ "Днепр-I" является кросс-шкаф, на который выведены все внешние каналы "Днепр-I", подведены все измерительные и управляющие каналы экспериментального стенда, а также ряд нестандартных для "Днепр-I" приборов. Более детальная информация по указанным выше каналам приведена на рис.7. Назначение элементов, указанных в схеме, следующее:

ЦАП-Т - цифро-аналоговый преобразователь для управления тиристорами нагревателя;



Р и с. 7. Функциональная схема НИС "GUNDEGA-2".

d_1 – вход вентиля подачи воды (типовой вентиль описан в работе [I]) в элементы камеры-держателя испытуемого образца;

d_2 – вход вентиля подачи воды в элементы камеры нагревателя;

d_3, d_4 – входы вентилей подачи воды в элементы камеры охлаждения;

d_5, d_{17} – входы вентиля подачи газа (типовой вентиль описан в работе [I]) в камеру-держатель образца;

d_6, d_{18} – входы вентиля подачи газа в полый образец;

d_{101}, d_{1} – входы вентиля смены величины давления газа (с $P_0 1$ на $P_0 2$);

d_7, d_{19} – входы общего вентиля пневмопривода устройства поворота образца из камеры нагрева в камеру охлаждения;

d_{12}, d_{13} – входы вентиля направления поворота образца;

d_9, d_{20} – входы вентиля обдува нагретого защитного кварцевого стекла нагревателя;

d_8, d_{21} – входы общего вентиля отсоса горящих газов;

d_{14} – вход вентиля отсоса горящих газов из камеры нагрева;

d_{15} – вход вентиля отсоса газов из камеры охлаждения;

d_{16} – вход вентиля подачи гелия в камеру нагрева;

$R_1, \ldots, R_{18}$ – датчики потоков жидкости и газов;

$P_0 1, P_0 2$ – датчики давления газа;

$T_1, \ldots, T_7$ – датчики температуры;

K_1 – контактный датчик фиксации образца в держателе;

K_2 – контактный датчик фиксации держателя образца;

K_3, K_4 – контактные датчики положения держателя образца в камере нагрева и в камере охлаждения.

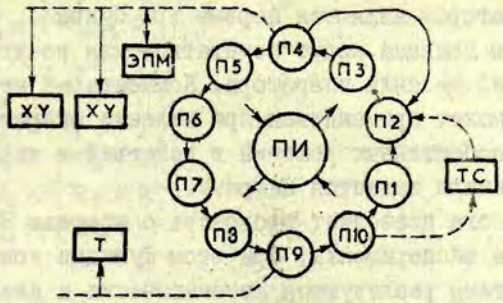
Принципы функционирования ИИС "GUNDEGA-2"

Основой программного обеспечения системы является набор универсальных модулей, общие принципы создания которых рассмотрены в статьях [3,4,5,6,7]*.

Разработанное программное обеспечение пригодно для широкого класса экспериментов. Универсальность достигнута за счет выделения и стандартизации формата информации, непо-

* См. также настоящий сборник, с. 3.

средственно зависящей от количества входных и выходных каналов системы, логики и режимов их работы, а также от графа самого эксперимента. Эта информация образует интерфейс на программном уровне, который связывает техническую часть системы и методику эксперимента с набором универсальных программ управления экспериментом. Таким образом, всякие изменения в системе учитываются программным интерфейсом и не требуют изменений в программах.



Р и с. 8. Блок-схема универсального программного обеспечения НИС "GUNDEGA-2".

На рис.8 приняты обозначения для элементов общего и технического характера:

ПИ – программный интерфейс,
ТС – технические средства стенда,
ХУ – 2-координатный регистрирующий прибор,
ЭПМ – электрическая печатающая машинка с режимом ввода,
Т – табло;

а также обозначения для универсальных модулей:

П1 – задание текущего состояния системы,
П2 – сбор всей измерительной и контрольной информации,
П3 – прием приказов оператора и их интерпретация,
П4 – вывод информации по требованию оператора,
П5 – вычисление параметров выходных каналов системы,
П6 – вычисления правил контроля системы,
П7 – контроль системы,
П8 – определение причин неисправности системы,
П9 – реакция на аварийные ситуации,
П10 – отработка выходных параметров системы.

Управление системой ведется в режиме диалога, с помощью специального языка, но это не требует знаний в области программирования. Операторский приказ, обрабатываемый в режиме диалога, имеет вид:

⟨ПРИКАЗ⟩ :: = ⟨ИДЕНТИФИКАТОР⟩ ⟨КОММЕНТАРИЙ⟩ ␣ ␣ ⟨НОМЕР КАНАЛА⟩ ⟨КОММЕНТАРИЙ⟩ ␣ ␣ ⟨ЧИСЛО⟩ ⟨КОММЕНТАРИЙ⟩ +

Идентификатором являются первые три буквы сообщения, остальные буквы приказа могут печататься для понятности или не печататься по желанию оператора. Комментарий не анализируется ЭВУМ и может применяться при желании оператора для пояснения его собственных решений и действий в ходе эксперимента. Номер канала задается цифрами.

Режим диалога позволяет проводить с помощью НИС "GUNDEGA-2" поисковые эксперименты, при этом функции контроля за состоянием системы реализуются автоматически и независимо от диалога.

Значения релейных каналов НИС "GUNDEGA-2" вычисляются универсальным модулем [illegible] в соответствии логическими функциями, которые должны быть заданы на стадии описания системы. Таким образом, вместо составления программ для каналов управления системы необходимо лишь описать в виде таблицы логику работы этих каналов. Любые изменения в системе вызывают изменения только в таблицах (дополнение, изменение, удаление строк).

Функционирование НИС "GUNDEGA-2" в данном варианте описывается следующим набором функций:

$$d_1 = (\beta \equiv 1) \vee (\beta \equiv 2) \vee (\beta \equiv 3),$$
$$d_2 = d_1,$$
$$d_3 = (\beta \equiv 3),$$
$$d_4 = (T_1 > T_1^\circ) \wedge d_3,$$
$$d_5 = (\beta \equiv 0),$$
$$d_6 = \bar{d}_{18} \, 1 \, (R_8 < R_8^\circ),$$

$d_7 = [d_{12} \wedge (R_{18} < R_{18}^{\circ})] \vee [d_{13} \wedge (R_{18} < R_{13}^{\circ})]$,
$d_8 = (d_{14} \vee d_{15}) \wedge (R_{11} < R_{11}^{\circ}) \wedge (\beta \neq 0)$,
$d_9 = d_3 \wedge (R_5 < R_5^{\circ})$,
$d_{10} = 0$,
$d_{11} = d_{17} \vee d_{18}$,
$d_{12} = \bar{d}_3 \wedge (K_3 < K_3^{\circ})$,
$d_{13} = d_3 \wedge (K_4 < K_4^{\circ})$,
$d_{14} = \bar{d}_3 \wedge (R_9 < R_9^{\circ})$,
$d_{15} = d_3 \wedge (R_{10} < R_{10}^{\circ})$,
$d_{16} = d_1 \wedge (T_3 > T_3^{\circ})$,
$d_{17} = d_1 \wedge (P_1 > P_1^{\circ})$,
$d_{18} = d_1 \wedge (P_2 > P_2^{\circ})$,
$d_{19} = \bar{d}_7 \wedge (R_{18} > R_{18}^{\circ})$,
$d_{20} = \bar{d}_3 \wedge (R_5 > R_5^{\circ})$,
$d_{21} = \bar{d}_8 \wedge (\beta \equiv 0) \wedge (R_{11} > R_{11}^{\circ})$.

Здесь приняты следующие обозначения: d – релейный канал, β – индекс состояния системы, T – сигнал датчика температуры, P – сигнал датчика давления, R – сигнал датчика потока, K – сигнал контактного датчика, T°, P°, R°, K° – уставки.

Эти функции отрабатываются последовательно сверху вниз. Полученные значения помещаются в таблицу выходов системы. Специальный модуль отработки выходов в соответствии с таблицей (или алгоритмом) соответствия d -номеров каналов с их реальными физическими адресами включает (d = I) или выключает (d = 0) вентили "GUNDEGA-2".

Однако отработка выходов зависит не только от приведенной выше таблицы, но и от результатов контроля состояния НИС "GUNDEGA-2".

Контроль системы осуществляется специальными универсальными модулями [5,6]. Для контроля используются три таблицы, которые должны быть занесены на стадии описания системы: таблица входов системы, таблица уставок для входных переменных и таблица вычисления правил контроля системы.

Таблица I (входы)	Таблица 2 (уставки)	Таблица 3 (правила)
R_1	$\alpha_1\ R_1^\circ$	$\alpha_1 = d_2$
R_2	$\alpha_2\ R_2^\circ$	$\alpha_2 = d_2$
R_3	$\alpha_3\ R_3^\circ$	$\alpha_3 = d_2$
R_4	$\alpha_4\ R_4^\circ$	$\alpha_4 = d_2$
R_5	$\alpha_5\ R_5^\circ$	$\alpha_5 - (\beta \equiv 3)$
R_6	$\alpha_6\ R_6^\circ$	$\alpha_6 = d_1$
R_7	$\alpha_7\ R_7^\circ$	$\alpha_7 = d_1$
R_8	$\alpha_8\ R_8^\circ$	$\alpha_8 = 0$
R_9	$\alpha_9\ R_9^\circ$	$\alpha_9 = 0$
R_{10}	$\alpha_{10}\ R_{10}^\circ$	$\alpha_{10} = 0$
R_{11}	$\alpha_{11}\ R_{11}^\circ$	$\alpha_{11} = 0$
R_{12}	$\alpha_{12}\ R_{12}^\circ$	$\alpha_{12} = d_4 \wedge d_3$
R_{13}	$\alpha_{13}\ R_{13}^\circ$	$\alpha_{13} = d_3$
R_{14}	$\alpha_{14}\ R_{14}^\circ$	$\alpha_{14} = 2$
R_{15}	$\alpha_{15}\ R_{15}^\circ$	$\alpha_{15} = 2$
R_{16}	$\alpha_{16}\ R_{16}^\circ$	$\alpha_{16} = d_{16}$
R_{17}	$\alpha_{17}\ R_{17}^\circ$	$\alpha_{17} = 2$
R_{18}	$\alpha_{18}\ R_{18}^\circ$	$\alpha_{18} = 2$
T_1	$\alpha_{19}\ T_1^\circ$	$\alpha_{19} = 1$
T_2	$\alpha_{20}\ T_2^\circ$	$\alpha_{20} = 1$
T_3	$\alpha_{21}\ T_3^\circ$	$\alpha_{21} = 1$
T_4	$\alpha_{22}\ T_4^\circ$	$\alpha_{22} = 1$
T_5	$\alpha_{23}\ T_5^\circ$	$\alpha_{23} = 1$
T_6	$\alpha_{24}\ T_6^\circ$	$\alpha_{24} = 1$
P_1	$\alpha_{25}\ P_1^\circ$	$\alpha_{25} = 1$
P_2	$\alpha_{26}\ P_2^\circ$	$\alpha_{26} = 1$
K_1	$\alpha_{27}\ K_1^\circ$	$\alpha_{27} = 2$
K_2	$\alpha_{28}\ K_2^\circ$	$\alpha_{28} = 2$
K_3	$\alpha_{29}\ K_3^\circ$	$\alpha_{29} = (\beta \neq 3)$
...	...	...

3-я таблица отрабатывается модулем, определяющим текущие правила контроля системы; как видим, они зависят от выходных каналов системы, состояния системы, а также могут быть фиксированы для данного эксперимента. Вычисленный индекс правила контроля системы α заносится во 2-ю таблицу. Значение индекса определяется по 3-й таблице с учетом динамики аргументов, Например, для трех тактов времени индекс $\alpha(t)$, зависящий от одного аргумента $d(t)$, вычисляется по формуле

$$\alpha(t)=\begin{cases}2, & d(t)\wedge d(t-1)\wedge d(t-2)=1,\\ 1, & d(t)\vee d(t-1)\vee d(t-2)=0\\ 0 & \text{в остальных случаях.}\end{cases}$$

Такой подход позволяет избежать появления "псевдоаварий", возникающих при переходных процессах в каналах системы.

Модуль контроля [5] попарно сравнивает I-ю таблицу с уставками 2-й таблицы, имея в виду, что при $\alpha = 2$ параметр должен быть больше уставки, при $\alpha = 0$ результат сравнения не формируется. Все отклонения от уставок запоминаются в виде аварийного сообщения

$$S_a(\delta_1, \delta_2, \dots, \delta_k, \dots, \delta_n),$$

где δ_k – индекс отклонения параметра канала от уставки.

Принято, что $\delta_k = 1$, если обнаружено отклонение, в противном случае $\delta_k = 0$.

Далее S_a сравнивается с набором возможных аварийных ситуаций в каждом узле НИС и вырабатывается сигнал аварийности этого узла. Этот сигнал отображается на мнемосхеме 3 системы "GUNDEGA-2". Таблица определения аварийности узлов следующая:

$$g_1 = \delta_1 \vee \delta_2 \vee \delta_3 \vee \delta_4 \vee \delta_5 \vee \delta_{22} \vee \delta_{23}$$
$$g_2 = \delta_6 \vee \delta_7 \vee \delta_{19} \vee \delta_{20} \vee \delta_{23} \vee \delta_{25} \vee \delta_{26} \vee \delta_{27} \vee \delta_{28}$$
$$g_3 = \delta_{18} \vee \delta_{29} \vee \delta_{30}$$
$$g_4 = \delta_{12} \vee \delta_{13} \vee \delta_{24}$$
$$g_5 = \delta_{14} \vee \delta_{15} \vee \delta_{17}$$

где $g_1, \dots, g_5$ – сигналы аварийности соответствующих блоков: блок нагрева, камера с образцом, механизм поворота, камера охлаждения, магистрали. Кроме этих сигналов, выдаются

сообщения о результатах диагностики ЭВУМ "Днепр-I" (микротест во время эксперимента) и проверки некоторых вычислительных процедур.

ЛИТЕРАТУРА

I. Зариньш М.Я. Системный канал управления мощным тепловым потоком. – Уч.зап.ЛГУ им.П.Стучки, т.I96. Кибернетизация научного эксперимента, вып.5. Рига. I973, с.99–I05.

2. Страумен Я.Я., Паспарне А.К. Дискретные каналы научно-исследовательской системы для управления подачей жидкости и газов. – Уч.зап.ЛГУ им.П.Стучки, т.I96. Кибернетизация научного эксперимента, вып.5. Рига, I973, с.I06–III.

3. Кузьмин Ю.Я., Гвоздев С.В., Банга А.Я., Котомин Е.А. Особенности режима диалога в автоматизированном эксперименте. – Уч.зап.ЛГУ им.П.Стучки, т.I96. Кибернетизация научного эксперимента, вып.5. Рига, I973, с.32–49.

4. Кузьмин Ю.Я. Об одной альтернативе программирования эксперимента. – Уч.зап. ЛГУ им.П.Стучки, т.I96. Кибернетизация научного эксперимента, вып.5, Рига,I973, с.3–I6.

5. Гвоздев С.В. К реализации группового алгоритма сбора и контроля измерительной информации. – Уч.зап.ЛГУ им. П.Стучки, т.I96. Кибернетизация научного эксперимента, вып. 5. Рига, I973, с.I7–26.

6. Кузьмин Ю.Я., Цирулис Я.П. Способ формирования закона контроля в автоматизированной системе. – Уч.зап. ЛГУ им.П.Стучки, т.I96. Кибернетизация научного эксперимента, вып.5. Рига, I973, с.27–3I.

7. Кузьмин Ю.Я. Групповой алгоритм управления каналами научно-исследовательской системы. – Уч.зап.ЛГУ им.П.Стучки, т.I96. Кибернетизация научного эксперимента, вып.5. Рига, I973, с.55–59.

УДК 007.5.001.3:62

А.Я.Банга

КОНФИГУРАЦИЯ ЭВУМ М-6000 В СИСТЕМЕ ДИАГНОСТИРОВАНИЯ АВТОМОБИЛЕЙ

Рассматриваются особенности системы диагностирования автомобилей с ЭВУМ в качестве центрального звена, вопросы выбора ЭВУМ и ее конфигурации.

Система диагностирования автомобилей (СДА), использующая в качестве центрального звена электронно-вычислительную управляющую машину (ЭВУМ), является следующей ступенью развития систем диагностирования по сравнению с системами, где функции управления, сбора и обработки распределены между автоматом с жесткой структурой и операторами системы (например, станция диагностирования автомобилей СДА-70). В системе диагностирования ЭВУМ выполняет следующие функции:

- управляет процессом диагностирования;
- обеспечивает взаимодействие СДА и операторов;
- выдает управляющие воздействия на исполнительные элементы СДА;
- собирает информацию с датчиков диагностических параметров;
- обрабатывает диагностическую информацию;
- выводит результаты диагностирования.

Преимущества машинно-управляемой СДА следующие:

- автоматизирован процесс измерения значений диагностических параметров (за исключением некоторых, о чем пойдет речь ниже) и сбор измерительной информации;
- автоматизирована выдача большей части управляющих воздействий;
- более удобны формы взаимодействия операторов и системы;
- возможно накопление результатов диагностирования за длительный срок и статистический анализ этой информации, что позволяет использовать такую систему в качестве низшего уров-

ня АСУ соответствующего ведомства.

Таким образом, с применением ЭВУМ уменьшается трудоемкость диагностирования, повышается пропускная способность СДА (при наличии в автомобиле системы встроенных датчиков это еще более спутимо [1]) и качество диагностической информации .

Для системы диагностирования, создаваемой в виде базовой для дальнейших исследований процесса диагностирования, развития методики и средств диагностики, важно иметь следующие возможности: во-первых, менять и развивать алгоритмы диагностирования, что крайне трудно или часто невозможно в системе, управляемой автоматом с жесткой структурой; во-вторых, развивать технические средства системы – датчики, исполнительные элементы и т.п. Этим определяются требования при выборе типа ЭВУМ. Для решения такой задачи больше всего подходит мини-ЭВМ с модульной структурой, позволяющей по мере необходимости наращивать количество разнообразных каналов, а также емкость оперативного запоминающего устройства, и включать ее в многоуровневую систему в качестве нижнего звена. Эта ЭВУМ должна располагать достаточно широким набором модулей связи с объектом, обеспечивающим:

– ввод в машину дискретных, частотных и аналоговых сигналов, разнотипных по электрическим и временным характеристикам ввиду большого разнообразия датчиков диагностических параметров, так как унификация этих сигналов требует создания большого количества нестандартных согласователей;

– вывод управляющих воздействий в виде дискретных и аналоговых сигналов;

– изменение структуры устройства связи с объектом (УСО) соответственно возможным изменениям в наборе датчиков и исполнительных элементов.

Из отечественных ЭВУМ этим требованиям наиболее полно соответствует М-6000, краткие характеристики которой следующие:

– быстродействие до $2 \cdot 10^5$ адресных и до $1{,}8 \cdot 10^6$ безадресных операций в секунду;

– диапазон наращивания оперативной памяти от 4К до 32К двухбайтовых слов;

- максимальное количество подключаемых внешних устройст
54;

- выход на сопряжение с внешними устройствами - типа

ЭВУМ М-6000 имеет широкий набор устройств ввода-вывода: устройство ввода с перфоленты, устройство вывода на перфоленту, устройство ввода-вывода на базе телетайпа, устройство ввода-вывода на базе пишущей машинки "Консул-260", устройство привязки осциллографа для вывода графической информации и другие. Некоторые из упомянутых устройств объединены в группы с целью упрощения связи оператора с машиной и минимизации аппаратуры.

Модули связи ЭВУМ с объектом требуют более подробного рассмотрения. Имеются три группы этих модулей:

- модули ввода аналоговой информации,
- модули ввода дискретной информации,
- модули вывода дискретной информации.

В первую группу входят следующие модули:

- быстродействующий аналого-цифровой преобразователь (АЦП) двух модификаций: с заземленным и с изолированным входом; имеет 4 входа;
- интегрирующий аналого-цифровой преобразователь (АЦП-И) на один вход;
- нормирующие усилители (УН) для приведения сигналов милливольтового уровня к среднему уровню;
- коммутатор сигналов низкого уровня (КСНУ) на 16 входов;
- коммутатор сигналов среднего уровня (КССУ) на 16 входов;
- модуль управления коммутаторами (МУК), вместе с расширителем управления коммутаторами (РУК), обеспечивающий управление 16 коммутаторами КССУ или КСНУ;
- модули фильтров.

Перечисленные модули позволяют строить одно- и двухступенчатые однополюсные и двухполюсные схемы ввода аналоговых сигналов.

Для ввода дискретных сигналов служат следующие модули:

- модуль ввода дискретной информации (МВвДИ) для приема информации от 16 датчиков;
- модуль ввода инициативных сигналов (МВвИС), имеющий 8 входных каналов для ввода дискретных сигналов и выдающий сигнал запроса на обслуживание при изменении состояния датчиков;

– модуль ввода число-импульсных сигналов (МВвЧИС), имеет один вход и накапливает до 4096 импульсных сигналов, поступающих с частотой до 200 кгц;

– модуль группового управления вводом дискретной информации (МГУ), обеспечивающий вместе с расширителем подключение к одному выходу на сопряжение 2К до 22 модулей МВвДИ, МВвИС, МВвЧИС.

Модули МВвДИ, МВвИС, МВвЧИС имеют I4 модификаций для сигналов различных уровней и полярности, с фильтрацией входного сигнала или без этого.

Вывод дискретных сигналов обеспечивается следующими модулями:

– бесконтактный модуль кодового управления (МКУБ) трех разновидностей, отличающихся по допустимым значениям коммутируемого тока и напряжения; имеет I0 выходов;

– контактный модуль кодового управления (МКУК) на I0 выходов;

– модуль позиционного управления (МПУ) на 32 выхода;

– модуль импульсного управления (МИУ) на 5 выходов, выдающий сигнал фиксированной длительности;

– контактный модуль переключения (МПК) на 28 выходов;

– модуль группового управления выводом дискретной информации (МУВ), обеспечивающий управление до 64 перечисленных выше модулей.

Недостатком данного набора модулей связи с объектом является отсутствие модулей для вывода аналоговых сигналов.

В настоящее время совместными усилиями Министерства автомобильного транспорта и шоссейных дорог Латвийской ССР и Проблемной лаборатории физики полупроводников ЛГУ создается система диагностирования автомобилей путем сопряжения станции диагностирования СДА-70 с ЭВУМ М-6000. Диагностические параметры, исследуемые на станции СДА-70, можно разделить на две группы: параметры, вводимые в ЭВУМ непосредственно через УСО, и параметры, для ввода которых в ЭВУМ требуется участие оператора. К последним относятся, во-первых, результаты визуальных проверок, для ввода которых требуется переносной пульт оператора, во-вторых, исследования стуков в двигателе и системы зажигания, представляющие собой

анализ колебательных процессов высокой частоты. При непосредственном вводе соответствующих сигналов через УСО для запоминания до $4 \cdot 10^4$ мгновенных значений сигнала с частотой измерения свыше 100 кгц требуется расширение емкости ОЗУ до 40К только лишь для этой цели, но это превышает возможности ЭВУМ М-6000; быстродействие АЦП комплекса М-6000 также недостаточно для такой частоты опроса. Поэтому целесообразно поручить оператору ввод оценок этих параметров по осциллограммам сигналов, используя устройства ввода-вывода ЭВУМ.

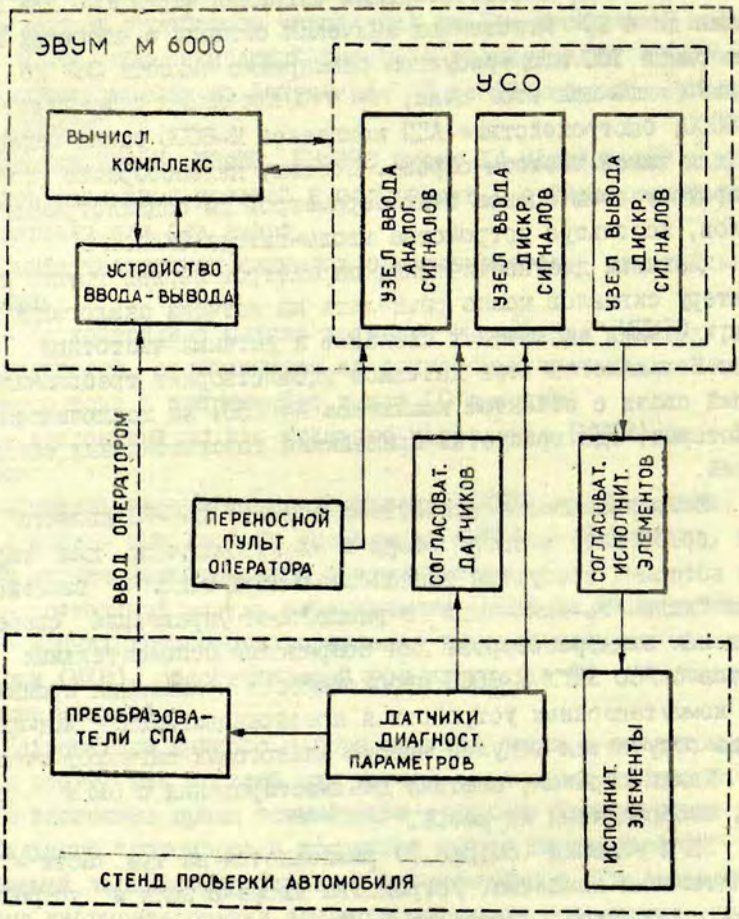
Датчики диагностических параметров первой группы по характеру сигналов можно разделить на датчики аналоговых сигналов, датчики дискретных сигналов и датчики частотных сигналов. Большинство этих датчиков удовлетворяют требованиям модулей связи с объектом комплекса М-6000, за исключением лишь некоторых, где требуется применение дополнительных согласователей.

Исполнительными элементами станции СДА-70 являются мощные (до 30 квт) электромоторы и электромагниты, для управления которыми требуются воздействия двух типов - воздействия типа "включить-выключить" и аналоговое управление скоростью вращения электромоторов. Для сопряжения исполнительных элементов с УСО ЭВУМ М-6000 необходимость применения промежуточных коммутационных устройств и преобразователей код-аналог из-за отсутствия модулей вывода аналоговых сигналов очевидно.

Таким образом, система диагностирования с ЭВУМ имеет вид, изображенный на рис.1.

ЭВУМ условно (см.рис.1) разделяется на три части - вычислительный комплекс, устройства ввода-вывода и устройство связи с объектом, состоящее из узла ввода аналоговых сигналов, узла ввода дискретных сигналов и узла вывода дискретных сигналов.

Конфигурация вычислительного комплекса и устройств ввода-вывода зависит от степени использования стандартного программного обеспечения [2]. Для выполнения задач диагностирования достаточно емкости оперативного запоминающего устройства 8К слов. Наличие арифметического расширителя (РА) обязательно для эффективного использования быстродействия процессора (Пр). Требуемый набор устройств ввода-вывода следующий:



Р и с. I. Блок-схема СДА с ЭВУМ М-6000 в контуре управления.

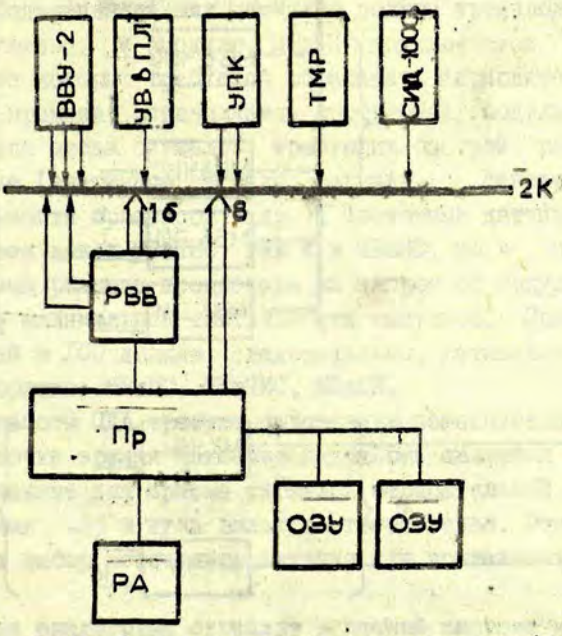
– вводно-выводное устройство ВВУ-2, используемое в качестве центрального пульта ЭВУМ и содержащее пишущую машинку "Консул-260", считыватель с перфоленты СП-4П, ленточный перфоратор;

– устройство ввода с перфоленты (УВвПЛ), выполненное на базе фотосчитывателя FS-150I и обеспечивающее достаточную скорость при вводе программ;

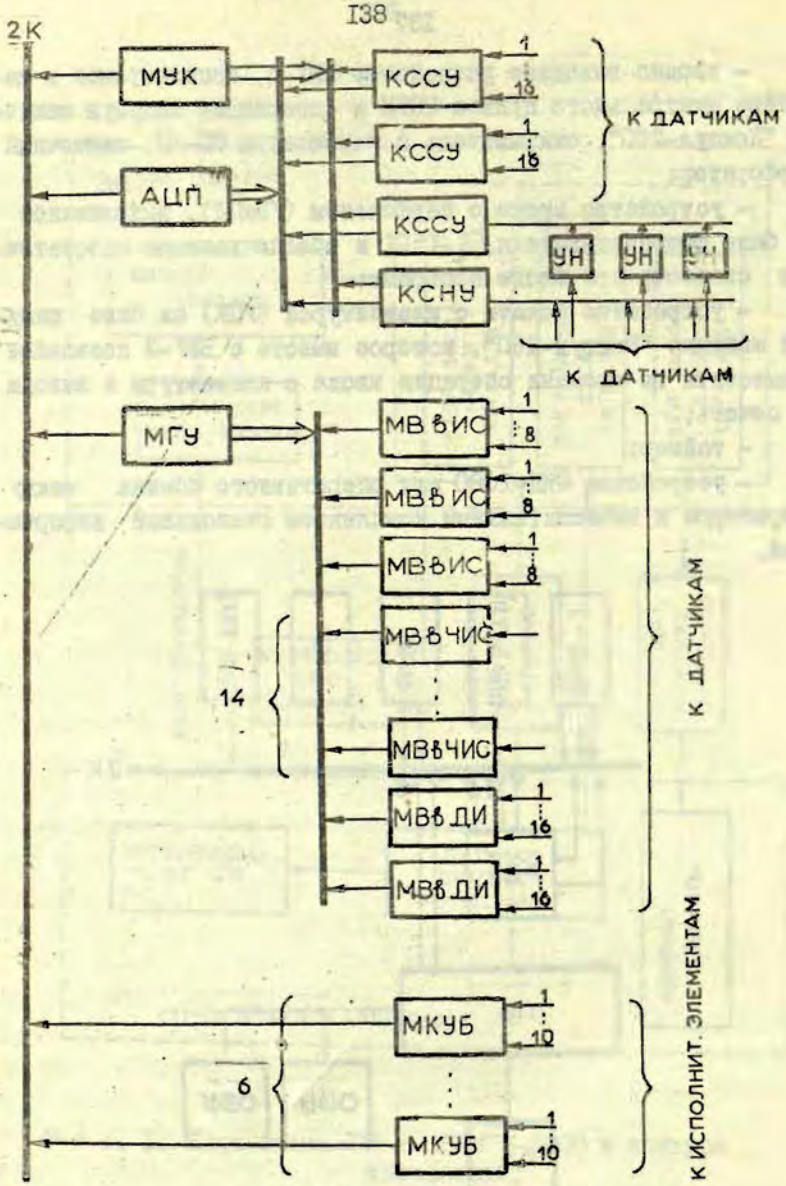
– устройство печати с клавиатурой (УПК) на базе пишущей машинки "Консул-260", которое вместе с ВВУ-2 позволяет совместить во времени операции ввода с клавиатуры и вывода на печать;

– таймер;

– устройство СИД-1000 для оперативного обмена между оператором и вычислительным комплексом символьной информацией.



Р и с. 2. Конфигурация вычислительного комплекса с устройствами ввода-вывода.



Р и с. 3. Конфигурация устройства связи с объектом ЭВУМ М-6000 в СДА.

Как видно из рис.2 , перечисленный набор устройств ввода-вывода занимает 8 выходов на сопряжение 2К, т.е. все выходы процессора. Поэтому для подключения УСО обязательно наличие одного расширителя ввода-вывода (РВВ), увеличивающего количество выходов до 22.

Конфигурация УСО (рис.3) в основном определена структурой объекта управления. Для управления 26 исполнительными механизмами (из них 4 электромотора, требующие аналоговое управление) достаточно 6 модулей МКУБ, подключаемых непосредственно к сопряжению 2К, чем достигается максимальное быстродействие аппаратуры вывода. Таким образом, узел вывода дискретных сигналов занимает 6 выходов на сопряжение 2К.

Узел ввода дискретных сигналов данной системы должен обеспечить ввод сигналов от дискретных и частотных датчиков и переносного пульта оператора с общим количеством двоичных разрядов 70. Подключение этих датчиков должно производиться с учетом следующего: к модулю МВвДИ подключаются датчики, состояние которых требуется опрашивать периодически или в моменты времени, определяемые программой; модуль МВвИС используется для ввода сигналов, требующих быстрой реакции на их изменение (аварийные сигналы; сигналы, фиксирующие отсчет длительности процессов и др.). Частотные датчики могут подключаться как к МВвЧИС, так и к МВвИС, но в последнем случае время реакции процессора на запрос от модуля должно быть меньше минимальной длительности импульса. Приоритеты этих модулей в УСО должны, следовательно, устанавливаться в следующем порядке: МВвИС, МВвЧИС, МВвДИ.

Условия работы СДА требуют повышенную помехоустойчивость узла. С этой точки зрения наиболее надежными являются модули, предназначенные для приема сигналов отрицательной полярности с уровнями -24 и нуль вольт соответственно. Этим определяется и выбор источников питания для контактных датчиков.

Узел ввода аналоговых сигналов в данной системе диагностирования состоит из одноступенчатых коммутаторов по соображениям быстродействия узла и из-за небольшого количества (26) датчиков. Сигналы низкого уровня для исследования электрооборудования автомобиля требуют нормализации, т.е. приве-

дения к среднему уровню и двухполюсной коммутации. Остальные датчики на входе быстродействующего АЦП с изолированным входом коммутируются двумя однополюсными коммутаторами. Потребность в фильтрах должна определяться экспериментально.

Следует отметить, что дальнейшие разработки программного обеспечения системы могут повлиять на конфигурацию ЭВУМ по соображениям удобства обращения и быстродействия программ.

ЛИТЕРАТУРА

1. Электронная система для диагностики неисправностей автомобилей. – "Электроника", 1971, I, с.4–5.

2. Логическая компоновка систем на базе процессора М-6000 АСВТ-М. Руководящий технический материал. НИИУВМ, Северодонецк, 1972. 182 с.

С О Д Е Р Ж А Н И Е

РЕЖИМ ДИАЛОГА В ИССЛЕДОВАНИЯХ

СИСТЕМЫ ЭКСПЕРИМЕНТИРОВАНИЯ

Ученые записки, том 225

КИБЕРНЕТИЗАЦИЯ НАУЧНОГО ЭКСПЕРИМЕНТА
Выпуск 6

Редактор Э.Тарденак
Технический редактор З.Клейнштите
Корректор М.Штерн
Латвийский государственный университет
Рига I975

Подписано к печати 3I.I2.I974 ЯТ2I354 Зак.№264.
Ф/б 60x84/I6. Бумага №I. Физ.п.л. 9,3. Уч.-и.л. 7,0
Тираж 500 экз. Цена 70 к.

Отпечатано на ротапринте, Рига-50, ул.Вейденбаума,5
Латвийский государственный университет им. П.Стучки

Цена 70 к.

Учен. зап. (ЛГУ им. Петра Стучки), 1975, т.225, I-I42