МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
им. М.В. ЛОМОНОСОВА

НАУЧНО-ИССЛЕДОВАТЕЛЬСКИЙ ИНСТИТУТ
ЯДЕРНОЙ ФИЗИКИ им. Д.В. СКОБЕЛЬЦЫНА

И.В. Грибов, И.В. Шведунов, В.Р. Яйлиян

ТЕХНОЛОГИЯ СОЗДАНИЯ СИСТЕМЫ УПРАВЛЕНИЯ СОВРЕМЕННЫМИ УСКОРИТЕЛЯМИ ЭЛЕКТРОНОВ.

Препринт НИИЯФ МГУ – 2002-17/701

УДК 621.384.6:004
ББК 32.85

I.V. Gribov, I.V. Shvedunov, V.R. Yalijan

ELECTRON ACCELERATORS CONTROL SYSTEM TECHNOLOGY.

Preprint INP MSU

Abstract

We have developed a tri-level personal computers-based electron accelerators control system. Custom smart front-end devices monitor some 20 pulsed parameters while other parameters are controlled with commercial multi-channel data acquisition cards. We use a real time Linux operating system and fulfill both hard and soft real time tasks in the first and second levels. Static optimization and dynamic direct digital methods are used in the second level to control parameters and assure long-term stabilization. We use a parametric technology to form the accelerators information model. In the third level we support a database containing fully described accelerator parameters and operational rules, man-machine and world interfaces, and implement script technology.

И.В. Грибов, И.В. Шведунов, В.Р. Яйлиян

ТЕХНОЛОГИЯ СОЗДАНИЯ СИСТЕМЫ УПРАВЛЕНИЯ СОВРЕМЕННЫМИ УСКОРИТЕЛЯМИ ЭЛЕКТРОНОВ.

Препринт НИИЯФ МГУ

Аннотация

В работе описана универсальная трехуровневая система управления, поддерживающая работу нескольких электронных ускорителей. В системе используются коммерческие многоканальные устройства аналогового ввода-вывода, а также специализированные модули импульсных мониторов тока электроного пучка. Программное обеспечение системы управления разработано на основе операционных систем Linux и Real Time Linux. Используются методы прямого цифрового регулирования и статические оптимизационные алгоритмы. Информационная модель объектов управления формируется с помощью параметрической технологии. На верхнем уровне системы поддерживаются база данных объектов, два типа человеко-машинных интерфейсов и реализована техника сценариев.

E-mail: gribov@depni.npi.msu.su
E-mail: ivan_iv@depni.npi.msu.su
http://www.sinp.msu.ru

Введение.

В НИИ Ядерной Физики МГУ имени М.В. Ломоносова при поддержке фирмы World Physics Technologies разработаны и запущены в работу компактные ускорители электронов нового поколения – два импульсных разрезных микротрона на энергию 35 и 70 Мэв. Работа обоих машин поддерживается современной трехуровневой системой управления /1/, созданной на основе PC совместимых компьютеров с операционной системой **R**eal **T**ime **Linux** (рис. 1). Система состоит из нескольких станций, объединенных сетью Ethernet: бездисковых, поддерживающих нижние (первый и второй) уровни управления и одной полнофункциональной станции верхнего уровня. Станции нижнего уровня оснащены многоканальными устройствами сбора и распределения данных: 32 канальными 16 разрядными аналого-цифровыми преобразователями (АЦП) типа ADLink PCI-9114, также поддерживающими 16 битовые порты цифрового ввода-вывода, и 16 канальными 16 разрядными цифро-аналоговыми преобразователями (ЦАП) ADLink PCI-6216. Кроме того, на первом уровне системы задействованы импульсные мониторы пучка, разработанные на основе цифровых сигнальных процессоров и подключенные к станциям второго уровня посредством сети CAN-bus /2/.

Для оптимизации стоимости системы управления нами были разработаны и изготовлены дополнительные оконечные устройства:

- двуполярные аналоговые источники тока до 2 ампер для магнитных элементов ускорителя;
- устройства управления шаговыми двигателями, использующие прямое цифровое регулирование;
- устройства управления теплоэлектронагревателями (ТЭН) на твердотельных реле, не создающие электромагнитных помех;
- импульсные устройства измерения расхода охлаждающей воды.

Разработка системы велась таким образом, чтобы большинство низкоуровневых задач контроля и управления решалось программным обеспечением. Использование методов прямого цифрового управления позволило существенно упростить конструкцию оконечных апаратных устройств.

В работе рассмотрены основные программные технологии, применяемые в действующей системе управления.

1. Структура программного обеспечения нижних уровней системы управления.

Созданная для новых ускорителей система управления опирается на разработки, проведенные нами в конце восьмидесятых годов, когда была сконструирована полномасштабная система управления инжектором разрезного микторона /3/. Мы разработали технологию создания высоко надежного программного обеспечения реального времени, которая включала в себя:

- Безопасные по отношению к сигналам компоненты, в частности, драйверы устройств сбора и распределения данных.
- Статистическую обработку первичных данных для повышения эффективной разрядности аналого-цифровых преобразователей и точного измерения медленно изменяющихся параметров в условиях высоких электромагнитных помех.
- Динамические методы прямого цифрового регулирования, реализующие ПИ закон.
- Статические методы управления на основе алгоритмов оптимизации.

- Параметрическое представление управляемого объекта.

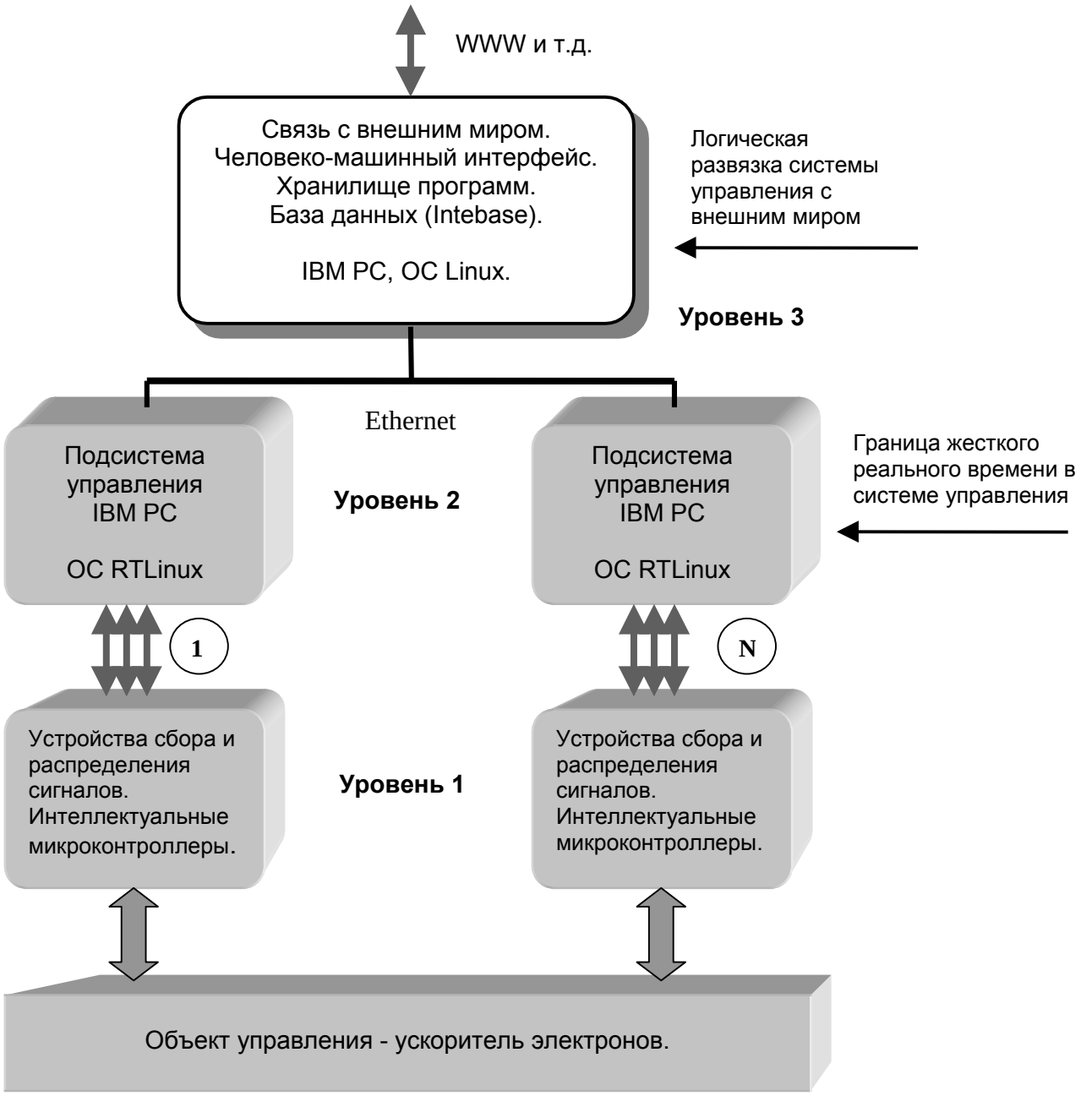
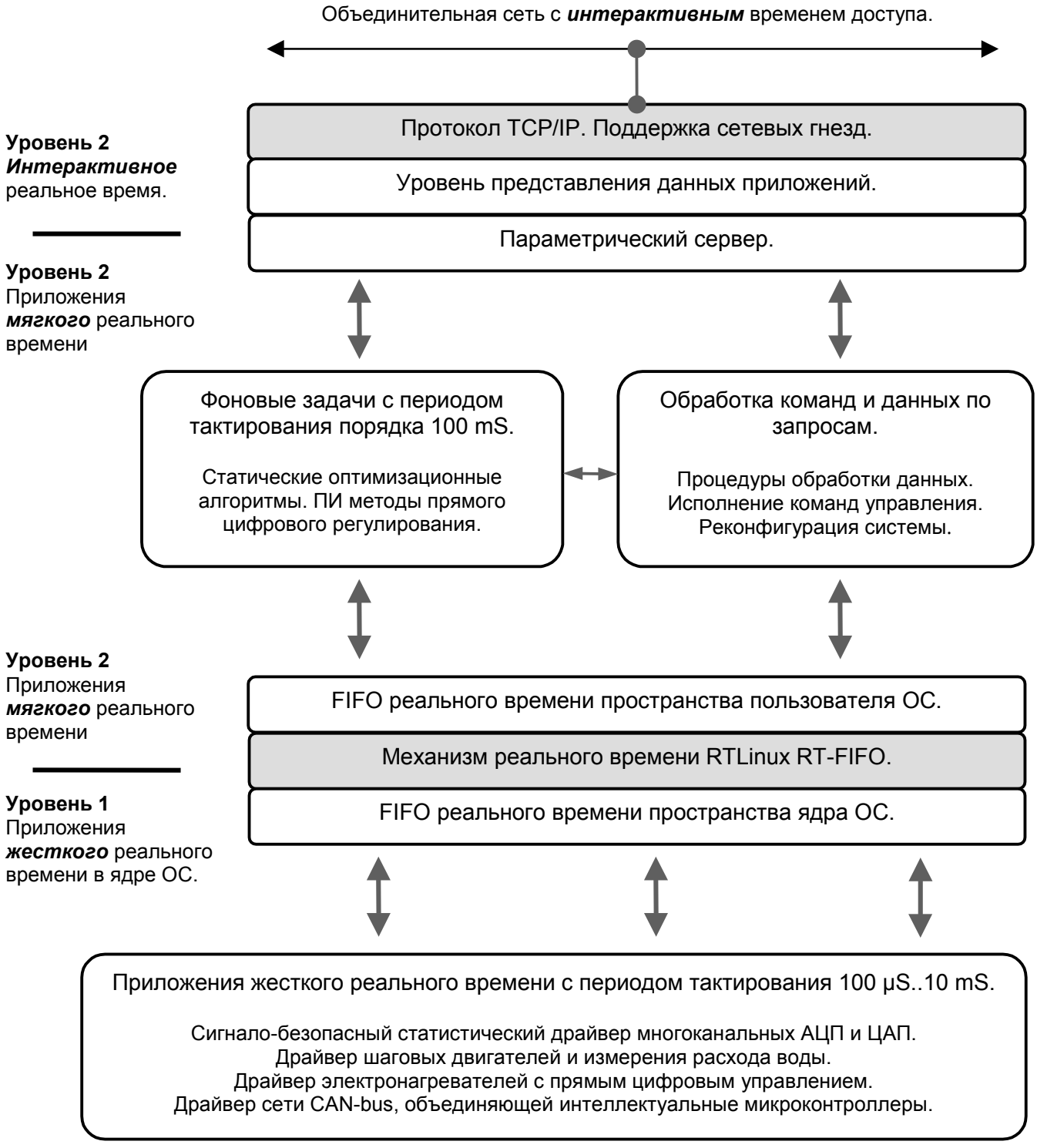


Рис. 1. Система управления ускорителя.

Для нижних уровней новой системы управления (рис. 2) мы использовали хорошо зарекомендовавшие себя решения из прежней системы. Были усовершенствованы и переписаны для операционной системы RTLinux сигнало- безопасные драйверы устройств и алгоритмы прямого цифрового регулирования. Задачи жесткого реального времени, работающие на первом уровне системы управления, либо тактируются аппаратными прерываниями либо, если задание работает в режиме ведомого, оно активируется со второго уровня посредством RT-FIFO механизма операционной системы RTLinux. Все процессы, исполняемые на втором уровне системы управления, синхронизуются двумя сигналами: 10 Гц таймером, запускающим периодические задачи, и сигналом приема данных из сети, инициирующим задания, исполняемые станцией в качестве параметрического

сервера. Причем приоритет таймерного сигнала выше сетевого, то есть процедуры обработки данных и команд, принятых из сети могут прерываться таймером, но не наоборот. Это обеспечивает гарантированное качество исполнения станцией всех периодических функций управления даже при высоком уровне сетевых запросов.



Затемненные блоки являются компонентами операционной системы RT Linux.

Рис. 2. Структура программного обеспечения станции первого/второго уровня.

1.1. Драйверы устройств сбора и распределения аналоговых данных.

Драйверы многоканальных устройств аналого-цифрового и цифро-аналогового преобразователей функционируют в области ядра ОС RTLinux, взаимодействуя с пользовательскими приложениями посредством механизма RT-FIFO. Оба драйвера являются сигнало-безопасными, то есть обеспечивают постоянную программную доступность соответствующих устройств.

Драйвер ЦАП ADLink PCI-6216 осуществляет транзакцию записи заданного значения в выбранный канал устройства. С целью обеспечения сигнало-безопасности все управляемые прерывания центрального процессора на время выполнения транзакции запрещаются. Максимальное время блокирования прерываний не превышает 2.2 мкс и возникает лишь в том случае, когда данные записываются подряд в несколько каналов одного преобразователя. Драйвер также отрабатывает тайм-аут готовности устройства.

Драйвер АЦП ADLink PCI-9114 обеспечивает истинно повторно-входимый режим измерения данных, что означает постоянную доступность любого канала устройства при очередности «последний пришел – первый ушел». Драйвер не использует управление прерываниями центрального процессора, ограничиваясь при необходимости повторным выполнением программного кода собственных критических секций.

Статистическая обработка первичных измерений производится с использованием метода усеченного среднего в предположении симметричного распределения выборки данных. При этом из первичной выборки отбрасывается как минимум по одному минимальному и максимальному значению, а остальные усредняются. Статистическая обработка позволяет достичь 15.5 бит эффективной разрядности измерений для 16 разрядного АЦП. Этот результат проверялся следующим образом: производилось непрерывное измерение медленно и линейно изменяющегося аналогового сигнала. При этом, когда результат измерения начинал переходить с N-го канала АЦП на N+1-ый, результатов N-1-го канала уже не фиксировалось. В то же время, при отсутствии статистической обработки, нестабильность собственных измерений АЦП достигала трех младших разрядов. Помимо повышения эффективной разрядности, метод усеченного среднего дополнительно отфильтровывает кратковременные (до 20 мкс для нашего режима измерений) импульсные помехи в исходном сигнале. Конечно, статистическая обработка увеличивает минимальный период измерений пропорционально числу данных первичной выборки.

Помимо обработки первичных данных драйвер АЦП дает возможность отфильтровать наводки с частотой 50 Гц от сети переменного тока для медленно изменяющихся параметров. С этой целью любой канал АЦП может быть переключен в режим проведения фоновых измерений с периодом около 8.3 милисекунд. Результаты, набранные за последние 100 либо 200 милисекунд при каждом новом измерении усредняются, формируя окончательное значение соответствующей величины. Для обеспечения анти-алиасинга гармоники с частотой, превышающей 60 Гц дополнительно подавляются в исходном аналоговом сигнале. В совокупности методы статистической обработки и фоновых измерений обеспечивают точность контроля аналоговых сигналов при работающем ускорителе не хуже 15 бит для 16 разрядного АЦП.

1.2. Драйверы устройств цифрового управления.

Многоканальный драйвер измерения расхода воды регистрирует частоту меандров, формируемых устройством нормализации сигналов шариковых расходомеров. Набор меандров переменной частоты (от единиц до 30 Гц) считывается портом цифрового ввода модуля ADLink PCI-9114 с разрешением около 1 мс. Полный цикл измерения частоты составляет пять секунд. Для повышения устойчивости измерений драйвер формирует временное окно чтения, исключающее возможные просчеты вследствие переколебаний на фронтах меандра. По окончании цикла измерения в соответствии с калибровочной характеристикой расходомера частота меандров пересчитывается в величину расхода воды.

Управление шаговыми двигателями осуществляется с помощью аппаратного модуля усилителей, формирующих 4-фазную последовательность импульсов тока обмоток на основе двух цифровых битов. Модуль имеет общую схему формирования токов обмоток и позволяет подавать питание только на один выбранный двигатель. Программный драйвер осуществляет управление, формируя на основе числа и знака неотработанных шагов, а также номера выбранного двигателя соответствующую битовую маску. Максимальная скорость вращения составляет 960 шагов в секунду и может быть уменьшена с помощью программного делителя тактовой частоты. После совершения каждого шага проверяется состояние концевых выключателей и при срабатывании одного из них вращение двигателя в соответствующую сторону прекращается. Драйвер может постоянно принимать новые значения требуемого числа шагов, что позволяет, например, остановить двигатель в любой момент времени заданием нулевого значения.

Тактирование драйверов измерения расхода, управления шаговыми двигателями, равно как и фоновых измерений АЦП производится от свободного COM порта станции, настроенного на частоту генерации прерываний 960 Гц.

Управление мощностью теплоэлектронагревателей, являющихся частью системы высокоточной стабилизации температуры ускоряющих секций, также осуществляется прямым цифровым методом. С этой целью было сконструировано устройство коммутации переменного тока на основе твердотельных реле. Отличительным свойством таких реле является наличие в них детектора нуля сетевого напряжения; поэтому они выдают мощность в нагрузку только целым числом полупериодов, практически не создавая электромагнитных помех. Каждый канал управления температурой содержит по четыре ТЭНа, что обеспечивает 257 уровней нагрева при полном периоде регулирования 640 милисекунд. Динамический диапазон регулирования составляет около 4°C, таким образом дискретность управления температурой не превышает 0.016°C. Значительная инерционность тракта терморегулирования позволяет улучшить точность задания температуры в 2 – 4 раза путем соответственного увеличения периода управления нагревателями.

Включение твердотельного реле в очередном полупериоде сети производится подачей на вход управления цифрового сигнала, который должен находиться в активном состоянии во время пересечения сетевым напряжением нуля. Для согласования работы программы драйвера и фазы сетевого напряжения был изготовлен модуль синхронизации. Он обеспечивает выработку сигнала прерывания в пределах 10..30 градусов от начала очередного полупериода, что оставляет достаточно времени (свыше полутора милисекунд) для формирования драйвером битовой маски включения нагревателей в любой из трех фаз переменного тока. Прерывания вводятся в станцию нижнего уровня через бит подтверждения параллельного порта ЭВМ. При каждом прерывании драйвер формирует маску включения всех реле в ближайшем полупериоде сети. Для лучшего сглаживания колебаний температуры нагревателей в пределах полного периода регулирования драйвер производит максимально равномерное его заполнение включенным состоянием полупериодов сетевого напряжения. При этом используется метод

половинного деления неактивных отрезков периода регулирования, а также взаимный сдвиг управления каждым из четырех ТЭНов на четверть полного периода регулирования.

1.3. Основные алгоритмы управления второго уровня.

На втором уровне системы управления реализованы два класса однопараметрических алгоритмов: статические безмодельные алгоритмы оптимизации и динамические модельные алгоритмы, реализующие классическую схему пропорционально-интегрально метода регулирования. Задачей этих алгоритмов является точный установ и долговременное поддержание заданных величин различных параметров ускорителя. Исполнение всех алгоритмов тактируется единым таймером с периодом 100 милисекунд.

1.3.1. Статический безмодельный оптимизационный алгоритм.

Оптимизационный алгоритм использует статическое управление в предположении, что время поддержания собственной стабильности управляемого параметра превышает длительность переходного процесса после подачи управляющего воздействия. Таким образом, алгоритм работает только с установившимися значениями параметров управления. Алгоритм основан на методе секущих /4/, используемом для поиска изолированных корней алгебраических уравнений. Это означает, что значение управляемого параметра должно монотонно зависеть от величины управляющего воздействия в пределах диапазона регулирования.

Алгоритм производит измерение значения управляемого параметра в каждом такте, инициируя новый цикл подстройки если его отклонение от заданной величины превысило максимально допустимое. Критерием завершения цикла является итерация, в которой рассчитанное по методу секущих смещение управляющего параметра становится менее половины минимального шага управления.

Помимо основного метода, алгоритм содержит дополнительную логику, придающую ему высокую устойчивость и надежность при работе в управляющей среде:

- Алгоритм адаптивно ограничивает максимальный шаг управления в каждом цикле подстройки. Допустимое приращение определяется в начале очередного цикла из соотношения измеренного отклонения управляемого параметра от заданного значения и полного диапазона регулирования.
- Алгоритм контролирует величину изменения управляемого параметра в каждой итерации. Если она становится менее допустимой в нескольких итерациях цикла подстройки, то регистрируется потеря управления и расчет по методу секущих не проводится.
- В случае, если цикл подстройки завершается с ошибкой и отклонение измеряемого параметра от заданного значения превышает определенную величину, алгоритм восстанавливает значение управляющего параметра, которое он принимал до начала цикла подстройки.
- Когда полное число ошибок управления для некоторого параметра становится слишком большим, алгоритм прекращает работу с ним, переходя в состояние ожидания. Такое происходит, например, при отключении управляемого прибора либо его аппаратной неисправности.

Одной из областей применения статического алгоритма в системе управления ускорителя является долговременная подстройка значений токов источников, запитывающих различные системы ускорителя. Существенным удобством при его

использовании является отсутствие необходимости калибровки и автоматическая компенсация нелинейности тракта управления. Алгоритм всегда обеспечивает заданную величину измеряемого значения параметра, поддерживая долговременную стабильность источников тока с точностью не хуже $5*10^{-5}$.

Оптимизационный алгоритм используется также для поддержания долговременной стабильности средней величины СВЧ мощности клистрона, запитывающего ускоряющие секции (рис. 3). Мощность клистрона может регулироваться как вручную, путем непосредственного управления P-I-N аттенюатором, так и автоматически. В последнем случае производится измерение величины мощности на момент переключения в режим стабилизации с последующим поддержанием этого значения. Соответственно, при возврате в режим ручного управления исполнение алгоритма стабилизации приостанавливается.

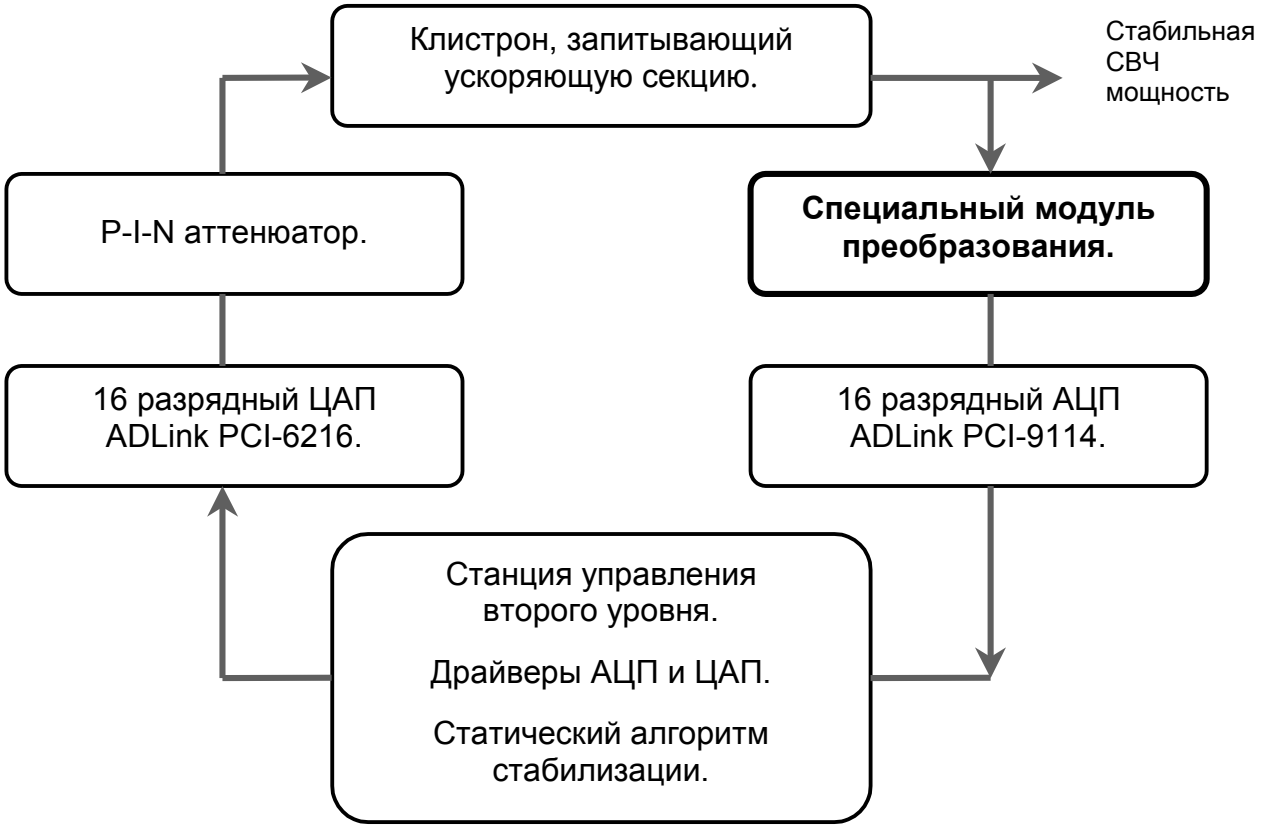


Рис. 3. Блок-схема алгоритма стабилизации СВЧ мощности.

1.3.2. Динамический модельный алгоритм.

Динамический модельный алгоритм предназначен для точной стабилизации температуры воды, охлаждающей ускоряющие секции и поворотные магниты ускорителя. Температура измеряется датчиком сопротивления и после аппаратной нормализации поступает на вход одного из каналов АЦП, переключенного в фоновый режим. Точность измерения температуры ограничена дискретностью АЦП и составляет около 0.015°C. Управление нагревом воды осуществляется с помощью драйвера мощности термонагревателей, описанного выше в пункте 1.2.

Алгоритм использует пропорционально-интегральный метод регулирования. Расчет интеграла сигнала ошибки осуществляется методом трапеций. Программный интегратор наделен свойством насыщения, что позволяет быстро восстановить управление после длительной работы на границе динамического диапазона вследствие, например, общего прогрева воды. Кроме того, обеспечивается

безударное включение алгоритма путем пересчета текущей величины уровня мощности нагревателей в значение интеграла ПИ метода.

Параметры алгоритма – постоянная времени интегратора, коэффициенты пропорциональности и общего усиления были рассчитаны, исходя из динамических характеристик отдельных устройств тракта управления, а также полного диапазона регулирования. Использование описанного алгоритма позволило достичь долговременной стабильности температуры охлаждающей воды не хуже ±0.04°C в полном диапазоне регулирования около 4°C.

2. Информационное представление объекта управления. Параметрическая технология.

2.1. Инструментарий и методика программирования систем управления.

Сегодня в области классического программирования наиболее широко применяются разновидности двух базовых технологий: процедурной и объектной. Выбирая средства разработки программ для системы управления, нужно учитывать весьма широкий диапазон реализуемых приложений – от непосредственного программирования оборудования объекта до создания развитых средств управления базами данных, человеко-машинного интерфейса и коммуникационных подсистем.

Для нижних - первого и второго уровней системы управления наиболее подходящей является структурная технология разработки. Программирование здесь ведется как бы изнутри управляемых объектов, когда доминирующей является именно процедура или функция, непосредственно реализующая заданный алгоритм управления. Дополнительным аргументом в пользу структурного подхода является наличие на нижних уровнях системы управления явно программируемых задач жесткого и мягкого реального времени.

Для верхнего уровня системы, предназначенного для взаимодействия с управляемым объектом извне - как единым целым - ситуация существенно меняется. Здесь целесообразно получать информацию о поведении отдельных подсистем или параметров объекта и манипулировать ими, основываясь на общих задачах управления, принимая во внимание информационную модель объекта как целого и с учетом правил совершения операций над конкретными параметрами. Кроме того, в отличие от нижних уровней, здесь вполне достаточно интерактивного времени отклика системы. Таким образом, для программирования верхнего уровня более подходящей становится объектная парадигма.

Задачей параметрической технологии как раз и является полное, прозрачное и взаимно согласованное представление свойств и характеристик всех подсистем объекта управления. Оно включает в себя качественное и количественное описание параметров объекта, а также правил и методов работы с ними. Таким образом формируется информационная модель управляемого объекта, а также обеспечивается интерфейс между процедурно-функциональным подходом нижних уровней и объектно-ориентированным представлением верхнего уровня. Использование параметрической методики дает возможность программировать каждый уровень системы управления применяя наиболее подходящий, максимально естественный инструментарий, не ограничиваясь рамками лишь одной технологии составления программ. Используя современную классификацию программных систем, параметрическую технологию можно считать ядром EPICS/SCADA (Experimental Physics and Industrial Control System / Supervisory for Control And Data

Acquisition) пакета, ориентированного на работу с быстро и значительно эволюционирующими объектами.

2.2. Виды параметров. Параметрические клиент и сервер.

Параметрическая технология, основные программные компоненты которой приведены на рис. 4, оперирует двумя видами параметров: стандартным и мультиданными. Стандартный вид включает в себя традиционные типы данных длиной до восьми байт (байт, целое, форматы с плавающей точкой и т.п.). Любой параметр стандартного вида может иметь до 4G (2**32) подпараметров единого формата, значения каждого из которых задаются индивидуально. Это позволяет с помощью единственного параметра описать полный набор характеристик сложного объекта. Так, в системе управления один стандартный параметр может описывать ток магнитного элемента, его ограничения, пороги установки и т.д. Параметр вида мультиданных может иметь собственный размер до 4G, что дает возможность с

Затемненные блоки являются компонентами операционной системы Linux.

Рис. 4 Основные компоненты параметрической технологии.

помощью единственного параметра определить, например, изображение объемом до 4G пиксел. В то же время, для параметров этого вида подпараметры не определены.

Два вида параметров отличаются способами обработки и интерпретации. Параметры стандартного вида полностью обслуживаются параметрическими клиентом и сервером - для них поддерживается механизм базовых транзакций. В случае мультиданных клиент обеспечивает запрос и доставку приложению соответствующего сегмента данных. Их интерпретация, обработка и сохранение являются задачей самого приложения.

При обмене данными стандартного вида один сетевой кадр может содержать несколько сегментов, каждый из которых несет полную информацию об отдельном параметре. В сегменте записаны адрес, идентификатор и статус параметра, код операции, исполняемой при его обработке, номер подпараметра и восьмибайтовый блок данных. Сегмент мультиданных может иметь различную длину и должен целиком размещаться в одном сетевом кадре. Последний содержит также заголовок в формате стандартного параметра, полностью характеризующий сегмент мультиданных. Благодаря тому, что каждый запрос, направляемый от клиента к серверу, содержит полное описание требуемых данных, программная реализация последнего заметно упрощается. Так, при обмене большими блоками данных, когда требуется проведение нескольких транзакций, серверу нет необходимости поддерживать контекст процесса обмена данными до его полного завершения – эта задача ложится исключительно на клиента.

Все операции с параметрами, в том числе передача команд, данных и статусов поддерживаются с помощью шести базовых операций. Последние, в свою очередь, обеспечиваются тремя типами транзакций, способных одновременно обрабатывать любое разумное число запросов клиента к одному или нескольким серверам.

1. Транзакция чтения. Клиент производит запрос значений или статусов параметров у станций-серверов, ожидает выработки результатов и обеспечивает их прием. Транзакция поддерживает как стандартные параметры так и параметры вида мультиданных. Сервер обслуживает запросы чтения данных асинхронно, но с меньшим приоритетом чем собственные периодические процессы управления. В транзакции задействован тайм-аут контроль.
2. Транзакция записи. Фактически является процедурой передачи данных серверам без контроля результата операции. Обслуживает параметры обоих видов.
3. Транзакция установа. После заказа новых значений параметров с помощью транзакции записи ожидает, пока серверные станции завершат все операции обслуживания этих параметров. Контроль завершения обеспечивается тем, что каждый параметр во время обработки новых данных устанавливает зарезервированное значение статуса «штатная операция». По окончании обработки статус изменяется, что приводит к выходу из транзакции установа для соответствующего параметра. Клиент производит периодический опрос статусов до тех пор, пока каждый из них хотя бы единожды не примет значение, отличное от «штатная операция». Возможен также активный режим завершения транзакции, когда сервер извещает клиента о завершении обработки с помощью выдачи активного статуса «операция завершена». В транзакции установа производится контроль тайм-аута.

Базовые операции, формирующие основу параметрической технологии, включают следующие процедуры:

1. Запрос статуса параметра. Используется транзакция чтения.
2. Запрос значения параметра. Операция осуществляет измерение и контроль. Используется транзакция чтения.

3. Запись значения параметра. Операция осуществляет управление, сопровождаемое передачей данных. Используется транзакция записи или установа.
4. Запрос записанного значения параметра. Используется транзакция чтения.
5. Предопределенное действие. Операция осуществляет управление без передачи данных. Используется транзакция записи или установа.
6. Обработка активного статуса параметра. Такой статус может быть асинхронно выдан любой станцией второго уровня в адрес верхнего уровня системы. Появление активного статуса аналогично возникновению программного сигнала и для его обработки на верхнем уровне используется специальная процедура.

Предполагается, что каждый сервер поддерживает все базовые операции и выполняет соглашения по типам поддерживаемых транзакций. Так, если какой-либо параметр может быть записан в станцию, то непременно обеспечивается возможность чтения записанного значения. Транзакции установа обязательно поддерживаются как для основных значений (нулевой номер подпараметра) всех параметров стандартного вида, так и для каждого предопределенного действия. В базовых операциях, осуществляющих управление, тип транзакции выбирается соответственно потребностям приложения. Например, для интерактивного режима вполне достаточно транзакции записи, а при работе сценариев как правило необходимы транзакции установа.

2.3. Особенности реализации.

При реализации параметрического сервера подсистемы управления используется ряд дополнительных соглашений. Так, в подсистеме резервируется определенное число параметров стандартного вида, обладающих собственным статусом. Они подразделяются на группы общих параметров, которые могут использоваться приложениями произвольным образом и системных параметров, описывающие объекты, определенные во всех подсистемах управления: драйверы стандартных устройств, процедуры обмена данными по сети, состояние общесистемных компонент программы и т.п. Для каждого из записываемых параметров резервируются подпараметры, фиксирующие границы диапазона управления. Значения статусов параметров также подразделяются на группы. Статус может указывать на отсутствие параметра в подсистеме, являться активным либо пассивным, принадлежать к группе общих, системных, специальных. Активные статусы передаются на верхний уровень по инициативе подсистем, пассивные лишь фиксируются и могут быть извлечены операцией запроса статуса параметра. Вместе взятые, эти соглашения позволяют реализовать развитый набор программ верхнего уровня, в частности, технологию сценариев.

Распределение параметров объекта управления между станциями второго уровня как правило производится таким образом, чтобы минимизировать либо совсем исключить необходимость обмена данными между отдельными подсистемами в режиме реального времени. Тем не менее, пересылка данных может понадобиться, если одна из подсистем периодически выполняет процедуры, требующие результатов измерения параметров в других подсистемах. Для осуществления такой возможности станция верхнего уровня поддерживает прозрачную коммутацию данных между подсистемами второго уровня. Если адрес назначения параметра отличен от адреса станции верхнего уровня, она перенаправляет этот параметр получателю.

Станция верхнего уровня также контролирует базовую функциональность подсистемем второго уровня. Для этого производится периодический запрос статуса выделенного параметра, определенного в каждой из подсистем.

3. Верхний уровень системы управления.

Программная структура верхнего уровня системы управления изображена на рис. 5. Ее основными компонентами являются:
• модуль базовых операций;
• база данных системы управления;
• модуль поддержки встроенных и внешних сценариев, а также внешних процедур;
• системный человеко-машинный интерфейс;
• интерфейсы оператора-пользователя и связи с внешним миром.

3.1. Описание свойств параметров. База данных системы управления.

В параметрической технологии каждая операция или процедура в системе управления определяется одним либо несколькими параметрами со своими свойствами и значениями. Полное описание параметров сохраняется в реляционной базе данных Interbase (производство фирмы Borland /5/). Она обеспечивает стандартные манипуляции с таблицами, обрабатывая SQL запросы от программы верхнего уровня, которая, в свою очередь, выполняет необходимые макро-операции. Так, при редактировании записей основных таблиц соответственно корректируются и нормализованные таблицы. Некоторое состояние или сценарий объекта могут служить прототипом другого состояния/сценария. А при изменении значений параметров объекта новые величины фиксируются в базе данных.

База включает в себя таблицы описания объектов управления, свойств и правил параметров, таблицы для регистрации их состояний и величин:
• таблицы объектов и станций управления;
• таблицы групп и классов параметров;
• таблица параметров объекта;
• таблицы предопределенных действий и описания статусов параметров;
• таблица массивов значений параметров;
• журнал активных статусов;
• дополнительные таблицы, используемые при работе системного интерфейса.

3.1.1 Таблица объектов.

Система управления может поддерживать одновременно несколько различных объектов. Так, в нашей реализации обеспечивается управление тремя ускорителями и одним испытательным стендом. Данная таблица содержит описания отдельных объектов системы управления. Каждая группа параметров, управляющая станция или панель управления привязана к конкретному объекту. Функция выбора активных объектов блокирует все элементы системы управления, не связанные с этим объектом, а также определяет порядок работы с массивами сохраненных значений.

3.1.2 Таблица станций управления.

Формирует описание управляющих станций второго уровня. Включает в себя следующие основные записи:
• Имя и индекс станции.
• Объект, к которому относится станция.
• Имя хост-компьютера, на котором работает программное обеспечение станции.
• Переключатель, позволяющий временно блокировать неиспользуемые станции.
• Описание станции управления.

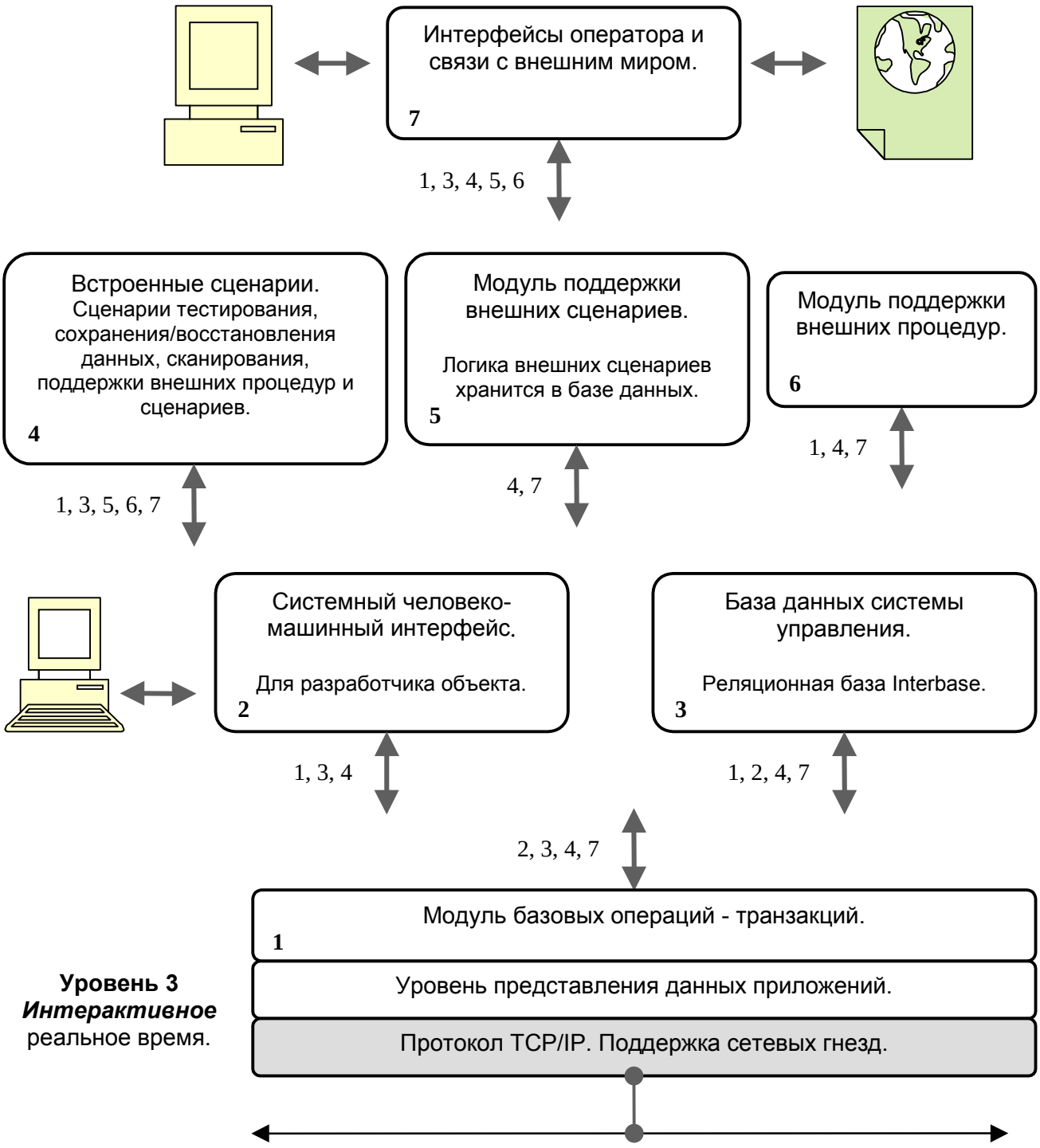


Рис. 5. Программная структура верхнего уровня системы управления.

3.1.3 Таблица групп параметров.

Используется для логической группировки параметров. Данные этой таблицы обеспечивают также выделение и блокирование параметров, связанных с неактивными в данный момент объектами. Содержит следующие поля:

• Имя группы и объект, к которому относится данная группа.
• Описание группы.

3.1.4 Таблица классов параметров.

Основное назначение таблицы классов (рис. 6) – выделение и описание общих свойств и правил совокупности параметров системы, таких как режимы чтения/записи, единицы измерения, возможные значения статусов, а также определение подпараметров и предопределенных действий. Классификация предотвращает многократное дублирование информации при описании каждого параметра. Таблица включает в себя следующие основные записи:

• Имя класса.

• Alias или "прозвище" класса. Используется, в частности, подсистемой сценариев. Каждый класс должен обладать уникальным "прозвищем".

• Единицы измерения для параметров данного класса. Это поле может быть пустым.

• Тип параметра. Характеризует базовые свойства параметра, исходя из которых выбирается алгоритм для его обслуживания. В действующей системе реализованы два типа параметров: абсолютные и относительные. Для абсолютного параметра можно задать результирующее значение, выраженное в его единицах измерения. Для относительного - смещение значения параметра от его текущего состояния.

• Формат параметра (байт, целое, плавающая точка, строка, логическое значение и др.). Он распространяется как на все подпараметры стандартного вида, так и на каждое значение вида мультиданных.

• Размер параметра. Когда параметр имеет единичный размер, он относится к стандартному виду. Если же размер параметра превышает единицу, он представляется видом мультиданных и поле определяет его размер в единицах формата параметра.

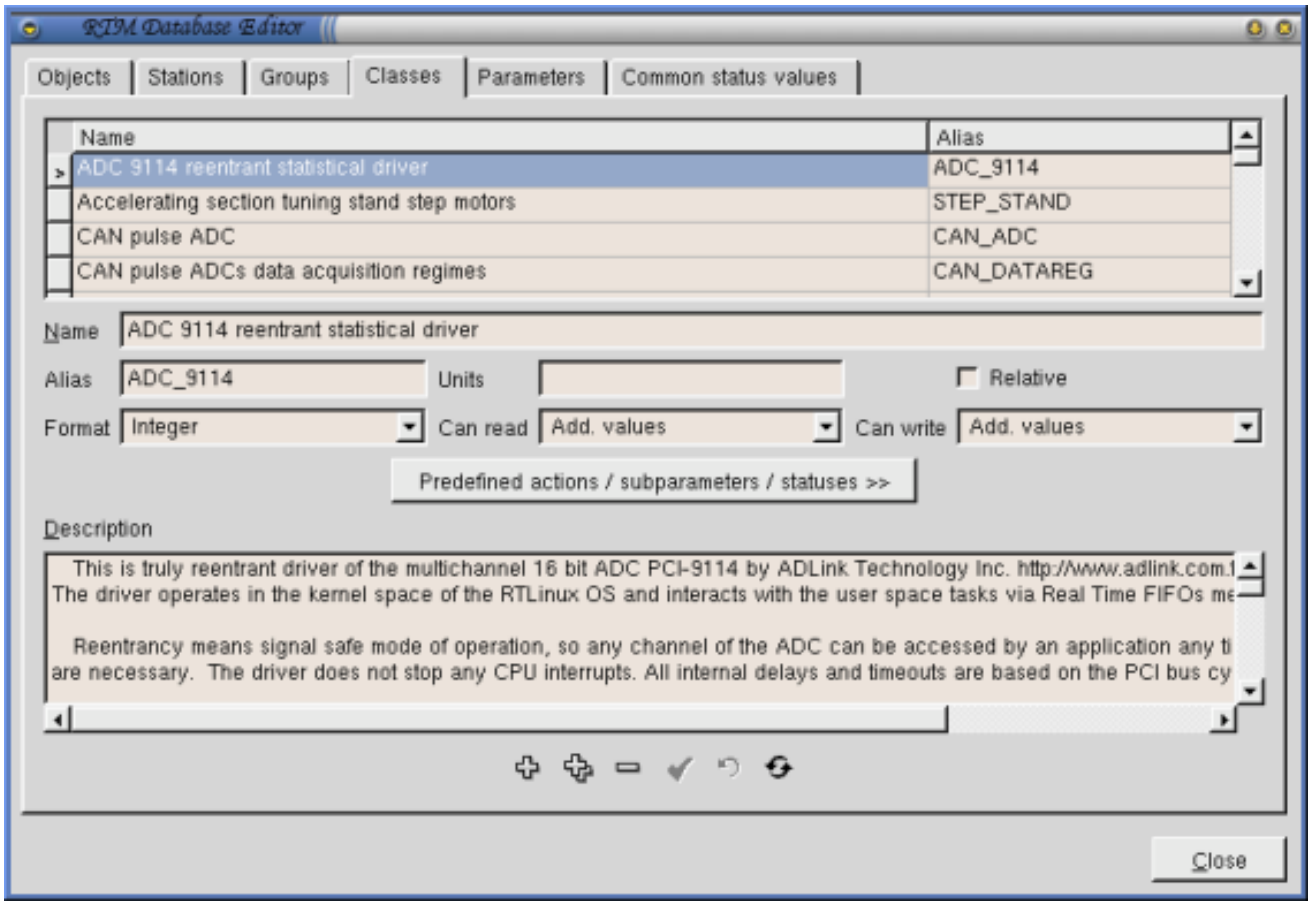


Рис. 6 Интерфейс таблицы классов.

• Допустимые операции чтения и записи для основного и дополнительных значений параметров стандартного вида.
• Описание класса.

Здесь же вызываются таблицы предопределенных действий, подпараметров и статусов, относящихся к данному классу.

Таблица предопределенных действий включает в себя следующие поля:
• Индекс предопределенного действия, передаваемый управляющей станции.
• Alias ("прозвище") предопределенного действия. Используется в подсистеме сценариев. Каждое предопределенное действие должно обладать уникальным для данного класса "прозвищем".
• Имя предопределенного действия.

Таблица подпараметров состоит из записей:
• Индекс дополнительного значения, передаваемый управляющей станции.
• Alias ("прозвище") дополнительного значения. Используется в подсистеме сценариев. Каждое дополнительное значение должно обладать уникальным для данного класса "прозвищем".
• Имя дополнительного значения.

Таблица статусов определяет индивидуальные статусы для всех параметров класса. Статус может принимать как значения, определенные в данной таблице, так и значения из таблицы общесистемных статусов.

3.1.5 Таблица параметров объекта управления.

В таблице параметров (рис. 7) формируется окончательная информационная модель объекта в том виде, как она представляется разработчику системы

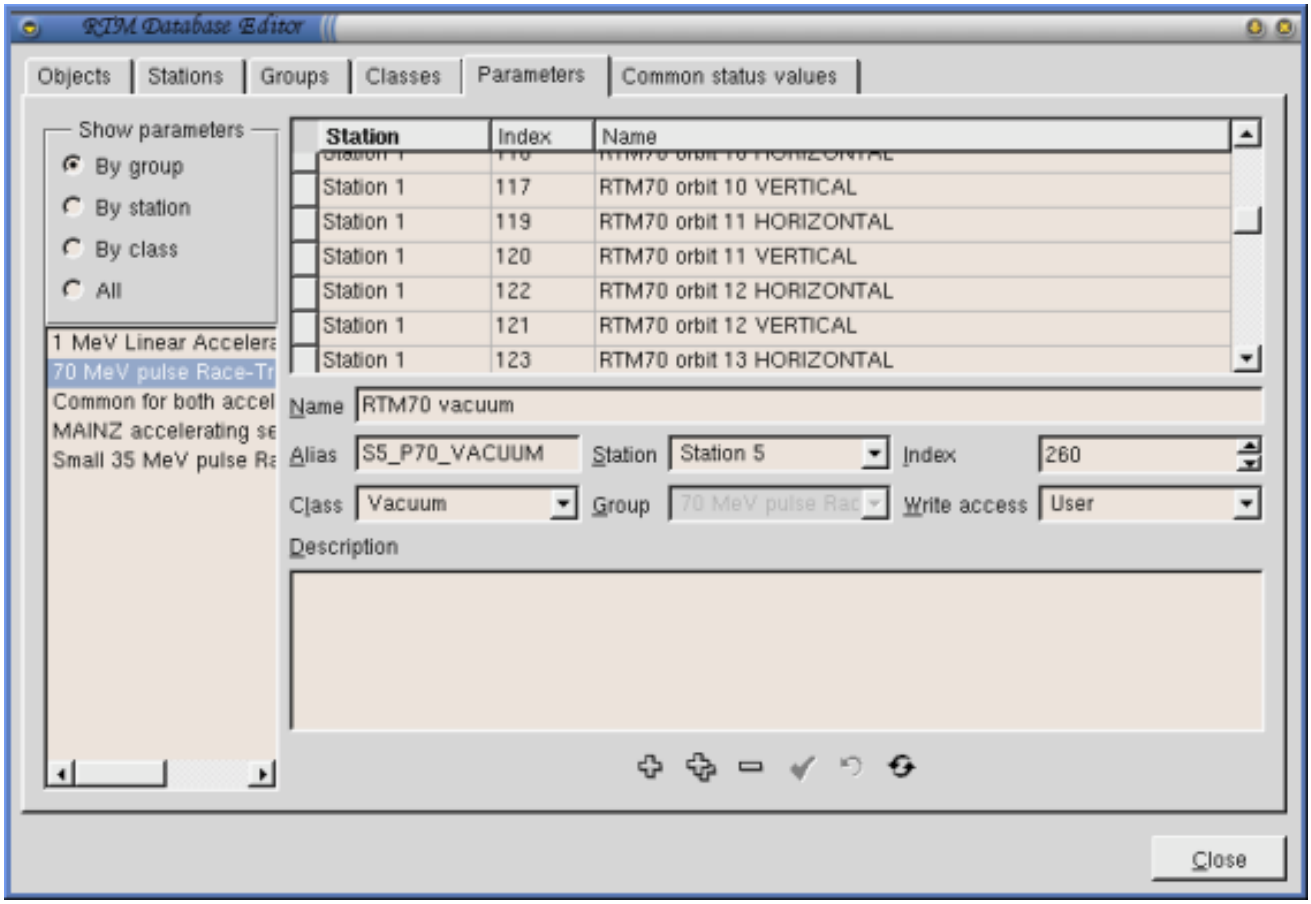


Рис. 7. Интерфейс таблицы параметров.

управления. Эта таблица служит основой для последующего конструирования различных интерфейсов оператора, формирования сценариев объекта и т.п.

Таблица содержит следующие основные записи:

• Имя параметра.
• Alias ("прозвище") параметра. Используется, в частности, в подсистеме сценариев. Каждый параметр должен обладать уникальным "прозвищем".
• Станция управления второго уровня, поддерживающая данный параметр.
• Индекс параметра, передаваемый управляющей станции.
• Класс и группа, к которым относится параметр.
• Описание параметра.

Важно отметить, что в нашей системе не осуществляется автоматическая поддержка согласованности информации, размещаемой в базе данных и той, которой располагают параметрические серверы подсистем управления. Это принципиальное решение, обусловленное первичностью информации, содержащейся в базе данных и формирущей единое общесистемное техническое задание, которое определяет, что именно должно происходить при различных манипуляциях с параметрами, в то время как параметрические серверы фактически исполняют соответствующие операции. В идеальном случае каждая выполняемая сервером операция должна полностью соответствовать ожидаемой, однако в большой, многопараметрической, активно эволюционирующей системе достичь такой согласованности чрезвычайно сложно. При этом навязывание со стороны параметрических серверов подсистем управления собственной интерпретации свойств и значений параметров было бы некорректным. В то же время, система управления должна поддерживать некоторый инструментарий для проверки и указания соответствующих расхождений. С этой целью на верхнем уровне реализуется необходимый набор сценариев.

3.2. Сценарии верхнего уровня.

Параметрическая технология поддерживает два типа сценариев: встроенные и внешние. Встроенные сценарии непосредственно входят в состав программного обеспечения верхнего уровня и выполняются в соответствии с собственными свойствами параметров. Внешние сценарии формируются по правилам выполнения внешних операций и реализуются соответствующим встроенным сценарием с учетом свойств параметров. Ниже приведены основные встроенные сценарии верхнего уровня.

3.2.1. Сценарий проверки согласованности описания параметров.

1. Сценарий выполняет проверку соответствия параметров базы данных и параметрических серверов подсистем второго уровня.

• Для каждой записи таблицы параметров запрашивается значение статуса. Если возвращаемый станцией-сервером статус указывает на отсутствие параметра, фиксируется нарушение соответствия.
• Запрашивается описание каждого определенного в сервере параметра. Оно включает в себя значения класса параметра, его формата и размера. Станция верхнего уровня сравнивает данные, поступившие из подсистемы, с соответствующим определением из базы данных и принимает решение о согласованности двух описаний параметра.
• Для каждой подсистемы второго уровня запрашиваются статусы всех зарезервированных параметров (как правило, это число близко к одной тысяче).

Если обнаружен параметр с нормальным значением статуса, но отсутствующий в таблице параметров базы данных, также фиксируется несогласованность описаний.

3.2.2. Сценарии поддержки управления.

1. Сценарий сохранения и восстановления состояния объекта.

В настоящее время реализован базовый вариант этого сценария, манипулирующий только с основными значениями параметров стандартного вида. Он позволяет единовременно сохранять и загружать значения всех активных параметров. Сохраняемые значения организуются в именованные массивы.

Базовая версия сценария имеет два ограничения. Во-первых, не производится работа с дополнительными значениями параметров которые, как правило, задают различные конфигурационные величины, определяют границы диапазона управления для основного значения и т.п. Во-вторых, не задается последовательность восстановления сохраненных значений – все они записываются в станции второго уровня одновременно. Тем не менее, опыт реальной повседневной работы с ускорителями показал, что указанные ограничения не являются существенными.

2. Сценарий линейного сканирования.

В связи с особым удобством использования метода сканирования при настройке объекта, было решено реализовать его в качестве встроенного сценария (рис. 8). Сканирование подразумевает линейное приращение величины выбранного параметра в пределах заданного диапазона с определенным шагом, установ нового значения и последующее измерение некоторого параметра.

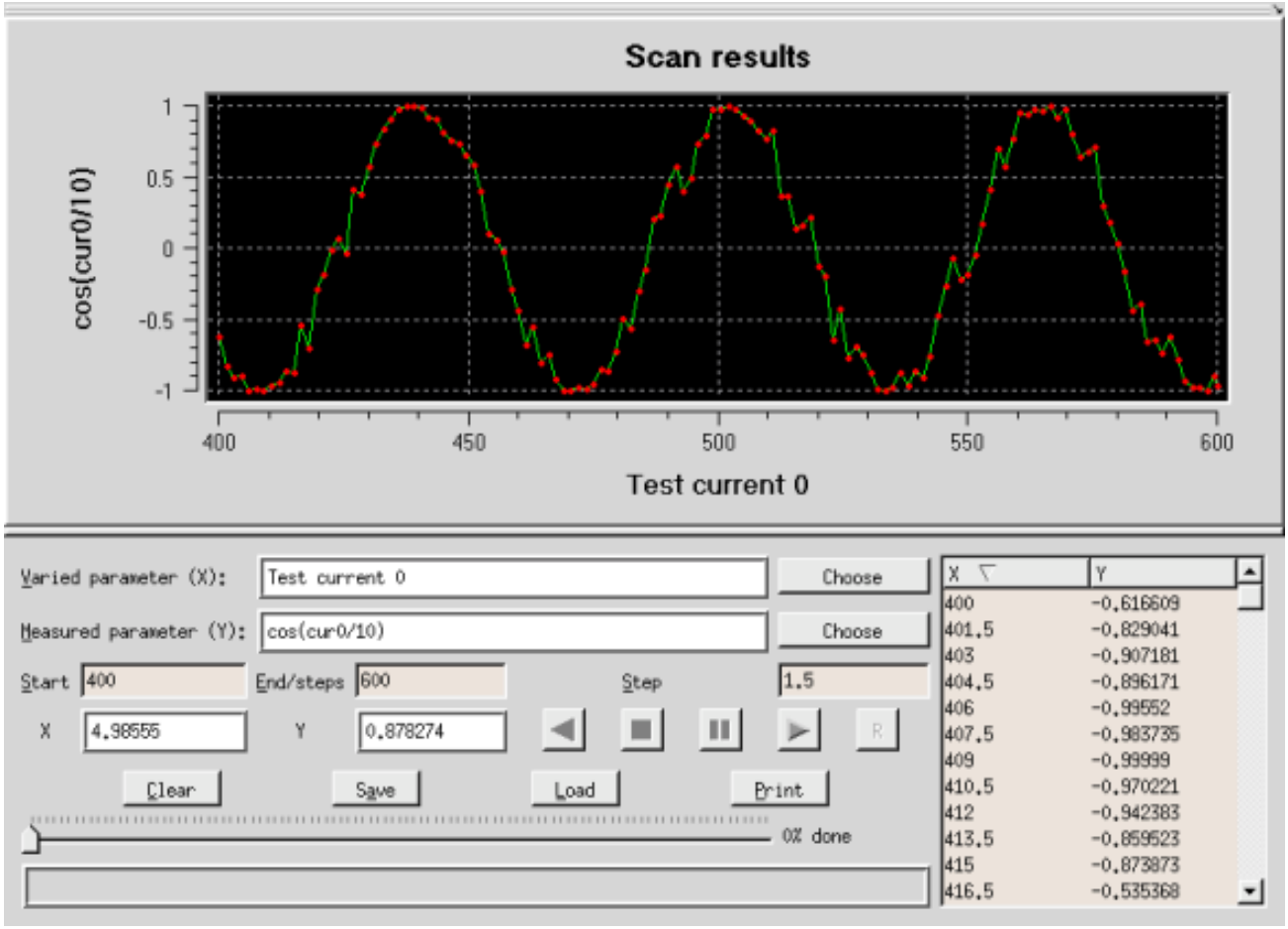


Рис. 8. Интерфейс сценария линейного сканирования.

• Сценарию передаются необходимые аргументы: параметр сканирования, измеряемый параметр, границы диапазона и шаг сканирования.
• С помощью транзакции установа выполняется запись очередного значения сканируемого параметра. Если при этом возникают ошибки или тайм-аут, работа сценария завершается.
• Запрашивается значение измеряемого параметра, которое выводится на график сканирования и записывается в буфер сохранения.
• Осуществляется приращение шага сканирования. При достижении параметром границы диапазона работа заканчивается.
• При любом завершении сценария восстанавливается первоначальная величина параметра.

По окончании сканирования данные из буфера сохранения могут быть записаны в файл и использованы для дальнейшей обработки. Поддерживается также загрузка сохраненных значений с графическим отображением результата сканирования.

3.2.3. Сценарии поддержки внешних заданий.

Основной задачей этих сценариев является формирование такого логического представления управляемых объектов, когда внешние задания взаимодействуют с ними также, как это происходит при проведении расчетов с использованием модельных функций. Единственным изменением, которое необходимо внести, является замена вызовов этих функций обращением к соответствующему сценарию.

Если последовательность установки параметров объекта не имеет значения, используется следующий сценарий:
• Прикладная задача передает сценарию "прозвища" параметров, соответствующих аргументам и значениям модельной функции, а также требуемые величины аргументов.
• Производится инициализация транзакции установа для всех параметров-аргументов.
• Осуществляется чтение и возврат прикладной задаче величин параметров, определяющих значения модельной функции.

В тех случая, когда последовательность установки параметров существенна, активируется дополнительный сценарий, сформированный для задания аргументов данной модельной функции. В этом случае после передачи сценарию параметров производятся следующие действия:
• Значения аргументов заносятся в выделенную запись таблицы массивов значений параметров.
• Запускается на выполнение дополнительный сценарий, использующий записанные значения параметров-аргументов. Он инициализирует транзакции установа в определенной последовательности для отдельных подгрупп параметров.
• По завершении работы дополнительного сценария производится чтение и возврат прикладной задаче величин параметров, соответствующих значениям модельной функции.

В настоящее время исследуется возможность использования для формирования сценариев языка "очень высокого уровня" Python /6/.

3.3 Интерфейсы верхнего уровня.

Доступ к системе управления может осуществляться посредством двух типов интерфейсов. В системном интерфейсе, используемом разработчиками объекта, поддерживаются все возможности базы данных и сценариев, обеспечивая простой,

но максимально полный доступ к параметрам и операциям. Интерфейс пользователей-операторов, основанный на интерактивной графической оболочке, наиболее удобен при штатной эксплуатации объекта. Оба интерфейса используют технологию X-Windows и разработаны с помощью компилятора C++ и графической библиотеки QT фирмы Trolltech /7/.

3.3.1. Системный интерфейс.

Системный человеко-машинный интерфейс верхнего уровня поддерживается рядом окон, оформленных в виде таблиц:

• основные рабочие окна параметров;
• диалоговые окна для формирования наборов параметров рабочих окон.
• окна установа дополнительных значений и активации предопределенных действий;
• окно управления массивами сохраненных значений;
• окна управления подключением станций и журнала статусов;

Основным "органом управления" системного интерфейса являются рабочие окна параметров (рис. 9), где отображаются в виде таблицы следующие поля:

• Имя параметра;
• Считанное (измеренное) значение параметра (для считываемых параметров). Это значение обновляется с небольшим временным интервалом для всех видимых параметров в таблице;
• Задаваемое значение параметра (только для записываемых параметров);

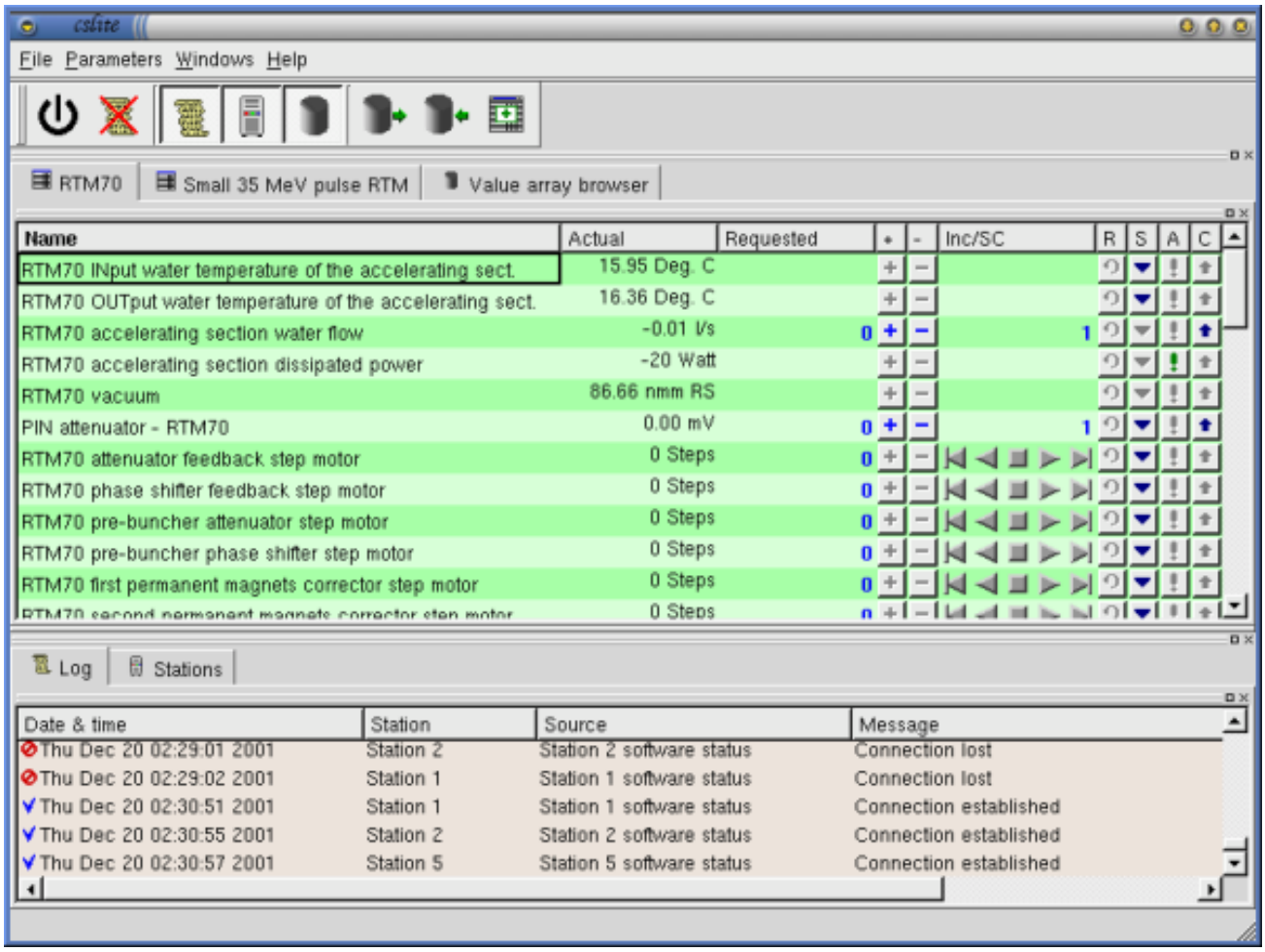


Рис. 9. Вид рабочего окна параметров.

• Инкремент и декремент значения параметра: кнопки + и −; только в случае абсолютного записываемого параметра;
• Панель управления относительным параметром. Состоит из кнопок управления смещением: ⏮ - влево до упора, ◀ - влево, ■ - остановка, ▶ - вправо, ⏭ - вправо до упора.

Кроме этого, имеются следующие кнопки:

для записываемых параметров - откат и история. Для каждого параметра ведется история значений. Текущее значение добавляется в историю до первого изменения величины параметра в новой серии управления.

для параметров с дополнительными значениями. Нажатие этой кнопки вызывает окно для просмотра и/или редактирования дополнительных значений параметра.

для параметров с предопределенными действиями. Вызывает всплывающее окно активации предопределенных действий.

для абсолютных записываемых параметров. Служит для повторной отправки задаваемого значения на управляющую станцию.

При формировании наборов параметров рабочих таблиц используется дополнительное диалоговое окно (рис. 10), позволяющее выбирать отображаемые параметры. Оно включает следующие компоненты:
• Выбор типа группирования параметров. Возможно отображение параметров только из указанной группы или относящихся к определенной станции или параметров указанного класса либо всех параметров.

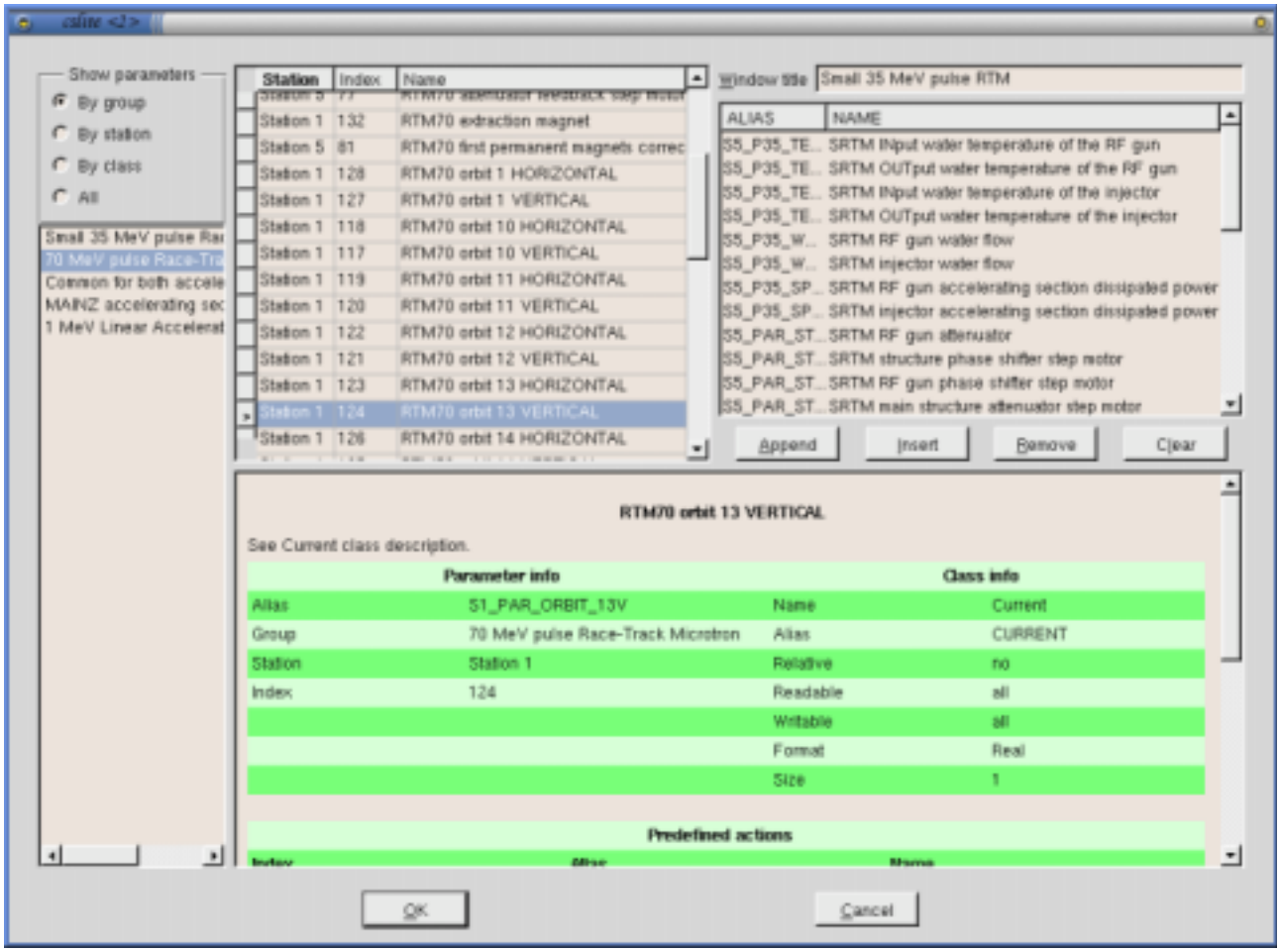


Рис. 10. Формирование набора параметров рабочей таблицы.

• Выбор группы/станции/класса. Здесь можно выбрать группу, станцию или класс параметров, которые должны быть отображены в списке выбора, в зависимости от типа группирования.
• Список выбора параметров. Здесь представлен список всех параметров из базы данных, удовлетворяющих заданным критериям (группа/станция/класс или все параметры).
• Список отобранных в рабочую таблицу параметров. Отображает параметры, внесенные оператором в формируемую рабочую таблицу.

3.3.2. Интерфейсы оператора и связи с внешним миром.

Интерфейс оператора-пользователя является надстройкой над системным интерфейсом и использует различные сервисы, предоставляемые последним. Основной чертой интерфейса оператора является замена табличного представления отдельных параметров объекта на их отображение с использование анимированных графических изображений.

Связь системы управления с внешним миром обеспечивается соответствующими компонентами операционной системы Linux. Так, с использованием стандартных утилит ssh daemon и ssh client возможен запуск приложений верхнего уровня с удаленной машины. Эта возможность активно используется при отладке ускорителей для управления отдельными параметрами непосредственно с места размещения объекта.

4. Заключение.

Система управления электронных ускорителей полномасштабно эксплуатируется в течении двух лет. Программные решения, используемые при проектирование всех ключевых компонент системы – драйверов устройств, алгоритмов управления, информационной модели и интерфейсов объектов управления – продемонстрировали свою состоятельность и добротность. Созданная система легко адаптируется к эволюции действущих ускорителей и расширяется для управления новыми объектами.

Авторы выражают искреннюю благодарность руководителю проектов создания ускорителей доктору физ.-мат. наук В.И. Шведунову, без участия которого проведение данной работы было бы невозможно, а также профессору W.P. Trower (World Physics Technologies Inc.) за поддержку настоящего проекта.

Литература.

1. I.V. Gribov, I.V. Shvedunov and V.R. Yalijan.
   World Physics Technologies, Blacksburg, VA 24060 USA
   “RaceTrack Microtron Control System”
   2001 Particle Accelerator Conference, Chicago, Illinois, June 18-22, 2001.
   http://pacwebserver.fnal.gov/papers/Monday/PM_Poster/MPPH024.pdf
2. F. Nedeoglo, O. Novojilov, S. Dudnikov, A. Chepurnov, I. Gribov, V. Shvedunov.
   “Distributed CAN-bus based beam diagnostic system for pulsed racetrack microtron”.
   Proceedings of the International Conference on Accelerator and Large Experimental Physics Control Systems. San Jose, CA, USA, Nov. 27-30, 2001. FRAT003.
3. Chepurnov A.S., Gribov I.V., Morozov S.Yu., Shumakov A.V., Zinoviev S.V.
   “Moscow University race-track microtron control system: ideas and development”.
   Proceedings of the International Conference on Accelerator and Large Experimental Physics Control Systems. KEK, Tsukuba, Japan, Nov. 11-15, 1991, p. 40-142.
4. Г. Корн и Т. Корн
   “Справочник по математике для научных работников и инженеров”.
   М., Наука, 1974 г., с. 654.
5. Borland Home Page. http://www.borland.com/
6. Python Language Website http://www.python.org/
7. Trolltech. http://www.trolltech.com/

Игорь Валерьевич Грибов
Иван Васильевич Шведунов
Василий Робертович Яйлиян

ТЕХНОЛОГИЯ СОЗДАНИЯ СИСТЕМЫ УПРАВЛЕНИЯ СОВРЕМЕННЫМИ УСКОРИТЕЛЯМИ ЭЛЕКТРОНОВ.

Препринт НИИЯФ МГУ – 2002-17/701

Работа поступила в ОНТИ 25 июля 2002 г.

ИД № 00545 от 06.12.1999

Издательский отдел
Учебно-научного центра довузовского образования

117246, Москва, ул. Обручева, 55А
119992, Москва, Ленинские горы, ГЗ МГУ, Ж-105а
Тел./факс (095) 718-6966, 939-3934
e-mail: izdat@abiturcenter.ru
http://www.abiturcenter.ru

Гигиенический сертификат № 77.99.2.925.П.9139.2.00 от 24.02.2000
Налоговые льготы – Общероссийский классификатор продукции
ОК-005-93, том 1 – 953000

Подписано в печать 04 сентября 2002 г. Формат 60x90/16
Бумага офсетная № 2. Усл. Печ. л. 1.56
Тираж 30 экз. Заказ № 179

Отпечатано в Мини-типографии УНЦДО