ГОСУДАРСТВЕННЫЙ НАУЧНЫЙ ЦЕНТР РОССИЙСКОЙ ФЕДЕРАЦИИ

ИНСТИТУТ ФИЗИКИ ВЫСОКИХ ЭНЕРГИЙ

ИФВЭ 96–71
ОНФ

А.А.Борисов, В.Н.Горячев., А.С.Кожин, В.В.Липаев

ОРГАНИЗАЦИЯ И ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ СИСТЕМЫ СБОРА ДАННЫХ ДЛЯ ДРЕЙФОВЫХ КАМЕР ИФВЭ НЕЙТРИННОГО ДЕТЕКТОРА ИФВЭ-ОИЯИ

Протвино 1996

УДК 539.1.075 М–24

Аннотация

Борисов А.А. и др. Организация и программное обеспечение системы сбора данных для дрейфовых камер ИФВЭ нейтринного детектора ИФВЭ-ОИЯИ: Препринт ИФВЭ 96–71. – Протвино, 1996. – 22 с., 6 рис., библиогр.: 8.

В работе описана общая организация электроники регистрации данных и *on-line* программное обеспечение системы сбора данных для дрейфовых камер ИФВЭ нейтринного детектора ИФВЭ-ОИЯИ. Дано описание языка символьного анализа и макрокоманд ввода/вывода.

Abstract

Borisov A.A. et al. Data Acquisition System for IHEP Drift Chambers of Neutrino Detector IHEP-JINR: IHEP Preprint 96–71. – Protvino, 1996. – p. 22, figs. 6, refs.: 8.

The paper reports the general electronics implementation and the software data acquisition system for the IHEP drift chambers of the neutrino detector IHEP-JINR. A description of simbol analysing language and input/output macrocomands is represented.

Для определения координат треков заряженных частиц в мишенной части нейтринного детектора ИФВЭ-ОИЯИ (НД) [1] используется система из 358 дрейфовых камер ИФВЭ [2]. Каждая камера имеет размеры $0,5 \cdot 4$ м^2 с дрейфовым промежутком 250 мм и 4 попеременно смещенными относительно центра камеры на 0,75 мм сигнальными проволоками. Смещение позволяет разрешать лево-правую неоднозначность прохождения частицы через камеру и определять для нее угол проекции трека [3]. В НД камеры объединены в 20 модулей, каждый из которых состоит из двух взаимноортогональных плоскостей по 9 камер (в модуле перед детектором электронов плоскости повернуты относительно остальных на $7^o34'$ и содержат по 8 камер). Таким образом, имеется по 20 плоскостей для определения координат X- и Y-проекций треков, а также их углов.

В каждой камере временные отсчеты с сигнальных проволок поступают на входы 4-канального усилителя-формирователя (УФ), откуда передаются на один из 24 входов модуля время-цифрового преобразователя (ВЦП) Р-94 [4], где происходят запоминание времени прихода сигнала в реальном масштабе времени и регистрация номера канала, по которому пришел сигнал. Все это заносится во внутреннюю память ВЦП на 64 сигнала. Измерение временных интервалов основано на счете тактовых импульсов с частотой 50 МгЦ внутренним счетчиком времени дрейфа и запоминанием его содержимого в момент прихода сигнала по одному из входов ВЦП. При этом временное разрешение электроники составляет 20 нс, однако вместе с этим запоминается номер полупериода (0 либо 1) тактового импульса, что позволяет довести разрешение до 10 нс. Максимально возможное регистрируемое время дрейфа составляет 81,92 мкс (12 двоичных разрядов на число полных тактовых импульсов и 1 разряд на номер полупериода).

Всего в состав электроники регистрации данных (ЭРД) входят 64 ВЦП, имеющих одиночную ширину стандарта СУММА и размещенных в 4 каркасах.

1. Аппаратная организация системы сбора данных

В качестве базовой ЭВМ в системе сбора данных (ССД) с ДК ИФВЭ используется ЭВМ СМ-4. Операционной средой для программного обеспечения (ПО) является

система RSX-11M, поддерживающая многозадачный режим в масштабе реального времени. На рис. 1 схематически представлены базовые связи элементов ЭРД и ЭВМ, отображающие основную структуру аппаратной части ССД.

В зависимости от режима работы, запуск ЭРД осуществляется либо от банчевого счетчика, расположенного в начале мишенной части (рабочее считывание), либо от отдельно выделенных сцинтилляционных счетчиков или плоскостей ЖСС [5] (регистрация космических мюонов), либо от импульсного генератора (контроль ЭРД). Одновременно запускается генератор тактовых импульсов и производится отцифровка приходящих отсчетов с сигнальных проволок ДК. Прекращение работы генератора происходит через предварительно установленный временной интервал, после чего выставляется сигнал BD в контроллере КС-30, что приводит к появлению прерывания в ЭВМ. Соответствующее ПО производит последовательное считывание данных с ВЦП со скоростью ≈ 30 мкс/слово. Затем осуществляется общий сброс в исходное состояние всех ВЦП и КС-30, после чего ЭРД вновь готова к регистрации физической информации.

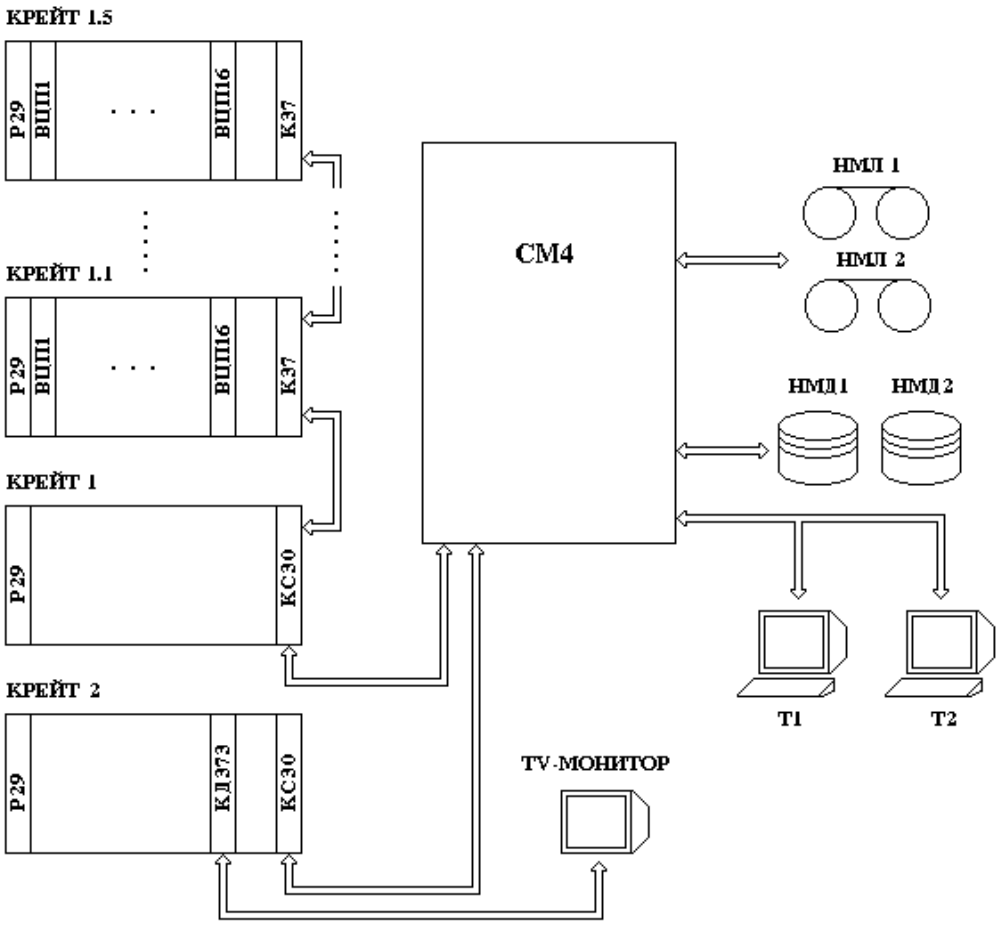

Рис. 1. Блок-схема вычислительных и основных компонентов электроники сбора данных ССД с ДК ИФВЭ.

Комплексная проверка электроники возможна имитацией рабочего режима путем подачи на входы УФ тестовых сигналов. Для этой цели используется генератор Г-50 [6], вырабатывающий по внешнему запуску или по команде от ЭВМ цуг импульсов с частотой 10 МгЦ. С помощью специального коммутатора и системы пассивных разветвителей импульсы цуга распределяются по отдельным ДК. Разветвление сигналов осуществляется таким образом, чтобы на любой из модулей ВЦП не приходили одновременные сигналы. Для проверки дифференциальной нелинейности ВЦП и точного измерения относительной разности времен прохождения сигналов включено два последовательных модуля управляемой от ЭВМ задержки с шагом 1 нс и максимальным временем задержки 128 нс.

2. Система пассивной коммутации ДК и электроники регистрации данных

Наибольшая загрузка камер в НД происходит в области развития адронных каскадов. При этом с каждой проволоки этих камер может поступать до нескольких десятков временных отсчетов, хотя средняя множественность их регистрации для одной проволоки составляет всего 2,7. Поэтому для обеспечения более равномерной загрузки ВЦП и уменьшения влияния сбоев в них на потерю физической информации используется система пассивной коммутации (СК) между выходами УФ и входами ВЦП.

В основе СК лежат пассивные разветвители между 8-, 32-, 48-ми контактными разъемами (по два контакта для прохождения сигнала с одной проволоки ДК), расположенных в 5 стойках (тип “А”) вблизи зоны НД и одной стойки (тип “Б”) в месте размещения ЭРД (рис. 2).

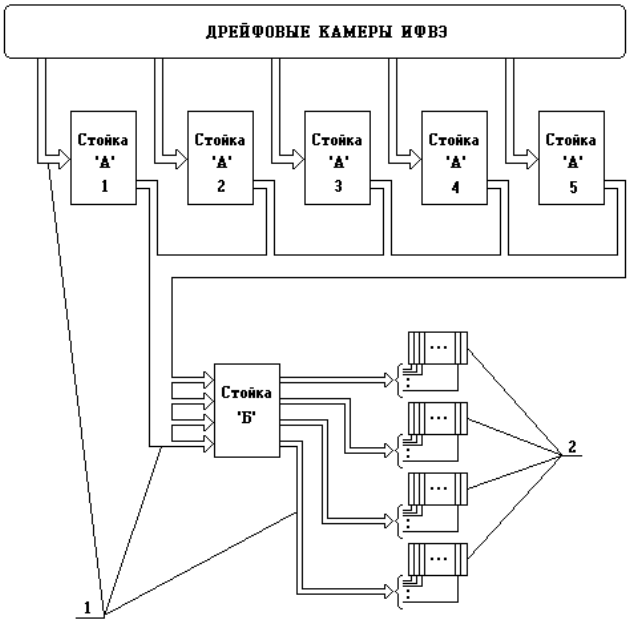

Рис. 2. Схема элементов системы коммутации дрейфовых камер ИФВЭ (1 — кабельные трассы; 2 — каркасы с ВЦП).

СК обеспечивает поступление на входы каждого ВЦП сигналов только с первых, вторых, третьих или четвертых проволок ДК, отстоящих (при их сквозной нумерации) на равном интервале друг от друга, т.е. расположенных в разных местах разных координатных плоскостей мишенной части НД.

Таким образом, имеется однозначное соответствие между каждой проволокой любой ДК и номером входа ВЦП, расположенного в одном из мест какого-либо каркаса. Это позволяет идентифицировать информацию, считываемую с ВЦП в режиме *on-line*. Считая фиксированной коммутацию между входами и выходами стоек “А” и “Б”, в СК имеются элементы (разъемы и ВЦП), у которых по каким-либо причинам может быть изменено местоположение, что приведет к изменению указанного соответствия. Чтобы это не повлияло на работу ССД, любая коррекция в СК должна найти отображение в соответствующем ПО.

3. Восстановление отрезков треков частиц в дрейфовой камере

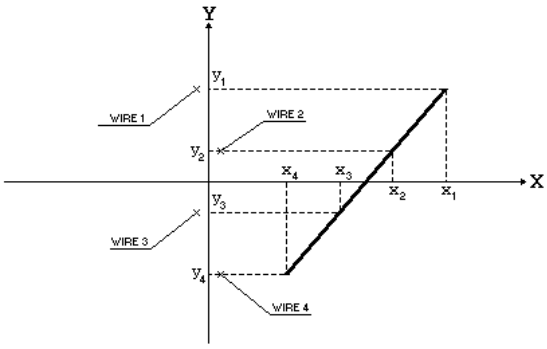

Рис. 3. Система координат в дрейфовой камере.

Для определения эффективностей отдельных проволок ДК, а также визуализации событий в режиме *on-line* был разработан и реализован алгоритм восстановления отрезков прямых треков частиц, проходящих через камеру (стрингов) [7]. При этом учитывались конструктивные особенности ДК — наличие всего 4 сигнальных проволок и смещения между ними. Минимальное число отсчетов, позволяющих определить, с какой стороны от плоскости, проведенной через центр камеры, прошла заряженная частица, равно трем. Поэтому предполагаем, что в стринг могут входить отсчеты со всех проволок или с трех проволок в любой комбинации.

Алгоритм основан на том, что в системе координат, приведенной на рис. 3, для трех отсчетов, лежащих на одной прямой, должно выполняться (без учета ошибок измерений) соотношение

$$T_k = \frac{T_j \cdot (Y_k - Y_i) + T_i \cdot (Y_j - Y_k)}{Y_j - Y_i}, \tag{1}$$

где T_i, T_j, T_k — отсчеты с проволок i, j, k $(i \neq j \neq k)$, а Y_i, Y_j, Y_k — координаты соответствующих проволок. Считая, что расстояние между соседними проволоками одинаковое, из (1) следует

$$T_2 = \frac{T_1 + T_3}{2}, \qquad T_4 = \frac{3T_3 - T_1}{2}, \tag{2}$$

$$T_3 = \frac{T_2 + T_4}{2}, \qquad T_1 = \frac{3T_4 - T_2}{2}, \tag{3}$$

где T_i — отсчет с i-ой проволоки ($i = 1, 2, 3, 4$).

Прежде чем начать поиск стрингов, проводится подготовка следующих рабочих массивов.

1. Упорядоченные наборы отсчетов с каждой проволоки в виде

$$T_1^i + \Delta_i, \cdots, T_{n_i}^i + \Delta_i, -T_1^i + \Delta_i, \cdots, -T_{n_i}^i + \Delta_i, \tag{4}$$

где i — номер проволоки, n_i — число отсчетов с проволоки i, Δ_i — X-координата i-ой проволоки, а T_j — временной отсчет ($T_k^j > T_m^j$ при $m > k$). Удвоение числа отсчетов связано с невозможностью заранее сказать, с какой стороны от плоскости сигнальных проволок прошла частица, отрезку трека которой принадлежит отсчет.

2. Поставленная в соответствие каждому отсчету из (4) пара чисел ($T_j^i - 3\sigma_i, T_j^i + 3\sigma_i$), определяющая интервал для отсчета с i-ой проволоки, если для проверки принадлежности к стрингу будут использоваться формулы (2) или (3). Здесь $\sigma_1, \sigma_2, \sigma_3, \sigma_4$ определяются из (5), где σ_i — среднеквадратичные отклонения отсчетов для i-ой проволоки при условии их нормального распределения:

$$\sigma_1 = \frac{1}{2}\sqrt{9 \cdot {\sigma_2}^2 + {\sigma_4}^2}; \quad \sigma_2 = \frac{1}{2}\sqrt{{\sigma_1}^2 + {\sigma_3}^2};$$

$$\sigma_3 = \frac{1}{2}\sqrt{{\sigma_2}^2 + {\sigma_4}^2}; \quad \sigma_4 = \frac{1}{2}\sqrt{9 \cdot {\sigma_1}^2 + {\sigma_3}^2}. \tag{5}$$

Уменьшение числа рассматриваемых комбинаций осуществляется как методом перебора, так и удалением из списка отсчетов тех из них, которые входят в стринги, состоящие из отсчетов со всех проволок. Перебор включает два этапа. Сначала для каждой пары T_1^i, T_3^k по формулам (2) вычисляются значения для T_2 и T_4, с которыми сравниваются отсчеты со 2- и 4-й проволок по критериям $T_2^j - \sigma_2 \le T_2 \le T_2^j + \sigma_2$; $T_4^k - \sigma_4 \le T_4 \le T_4^k + \sigma_4$. Если такие отсчеты имеются, считаем их входящими в стринг и исключаем из дальнейшего рассмотрения. При наличии отсчета только с одной проволоки, удовлетворяющего соответствующему критерию, отрезок прямой, проходящий через эти 3 отсчета, считаем кандидатом в стринг. Второй этап отличается от первого только тем, что там ищутся стринги и кандидаты в стринги, в которые обязательно должны входить отсчеты со 2- и 4-й проволок.

Все кандидаты в стринги, состоящие из 3 отсчетов, заносятся в специальный массив, который после окончания перебора анализируется по следующим правилам:

- из стрингов, имеющих по два общих отсчета и лежащих с одной стороны от плоскости сигнальных проволок, формируется один, и если в него входят отсчеты со всех 4 проволок, то удаляются те кандидаты, у которых есть хотя бы один отсчет из этой четверки;
- из стрингов, расположенных по разные стороны плоскости сигнальных проволок, оставляем тот, у которого меньше χ^2.

Для оставшихся после данного анализа и отобранных до этого, вычисляем параметры k и b прямой $Y = kX + b$.

4. Форматирование и упаковка данных с ДК

Одной из задач при работе с “сырыми” данными является представление их в виде, удобном для дальнейшего использования непосредственно в ПО ССД и по возможности преобразование в наиболее компактный формат в случае их долговременного хранения для последующей обработки. Успешное ее решение приводит, как правило, к значительному уменьшению времени доступа к требуемым данным, сокращению объема ПО, более полному (с точки зрения информативности) использованию пространства внешнего носителя.

Для ДК каждому отсчету с сигнальной проволоки во внутренней памяти ВЦП соответствуют два слова: в первом время T, а во втором N_t содержатся битовые поля с информацией о номере входа ВЦП, к которому относится данный отсчет; возможности одновременного прихода отсчетов с разных входов данного ВЦП; значение биты интерполятора, т.е. реальные физические данные занимают только одно слово. Направление, в котором искался способ более компактного представления данных с ДК, — замена слов с идентификационной информацией некой структурой, занимающей минимальный объем. В конечном итоге был выбран следующий вариант [8].

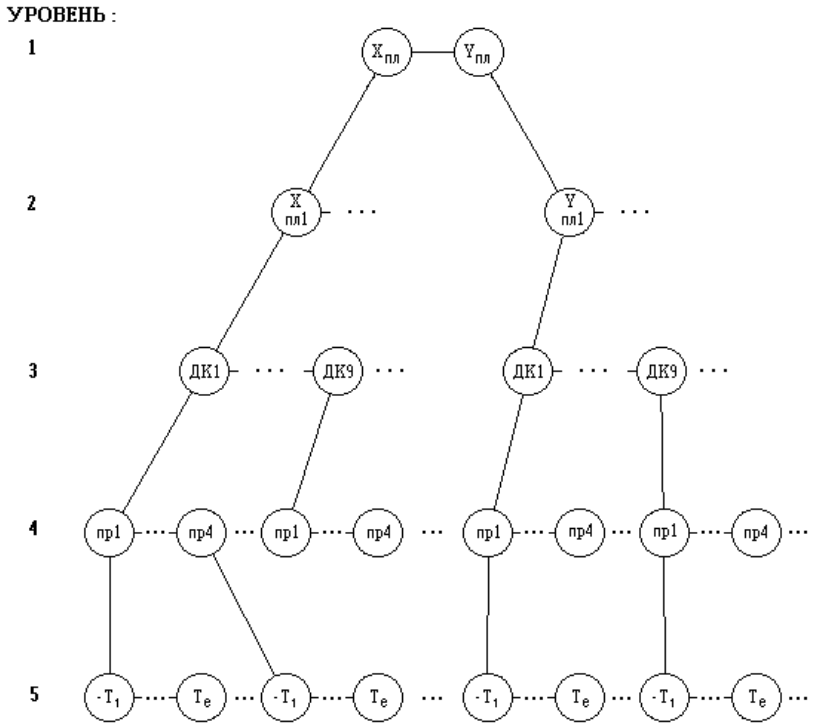

Рис. 4. Структура представления информации в запакованных данных с дрейфовых камер.

Структура части НД, относящейся к ДК, представляется в виде дерева (рис. 4), имеющего 4 уровня, значение узлов на которых следующее:

I уровень — все дрейфовые камеры, регистрирующие одну координату (X или Y);
II уровень — одна плоскость ДК;
III уровень — одна ДК;
IV уровень — одна проволока ДК.

Вместо пар T, N_t формируется массив из одних времен, причем времена, полученные с одной проволоки, расположены последовательно и упорядочены по величине (отдельно для $X-$ и Y-проекций). Времена с проволок, абсолютный номер которых в детекторе меньше, расположены раньше (этот номер имеет вид $N_{pl} \cdot 36 + N_{ch} \cdot 4 + N_{wire}$, где N_{pl}, N_{ch}, N_{wire} — номера плоскости, камеры в плоскости и проволоки в камере, которой принадлежит данное время). Так как все $T \neq 0$, признаком, по которому можно отделить времена с разных проволок, будет то, что первое время, полученное с каждой проволоки, будет отрицательным ($T = -T$). Этот массив добавляется к дереву в качестве пятого уровня. После этого для каждой ДК, у которой не сработала ни одна проволока, убираются соответствующие узлы на III и IV уровнях.

На основе получившегося дерева формируются два массива: один для работы с данными на уровне *on-line* (при этом под каждый узел уровней I-IV отводится одно слово с ссылкой на нижестоящий уровень), а другой для записи на МЛ (здесь узлам I-IV уровней соответствует одна бита со значением 0 — при отсутствии ссылок на нижестоящий уровень, или 1 в противном случае). Общее число слов, занимаемое этой структурой (при битовом представлении ссылок), будет равно $N_t + (N_{plx} + N_{ply} + N_{wpl} \cdot 9 + N_{wch} \cdot 4 + 15)/16$, где N_t — общее число временных отсчетов; N_{plx}, N_{ply} — число координатных плоскостей X и Y; N_{wpl} — число плоскостей, в которых сработала хотя бы одна камера; N_{wch} — число камер, где есть как минимум одна сработавшая проволока.

5. Программное обеспечение ССД с ДК ИФВЭ

Как и для любой экспериментальной физической установки, описываемое ПО ССД подразделяется на две группы, а именно:

- предназначенное для контроля, настройки и получения характеристических параметров непосредственно детектирующих элементов и регистрирующей аппаратуры;
- обеспечивающее считывание реальных физических данных, их первичный контроль, анализ, визуальное представление топологии событий, сохранение для последующей обработки и т.п.

На рис. 5 показана общая схема основных задач ССД с ДК ИФВЭ.

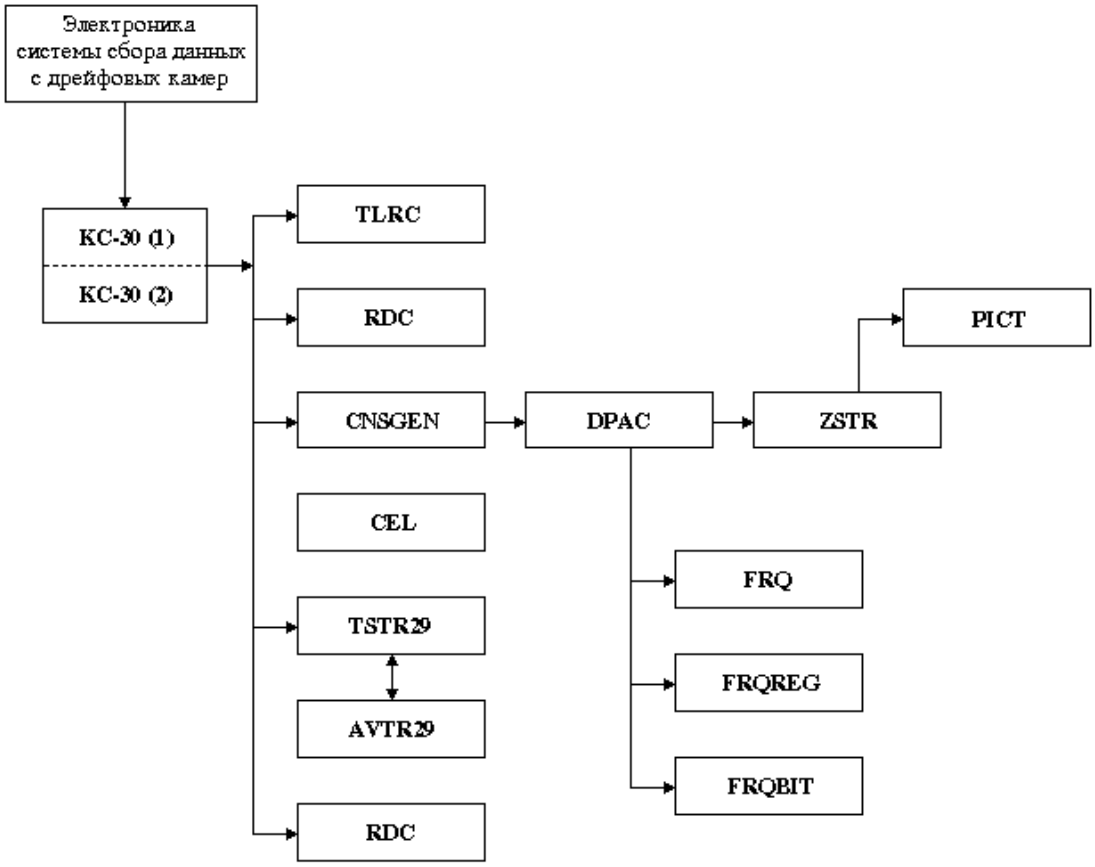

Рис. 5. Блок-схема основных задач, входящих в ССД с ДК ИФВЭ.

Структурно ПО состоит как из задач, функционально независимых друг от друга (т.е. получение входных данных, их обработка, взаимодействие с пользователем, вывод полученных результатов содержатся в каждой из них), так и задач, связанных по входным/выходным данным. Как правило, последние представляют собой линейные последовательности (цепочки), когда каждая последующая задача использует на входе данные, являющиеся выходными у предыдущей. Обычно первой задачей в таких последовательностях является задача чтения данных с электроники ДК. Это позволяет исключить внесение в разные задачи повторяющихся процедур.

5.1. Структура данных и межзадачный обмен данными

Все данные на уровне ССД имеют стандартный заголовок, состоящий из двух слов:

– размер данных в словах, включая само это слово;
– идентификатор данных.

Так как ПО является многозадачной системой, в ней необходимо иметь механизмы обмена данными между задачами. Для этих целей используются:

- ТК-драйвер, позволяющий посылку/прием данных любого размера в случае определенности адресата и его готовности к их получению;
- системные команды SDAT$, RCVD$ при посылке данных до 26 байт с возможностью программного прерывания в момент их прихода.

Помимо этого, в межзадачной синхронизации используются локальные, групповые и глобальные флаги.

Наибольшая эффективность в многозадачной системе при работе с данными достигается при использовании динамических разделяемых областей памяти (REGION). В этом случае не требуется многократного копирования одних и тех же данных в области локальных адресов задач, которым необходим доступ к ним. Одновременно сокращается объем ПО в ОЗУ ЭВМ. При этом возможна анонимность пользователя данных (процесс начала и окончания доступа фиксируется с помощью системных флагов и межзадачных сообщений).

5.2. ПО отображение системы коммутации

Любые изменения в системе коммутации электроники регистрации данных и дрейфовых камер должны быть обязательно адекватно отображены в ЭВМ, так как это определяет соответствие между проволоками в ДК и входами ВЦП. Хранящиеся в ЭВМ для этих целей данные имеют структуру, которая аналогична структуре объектов, входящих в СК. К этим объектам относятся: координатные плоскости дрейфовых камер; дрейфовые камеры в координатных плоскостях; разъемы; ВЦП; каркасы, в которых размещены ВЦП.

Структурными единицами данных являются: система ДК; “вход А”; “выход А”; “вход Б”; “выход Б”; электроника регистрации данных.

Отображение связей между объектами в структурных единицах осуществляется посредством ссылок, число которых между “входом А” и “выходом А”, “входом Б” и “выходом Б” фиксировано. Однако при работе в режиме реального времени использование этих данных в полном объеме нецелесообразно, так как оставаясь практически постоянным, путь от проволоки до входа в ВЦП содержит несколько промежуточных ссылок. Поэтому на базе данных, содержащих полное описание СК, формируются рабочие массивы, позволяющие по номеру входа каждого ВЦП (эта информация есть для каждого временного отсчета) определить абсолютный номер проволоки в системе ДК. Одновременно здесь же строится массив со схемой местоположения всех ВЦП, используемый задачами при чтении электроники сбора данных. Обновление всех этих массивов производится автоматически, после каждой коррекции в СК, для чего используется задача CEL. Любые изменения в СК посредством команд вводятся в задачу, что приводит к корректировке соответствующих ссылок. Помимо этого в задаче есть возможность вывести на печатающее устройство фрагменты всех структурных единиц СК, в которых присутствуют заданные списком или перечислением объекты определенной структуры СК.

Причем для разъемов это детализируется до номеров контактов, что позволяет значительно сократить время поиска неисправностей в СК, так как обычно исходной информацией в этом случае является отсутствие данных с какой-либо проволоки ДК.

5.3. Цифровой контроль “сырых” данных

Основной программой, позволяющей отображать на терминале без предварительной обработки данные, непосредственно считывемые с электроники, является TLRC. Работа с ней осуществляется в интерактивном режиме посредством набора команд (с параметрами или без), обеспечивающих возможность единичного или группового выбора части электроники (каркас, ВЦП, вход), либо системы ДК (плоскость, ДК, проволока) для вывода в указанной моде считанной только с них информации с выделением битовых полей и при необходимости анализа с целью выявления ошибок. Обычно данная задача используется при подаче на входы всех УФ набора тестовых сигналов от генератора, когда известен шаг между временными отсчетами.

5.4. Доступ тестовых задач к данным с ДК

Большинство тестовых задач сами не обращаются к электронике, а получают необходимые входные данные от других задач, а именно RDC, DPAC и ZSTR. Как показано на рис. 6, только RDC имеет доступ к ЭРД, остальные же используют полученные ею данные с ДК. Вспомогательные задачи DPAC и ZSTR служат для формирования структур, содержащих временные отсчеты и стринги в виде, обеспечивающем быстрый доступ к этим данным через набор библиотечных подпрограмм. Посредством этих подпрограмм также осуществляется отображение в область локальных адресов, необходимых REGION, где хранятся требуемые данные.

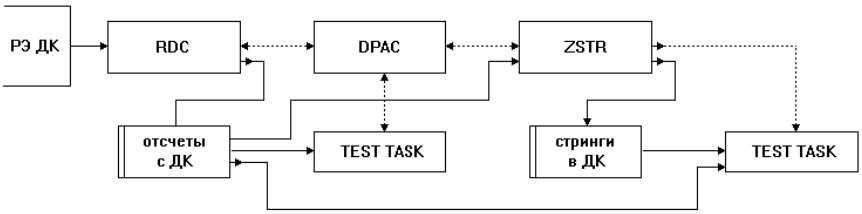

Рис. 6. Схема подключения тестовых задач к цепочке программ RDC, DPAC, ZSTR (пунктирными линиями показаны пути межзадачных синхронизаций).

Так как возможно одновременное использование в качестве входных данных как временных отсчетов, так и стрингов, причем несколькими задачами одновременно, необходимо обеспечить синхронизацию работы всех задач в этой схеме. Для этого используются системные флаги — по два на каждую пару задач, когда выходные данные одной из них (здесь это RDC, DPAC и ZSTR) являются входными для другой.

Задача RDC

Задача RDC обеспечивает прием, корректировку и первичный контроль временных отсчетов с ДК, которые затем передаются другим задачам, осуществляющим весь комплекс проверки работоспособности как самих ДК, так и ЭРД.

После общего сброса всех каркасов и крейт-контроллера КС-30 происходит подключение к прерыванию от сигнала BD, которое вырабатывается по окончанию отцифровки всех временных отсчетов в модулях ВЦП. После прихода прерывания последовательно (в соответствии с описанием ЭРД) считывается информация со всех ВЦП и для каждого из них производится корретировка данных, связанная с возможностью прихода одновременно сигналов с двух и более проволок, а также наличия или отсутствия биты интерполяции. Затем номер входа ВЦП заменяется на абсолютный номер проволоки в системе ДК. При этом отсчеты, которые нельзя идентифицировать с номером проволоки, вследствие сбоев в ЭРД, удаляются. Таким образом, после считывания всей информации с ЭРД имеется массив временных отсчетов и приписанный каждому отсчету абсолютный номер проволоки. Так как работать с подобной структурой неудобно (требуется достаточно большое время на поиск необходимых данных), на ее базе создается ссылочная структура типа дерево, которая вместе с отсчетами заносится в соответствующий REGION, доступный другим задачам. Однако форма представления этих данных не обеспечивает требуемую для режима *on-line* скорость доступа к этим данным. Требуемые преобразования выполняет задача DPAC.

Следует отметить, что во время набора статистики в нейтринном пучке, для чтения данных с ДК работает несколько видоизмененная версия RDC (задача RDC1). В основном отличие касается способа межзадачной синхронизации через ТК-драйвер и наличия записи данных на МЛ. Кроме этого, в RDC1 встроен программный модуль, который контролирует работу ВЦП в предположении, что за определенное число сбросов, вследствие наличия шумовых сигналов с проволок, с каждого ВЦП должен быть прочитан хотя бы один отсчет. Если это условие не выполняется, на терминал выдается соответствующее сообщение.

Задача DPAC

Эта задача получает от RDC временные отсчеты и преобразует их в формат, позволяющий организовать очень быстрый доступ к отсчетам с любой проволоки, камеры и плоскости дрейфовых камер. Структура данных имеет вид дерева и содержит область ссылок в абсолютных адресах от уровня координатных плоскостей (на верхнем уровне) до конкретной дрейфовой камеры (на нижнем уровне) и область временных отсчетов, куда указывают ссылки с нижнего уровня. Подобная структура занимает значительно большее место, чем компактное представление этих же данных, но именно она обеспечивает удовлетворительную скорость их обработки.

Задача ZSTR

Возможность использовать для контроля ДК стринги предоставляет задача ZSTR. На входе имеются отсчеты с ДК, а на выходе формируется структура данных, содержащая параметры стрингов (k, b); по отсчетам с какого числа проволок —

трех или четырех — восстановлен каждый стринг, а также с какой проволоки отсчет не вошел в стринг в случае его восстановления по отсчетам с трех проволок. Такая полная информация нужна для быстрого определения эффективности системы ДК. Так как при определении параметров стрингов требуется знать скорости дрейфа в каждой камере, они берутся из результатов обработки в режиме *off-line* треков от космических мюонов. Также для каждого отсчета учитывается поправка, позволяющая выровнять временные задержки сигналов, существующие вследствие неодинаковой длины кабельных трасс.

5.5. Получение характеристик эффективности системы ДК и РЭ

Для быстрого определения работоспособности проволок системы ДК и связанных с ними каналов ВПЦ используется программа FRQ. Она может быть запущена как в тестовом режиме, так и во время сбора физических данных. Принцип ее работы основан на накоплении числа полученных отсчетов с каждой проволоки и общего числа триггеров. Сразу после загрузки ей необходимо задать запрашиваемое общее число триггеров, а после их набора — для каких плоскостей и конкретно ДК следует выводить отношения N_i/N_T (1). Здесь N_i — число отсчетов с проволоки i, относящейся к множеству заданных камер, а N_T — общее число триггеров. В зависимости от типа устройства, на который будут выводиться отношения (1), а это может быть либо TV-монитиор, либо АЦПУ, определяется форма их представления. Для TV-монитора она выбрана так, чтобы дать качественную картину сразу для всех заданных ДК.

При работе данной программы в процессе набора статистики, ее результаты дают представление о распределении загрузки ДК по всему объему мишенной части, что было особенно важно на этапе работы по оценке фона в районе НД. Если же на входы УФ подавать сигналы с тестового генератора, то при нормальном функционировании всей ССД для каждой проволоки отношение (1) должно быть равно 1. Аналогичный метод и представление результатов используются для определения эффективности работы регистров (задача FRQREG) и отдельных бит буферной памяти регистров в словах, содержащих временные отсчеты (задача FRQBIT).

5.6. Контроль линии связи ЭВМ - ЭРД ДК

Для проверки линии связи ЭВМ с ЭРД используются блоки Р-29, расположенных в каждом каркасе и в память которых может быть записан, а затем считан из нее любой 16-разрядный двоичный код.

Программное обеспечение проверки линии связи включает задачи TSTR29 и AVTR29, которые позволяют:

- корректировать список контролируемых каркасов;
- осуществлять в интерактивном режиме запуск любого из трех режимов формирования тестирующей последовательности кодов (с выводом результатов на терминал), а именно:

– “бегущий 0” или “бегущая 1”;
– всех возможных комбинаций из “0” и “1” в 16 разрядах;
– определенного значения.

- корректировать параметры для режима автоматического контроля, к которым относятся :
  – типы внешних устройств, на которые должны выводиться сообщения о нарушениях в работе контролируемой линии связи;
  – периодичность контроля;
  – разрешение/запрещении записи обнаруженных ошибок на МЛ.

5.7. Определение временных смещений для проволок

Несмотря на то, что в системе коммутации предусматривалось, чтобы длина кабельной трассы от каждого УФ до входа в ВЦП была одинакова, время прохождения сигналов по СК для разных проволок различно. Это может оказать влияние на результаты обработки, особенно при восстановлении отрезков треков в отдельной камере, когда вносимое временное смещение отсчетов от реальных значений существенно отличается для разных проволок. Если знать ΔS_i^j $(i = 2, 3, 4; j = 1, 357)$ — смещения для 3 проволок относительно, например первой, для j-ой камеры, то этим можно скомпенсировать влияние этого смещения при восстановлением отрезков треков. Но остается смещение этих отрезков между камерами, так как существует ΔS_1^j — смещение для первых проволок всей системы ДК относительно какой-либо выбранной камеры.

Для определения ΔS_i^j и S_1^j используется программа CNSGEN. В ней с помощью управляемых временных задержек задаются значения для генератора Г-50, близкие к нормальному распределению, каждое из которых подается одновременно на входы всех УФ (это необходимо, чтобы обеспечить точность для ΔS_i^j и ΔS_1^j менее чем один отсчет генератора, т.е. 10 нс). Число запусков является входным параметром и выбирается, исходя из необходимой статистики для получения среднего значения A_i^j и среднеквадратичного отклонения σ_m^n по каждому из каналов. Если для некоторого набора m и n $\sigma_m^n > P$, где P — заданная величина, то необходимо либо повторить набор статистики, либо попытаться найти неисправность.

В результате формируется массив, содержащий все ΔS_i^j и ΔS_1^j , который заносится в файл. Данные смещения используются при восстановления отрезков треков в ДК как в *on-line*, так и в off-*line* режимах.

5.8. Графическое отображение физических событий в системе ДК

Одним из наиболее действенных методов контроля работы любой экспериментальной установки с достаточно простой топологией регистрируемых физических процессов и небольшой фоновой загрузкой является графическое отображение информации в детектирующих элементах. Особенно это относится к трековым детекторам (для НД таковыми и являются дрейфовые камеры), где, как правило,

требуется хорошее координатное разрешение. Это позволяет визуально заметить систематические отклонения от ожидаемой топологии, что обычно является следствием нарушений в работе электроники или самих детекторов.

Для НД влияние на мюонные треки магнитного поля в рамных магнитах невелико, поэтому общая картина вследствие этого искажается незначительно. Однако если в качестве графических объектов использовать отсчеты с проволок, видимая картина будет весьма далекой от реальности вследствие наличия послеимпульсов, ложных срабатываний и лево-правой неоднозначности. От этих недостатков можно освободиться, если перейти к рисованию фрагментов прямолинейных отрезков треков частиц в ДК. Эту задачу выполняет программа PICT, которая получает восстановленные стринги, подключаясь к задаче ZSTR. Вывод осуществляется на TV-монитор. Более сложный вариант рисования с учетом временной привязки, полученной из данных с ЖСС, реализован на ПЭВМ. В ССД для ДК отладка аппаратуры обычно ограничивается регистрацией космических мюонов в малом временном диапазоне, когда нет искажений треков.

6. Программные средства поддержки интерактивного режима

Одним из режимов интерактивной работы в ССД наряду с меню является командный или директивный, когда управление работой задач, а также ввод необходимых параметров осуществляются посредством команд. Для создания ориентированных на такой режим программных продуктов было создано специализированное ПО, включающее следующие компоненты:

- язык LADI для описания любых символьных структур (подмножеством которых являются команды);
- транслятор текста на языке LADI в исполнительные коды интерпретатора;
- программный интерпретатор исполнительных кодов;
- пакет подпрограмм, обеспечивающих взаимодействие ПО пользователя с интерпретатором.

В основе разработки данного языка было заложено предположение, что все возможные форматы команд предварительно определены и могут состоять из мнемонической части и, возможно, набора параметров. На программный интерпретатор языка возлагаются только функции анализа текста и выделение числовых и символьных параметров, а все вычислительные процедуры и работа с введенными данными производятся на уровне задачи. Это определило набор операторов языка, с помощью которых можно достаточно быстро и просто описать любые символьные структуры, которые затем ищутся в тексте, вводимом с терминала или передаваемом интерпретатору при его вызове.

6.1. Язык LADI

Язык LADI (Language Directives) является средством описания множества эталонных конструкций (ЭК). В частности, ими могут быть операторы языка программирования или директивы операционной системы. Конструкции могут состоять из фиксированных последовательностей символов, а также числовых и символьных переменных (параметров). Для выполнения на уровне задачи процедур, зависящих от конкретного входного текста и, кроме того, могущих быть общими для нескольких команд, в языке заложена возможность разбиения ЭК на группы (правила), каждой из которых приписывается числовое значение (номер правила). С группой ассоциируются параметры, выделенные во входном тексте между двумя правилами. Все найденные номера правил, а также параметры в машинном представлении и тип каждого из них хранятся в отдельных списках и передаются пользователю либо после анализа всего входного текста, либо на любом этапе этого анализа. Это позволяет разбить выполнение команды на несколько этапов. Важнейшим компонентом языка является построение деревьев грамматического разбора, которые задаются в языке с помощью скобок.

Тексты с описанием структур команд и деревьев грамматического разбора (или просто описания) организованы в виде модулей, содержащихся либо в отдельных файлах, либо вместе с текстами программ.

Каждый модуль состоит из заголовка, описания и терминатора конца модуля. Заголовок имеет формат +MODULE modnam[,subint]. Здесь и далее modnam — имя модуля, а subint — имя подпрограммы пользователя, через вызов которой будут обрабатываться все правила, найденные в этом модуле. Если subint отсутствует, его надо указать в параметрах при обращении к программному интерпретатору. Терминатором конца модуля является последовательность “****” в начале строки описания.

Основными компонентами языка являются скобки, операторы и метки.

6.1.1. Операторы

Входной текст, который характеризуется общим числом символов и текущим адресом, анализируется интерпретатором в соответствии с операторами описания. В случаях, когда используются операции сравнения в ЭК и входных символов, а также некоторых других, вырабатывается внутреннее состояние TRUE или FALLS. Ниже приведен список основных операторов языка.

- $L[n]'text'$ — требование на ввод по заданному логическому номеру канала новой строки. Здесь n — максимальное число символов во входной строке, а text — подсказка, выводимая в начале строки. После ввода текущий адрес содержит адрес первого введенного символа.
- F — проверка на наличие необработанных символов во входной строке. Результат TRUE, если входная строка пустая или все символы успешно обработаны, и FALLS при наличии необработанных символов.

- $'text'$ или $"text"$ — сравнение заданной символьной последовательности text с символами во входной строке, начиная с текущего адреса. При совпадении вырабатывается состояние TRUE, и к значению текущего адреса прибавляется величина, равная числу символов в text. Если последовательности символов не совпадают, внутреннее состояние будет FALLS, и значение текущего адреса не изменяется. В этом случае дальнейший порядок анализа определяется структурой скобок в описании.
- Rn — правило, где n — целое число, определяющее номер этого правила. При этом в список правил заносится номер n, а в список параметров — число параметров, найденное с момента обработки предыдущего оператора R, после чего текущее число параметров обнуляется. Обработка каждого из правил заключается в вызове соответствующей исполнительной подпрограммы пользователя.
- Z — выход из интерпретатора. В результате выполнения этого оператора будут обработаны все содержащиеся в списке необработанные правила, после чего управление возвращается в задачу пользователя.
- W — обработка всех найденных до данного оператора правил.
- X — пустой оператор. Обычно используется для разделения ПАС (см. пункт 6.1.2).
- $PcTtextT$ — вывод текста. Здесь c — управляющий символ для печати; T — терминатор текста (любой символ, отсутствующий в $text$); $text$ — подлежащая выводу последовательность ASCII символов.
- $\%type$ — проверка на наличие во входной строке параметра и, если результат проверки TRUE, занесение его в список параметров. Мнемоника $type$ указывает на тип параметра и может быть одним из символов: I (целое со знаком, 2 байта); S (целое со знаком, 4 байта); O (восьмеричное число, 2 байта); B (двоичное число, 2 байта); F (действительное число, 4 байта).

Так как кроме числовых, параметры могут быть и текстовыми, в последний оператор включена возможность их задания в кодах ASCII (тип T). Смысл проверки условия заключается здесь, во-первых, в проверке на принадлежность к списку разрешенных символов, а во-вторых, на наличие терминатора, которым должна завершаться последовательность символов во входной строке. Задание как списка разрешенных символов, так и терминаторов (а их может быть несколько), производится вызовом на уровне задачи подпрограммы SETLST$(lchar, lterm)$, где $lchar$ — список разрешенных символов, оканчивающийся нулевым байтом, а $lterm$ — список терминаторов. Если во входной строке тип параметра не совпадает с указанным типом, результат проверки FALLS.

Для занесения в список параметров значений из описания (т.е. констант) используются операторы: $\hat{}n$ (целое или действительное число в стандартном представлении); $\hat{}On$ (восьмеричное число n); $\hat{}Bn$ (двоичное число n); $\hat{}'text'$ (последовательность символов text).

6.1.2. Скобки

Скобки служат для объединения операторов в группы и задания деревьев грамматического разбора входного текста. В языке возможны три типа скобок:

$< \ldots >$ — альтернативные скобки;
$[\ldots]$ — скобки возможных, но необязательных конструкций;
$(\ldots)$ — скобки повторений.

Каждой открывающей скобке одного типа должна соответствовать закрывающая скобка того же типа при равенстве внутри числа открывающих и закрывающих скобок любого типа.

В программном интерпретаторе при работе со скобками используется специальный рабочий стек, для сохранения значений внутренних переменных интерпретатора при входе в каждую скобку B^i_{open}. Эти значения восстанавливаются в случае возникновения состояния FALLS на этом же уровне скобок. Но в текущий адрес описания в этом случае будет занесен адрес первого оператора (другой скобки) после закрывающей B^i_{close}, парной B^i_{open}. Это означает, что внутри пары скобок $B^i_{open} \div B^i_{close}$ структура одной из команды (или ее части) в описании не соответствует входному тексту. Поэтому будет продолжен поиск соответствующей ЭК. Если же от B^i_{open} до B^i_{close} сохраняется состояние TRUE, то указатель стека вернется в положение до B^i_{open}, т.е. либо внутри скобок не было операторов, могущих изменить внутреннее состояние, либо в пределах данной пары скобок описание соответствует входному тексту.

Конструкции вида "<><> ... <>" называются последовательностью альтернативных скобок (ПАС) и могут быть вложенными. Выход из ПАС происходит при конечном состоянии TRUE в одной из скобок, а при состоянии FALLS осуществляется переход к следующей скобке. Если в описании заданы ЭК для набора команд $C_1 - C_n$, то в общем случае основная часть описания будет иметь вид $< K_1 >< K_2 > \ldots < K_n >$, и структура входного текста будет последовательно сравниваться со структурами в описании $K_1, K_2, \ldots$, пока не будет найдена совпадающая, т.е. ПАС используется для описания всех допустимых вариантов как всего входного текста, так и любых его частей.

Если в скобках возможных конструкций возникло состояние FALLS, то все содержимое в них игнорируется, и анализ продолжается дальше, начиная с первого оператора после "]".

Если в описании встретились скобки повторений, то анализ входной строки будет проводиться в соответствии с описанием внутри скобок до тех пор, пока не возникнет состояние FALLS. В этом случае анализ продолжается, начиная с оператора после ")". При задании в описании конструкции типа (D)*n, где n — обязательное число повторений и D — выражение внутри скобок, это эквивалентно описанию D, повторенному n раз и заключенному в одни альтернативные скобки.

6.1.3. Метки

Отдельные, часто повторяющиеся части описания могут быть выделены в специальные блоки и, будучи помеченными, образуют структуры, аналогичные подпрограммам, выход из которых осуществляется с помощью оператора “O”, либо при состоянии FALLS на нулевом уровне скобок. Помеченные блоки имеют формат:

$$:label \qquad text \qquad O,$$

где $text$ — описание, могущее занимать произвольное число строк, а $label$ — алфавитно-цифровая последовательность произвольной длины. Оператором передачи управления в интерпретаторе на начало описания в помеченном блоке является $\#label$, вслед за которым находится точка возврата из этого блока. В результате этого оператора внутреннее состояние станет тем, каким оно было к моменту выхода из помеченного блока.

Метки $:label$ являются локальными, т.е. ссылки на них возможны только в пределах того модуля, в котором они определены. Для использования описаний из других модулей, а также создания библиотек описаний, в языке введены глобальные метки, имеющие формат $::label$. Ссылки на блоки, помеченные такими метками, производятся так же, как и на локальные, но если глобальные метки определены в другом модуле, они должны быть перечислены с помощью директив GLOBAL после заголовка модуля:
+GLOBAL $\quad glabel1, glabel2, \ldots$

Все номера правил и соответственно параметры, определенные или заданные во время обработки в блоках, помеченных глобальными метками, будут интерпретированы подпрограммами, указанными в заголовке тех модулей, где описаны эти блоки.

6.2. Транслятор

Обработка модулей осуществляется двухпроходным препроцессором, на выходе которого формируется преобразованный в исполнительные коды интерпретатора исходный текст описаний, который заносится в выходной файл в виде выделенного в отдельный программный модуль массива констант. Если в файле помимо модулей на языке LADI содержится другая текстовая информация, она переносится в выходной файл без изменений.

При первом проходе проводится синтаксический анализ текста описания: форматов операторов; структуры скобок; составляется таблица ссылок на метки. Трансляция ведется до обнаружения ошибки. В этом случае на терминал выдается сообщение об имени модуля, номера строки, в которой обнаружена ошибка, а также выводится часть строки с того символа, где трансляция была прекращена. Необходимо исправить ошибку и повторить трансляцию. Если есть отсутствующие метки, то об этом также выдается сообщение.

Во время второго прохода формируются исполнительные коды для интерпретатора с учетом меток, имеющихся в данном описании и заданных в заголовке

как глобальные. Последовательность операторов и скобок в исполнительных кодах соответствует описанию. Так как при нулевом уровне скобок происходит выход из интерпретатора, на уровне трансляции весь текст описания (до первой метки) заключается в альтернативные скобки.

6.3. Программный интерпретатор

Интерпретатор исполнительных кодов, полученных после трансляции модулей, представляет собой набор подпрограмм, часть которых выполняет анализ входного текста в соответствии с тем или иным описанием, а остальные обеспечивают взаимодействие с задачей пользователя.

Перед началом работы необходимо указать интерпретатору адреса и размеры массивов для хранения параметров и их типов через вызов подпрограммы

CALL LADIST(*arrpar*,*lenpar*,*arrtyp*,*lentyp*).

Передача управления интерпретатору происходит после вызова функции

CALL LADI(*modnam*[,*subint*][,*inptxt*]),

где *modnam* — имя модуля описания; *subint* — имя подпрограммы для обработки правил в этом модуле (если опущено, берется из заголовка модуля); *inptxt* — входной текст (можно не задавать, если в описании есть оператор L). При успешном анализе входной строки возвращаемое значение функции будет 1 и -1 в противном случае.

Работа интерпретатора начинается с выполнения операторов в указанном модуле и, по мере необходимости, происходит посимвольное считывание из входной строки (когда встречается соответствующий оператор). Порядок выполнения операторов определяется структурой скобок с учетом внутреннего состояния. На уровне интерпретатора организованы стеки для хранения указателей на номера правил, параметров, типов параметров в массивах пользователя, а также рабочий стек. При входе в открывающую скобку в рабочий стек заносятся:

- текущие адреса стеков номеров правил, параметров и типов параметров;
- адрес символа во входной строке;
- адрес описания за закрывающей скобкой этого же типа, парной данной;
- некоторые рабочие переменные.

Вызов подпрограммы пользователя для обработки правил с найденными для каждого правила параметрами осуществляется операторами W и Z. При этом будет вызываться пользовательская подпрограмма SUBINT:

CALL SUBINT(*nrule*,*nparam*,*adrpar*,*adrtyp*),

где *nrule* — номер правила; *nparam* — число параметров, заданных или найденных для этого правила; *adrpar* — адрес списка значений параметров в массиве *arrpar*; *adrtyp* — адрес списка типов этих параметров в массиве *adrpar*.

Возврат управления в задачу пользователя происходит в следующих случаях:

– если в описании встретился оператор E;

– при окончании обработки всех символов во входной строке и выходу на нулевой уровень альтернативных скобок;
– если в результате анализа входной строки, начиная с какого-то символа нет соответствующей эталонной структуры в данном описании (при этом выводится сообщение об ошибке).

7. Макрокоманды ввода/вывода для MACRO-11

Язык MACRO-11, на котором написана большая часть ССД с ДК ИФВЭ, обладает тем существенным недостатком, что в нем отсутствуют развитые встроенные средства организации ввода/вывода. В значительной мере устранить этот недостаток позволила разработка и реализация набора макрокоманд и пакета подпрограмм, дающих возможность достаточно просто осуществлять вывод на терминал или устройство печати в символьном представлении почти всех типов данных MACRO-11, а также производить обратную процедуру при вводе с терминала. Данный пакет во многом аналогичен структурам ввода/вывода для языка "C". Ниже дано краткое описание этих макрокоманд.

Присоединение устройства, на которое будет осуществляться вывод, производится с помощью макрокоманды

$$\text{TILUN} \qquad [device], [unit], [lun].$$

Здесь *device* — двухсимвольное имя устройства; *unit* — его физический номер; *lun* — номер программного канала. По умолчанию *device*=TI, *lun*=5.

Следующая макрокоманда выполняет ту же функцию, что и оператор FORMAT в языке FORTRAN:

$$\text{TITXT} \qquad < string > .$$

Параметр $< string >$ представляет собой строку из последовательностей ASCII символов, которые будут выводиться на внешнее устройство и символьных комбинаций управления форматами ввода/вывода, начинающихся с "&" для вывода и "$" для ввода (последовательности "&&" и "$$" будут интерпретированы соответственно как "&" и "$"). После управляющего символа для вывода следует описание формата, которое может иметь вид (здесь n - число позиций для символьного представления выводимой величины): Признаком запроса на ввод служит наличие в $< string >$ двухсимвольной комбинации вида "$X", в которой X определяет тип вводимой величины и может принимать одно из следующих значений:

B — целое типа INTEGER*4;
I — целое типа INTEGER*2;
S — целое типа INTEGER*1;
O — восьмеричное типа INTEGER*4;
A — последовательность ASCII символов, признаком конца которой является пробел, табуляция или CR;
E — действительное число по *E* или *F* формату.

Если требуется продолжить описываемый в $< string >$ формат, используется макрокоманда

TIADD $< string >$.

Задание списка параметров ввода/вывода осуществляется посредством двух макрокоманд

TIVAL $list_1$ или TIVAR $list_2$.

Параметры $list_1$ и $list_2$ представляют собой списки мнемонических имен переменных для ввода/вывода. Различие между ними состоит в том, что в помимо имени переменной указывается ее размерность. Если требуется операция, связанная с вводом/выводом параметров, а список адресов, заданных в TIVAL или TIVAR, пуст или исчерпан, ввод/вывод завершается.

Передача управления подпрограммам интерпретации в $< string >$ для реализации ввода/вывода производится с помощью макрокоманды

TIEND $[arrout]$.

Наличие адреса массива arrout приводит к тому, что сформированная выходная строка будет занесена в этот массив.

Для упрощения задания в $< string >$ последовательности повторяющихся групп символов введена конструкция вида "&N$[m]$ $< substring >$ &V", где m, если оно задано, является числом повторений $< substring >$. В противном случае, значение m берется из списка параметров.

В случаях, когда необходимы операции ввода/вывода с действительными числами, используется макрокоманда TIREAL. Ее функция заключается только в указании построителю задач присоединить соответствующие подпрограммы.

Последовательность макрокоманд для одной процедуры ввода/вывода следующая:

TITXT $< string >$
[TIADD $< string >$]
. . . . . .
[TIVAL/TIVAR]
. . . . . .
[TIVAL/TIVAR $list^n_{1/2}$]
TIEND $[arrout]$

Заключение

Программное обеспечение системы сбора данных для дрейфовых камер ИФВЭ на начальном этапе позволило осуществить весь комплекс работ по отладке как самих камер, так и регистрирующей электроники. С помощью специально созданных программ, которые не вошли в конечный вариант ССД, были исследованы фоновые условия в зоне НД, что помогло выработке технических решений по снижению его до приемлемого уровня. В эксплуатационном режиме, который ведется

с 1986 г., программный комплекс обеспечивал проведение предпусковой настройки, а при наборе статистики в нейтринном пучке — контроль электроники, съем и первичный анализ данных для системы дрейфовых камер ИФВЭ. Алгоритм восстановления отрезков прямых треков, проходящих через камеру, используется не только в режиме *on-line*, но и при формировании DST, причем для всех дрейфовых камер в мишенной части и мюонном спектрометре НД.

В заключение авторы считают своим долгом поблагодарить А.С. Вовенко, А.Г. Карева, М.М. Кирсанова, С.А. Мухина, Р.М. Фахрутдинова за плодотворные дискуссии и полезные советы в процессе разработки алгоритмов и создания программ.

Список литературы

[1] Barabash L.S. et al. — In: Proc. of Intern. Conf. Neutrino-82, 1982, v. 2, p.249; Барабаш Л.С. и др. ОИЯИ З1,2,13-83-81. - Дубна, 1983.

[2] Божко Н.И., Борисов А.А., Булгаков Н.К. и др. Система дрейфовых камер ИФВЭ нейтринного детектора ИФВЭ-ОИЯИ: Препринт ИФВЭ 87-93. Серпухов, 1987.

[3] Божко Н.И., Борисов А.А., Вовенко А.С. и др. Характеристики дрейфовых камер при регистрации наклонных треков. Препринт ИФВЭ 83-112. Серпухов, 1983.

[4] Бушнин Ю.Б., Исаев А.Н., Коноплянников А.К. и др. Электронная аппаратура для регистрации сигналов с дрейфовых камер нейтринного детектора. // ПТЭ. 1984, № 6, с.80.

[5] Божко Н.И., Вовенко А.С., Глебов В.Ю. и др. Большой жидкостной счетчик. // ПТЭ. 1985, №2, с.57.

[6] Бамбуров Н.С., Бушнин Ю.Б., Коноплянников А.К. и др. Управляемые модули автоматизированного стенда для тестирования аппаратуры: Препринт ИФВЭ 80-150. Серпухов, 1980.

[7] Беликов С.В., Липаев В.В. Метод восстановления отрезков треков частиц, проходящих в дрейфовой камере. – В кн.: Материалы V Рабочего совещания по нейтринному детектору ИФВЭ-ОИЯИ, ОИЯИ Д1,2,13-84-332, Дубна, 1984, с.162-164.

[8] Беликов С.В., Липаев В.В. Прием и форматирование данных с электроники дрейфовых камер. – В кн.: Материалы V Рабочего совещания по нейтринному детектору ИФВЭ-ОИЯИ, ОИЯИ Д1,2,13-84-332, Дубна, 1984, с.165-167.

Рукопись поступила 29 августа 1996 г.

А.А.Борисов и др.
Организация и программное обеспечение системы сбора данных для дрейфовых камер ИФВЭ нейтринного детектора ИФВЭ-ОИЯИ.

Оригинал-макет подготовлен с помощью системы LaTeX.
Редактор Н.В.Ежела. Технический редактор Н.В.Орлова.

Подписано к печати 4.09.96. Формат $60 \times 84/8$. Офсетная печать.
Печ.л. 2,37. Уч.-изд.л. 1,82. Тираж 240. Заказ 827. Индекс 3649.
ЛР №020498 17.04.97.

ГНЦ РФ Институт физики высоких энергий
142284, Протвино Московской обл.

Индекс 3649

П Р Е П Р И Н Т 96–71, И Ф В Э, 1996