ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ДЛЯ АВТОМАТИЗАЦИИ ИССЛЕДОВАТЕЛЬСКИХ УСТАНОВОК

А.В. Курякин, Ю.И.Виноградов
Российский Федеральный Ядерный Центр – Всероссийский Научно-Исследовательский Институт Экспериментальной Физики (РФЯЦ-ВНИИЭФ), 607188 Саров, Нижегородская обл., Россия

Обсуждаются особенности и требования к программному обеспечению систем автоматизации исследовательских физических установок. Описаны принципы построения программного пакета CRW-DAQ, который использовался для автоматизации ряда исследовательских установок во ВНИИЭФ, ОИЯИ и ЦЕРН.

Введение

Задачи сбора данных и автоматизации необычайно многообразны. При этом различные области применения предъявляют различные требования, как к аппаратному, так и к программному обеспечению. Эти различия связаны не только с техническими характеристиками установок, но и с условиями их эксплуатации, технической поддержки, характером задач, экономическими возможностями. Например, требования к системе промышленной автоматизации сильно отличаются от требований к исследовательской установке.

Основной задачей промышленности является бесперебойный выпуск той или иной продукции. Поэтому для промышленной автоматизации характерна работа с постоянным набором стандартного оборудования, условий эксплуатации и кругом задач. Обычно используется однородный по составу набор измерительных устройств, чаще всего одной фирмы. Вопрос о стоимости измерительного оборудования и программного обеспечения остро не стоит, так как это одноразовые долгосрочные затраты, ничтожные по сравнению со стоимостью выпускаемой продукции. Аппаратное и программное обеспечение может эксплуатироваться длительное время с минимальными изменениями, что снижает стоимость их технической поддержки. Для систем промышленной автоматизации создано множество программных пакетов, таких как LabView, Genesis и т.д. Эти пакеты доказали на практике свою пользу, высокую надежность, удобство. Надо отметить, что важную роль в промышленности играет юридическая сторона вопроса, наличие сертификата на оборудование и программное обеспечение. Поэтому коммерческие программные пакеты для промышленной автоматизации предпочтительнее как собственных разработок, так и систем с открытым кодом, где юридические гарантии чаще всего отсутствуют.

В задачах автоматизации исследовательских установок условия совершенно другие. Основной целью исследовательской деятельности является получение нового знания. Поэтому процесс исследования в своей природе содержит элемент непредсказуемости. Это неизбежно отражается и на характере задач автоматизации исследовательских установок, которые становятся существенно более динамичными.

1. Особенности автоматизации исследовательских установок.

При активном участии авторов за последние годы (примерно 1997-2004 г.) для проведения различных экспериментов в ВНИИЭФ (Саров), ОИЯИ (Дубна), СПГУ (Санкт-Петербург), ЦЕРН (Швейцария/Франция), был создан ряд систем сбора данных и автоматизации для исследовательских физических установок. Большая часть этих исследовательских установок относится к области тритиевых технологий. Среди автоматизированных установок стоит особо упомянуть:

- ПРОМЕТЕЙ – установка для изучения проницаемости и диффузии изотопов водорода, ВНИИЭФ, Саров [1];
- ТРИТОН – тритиевая мишень для исследования μ-катализа на фазотроне ЛЯП ОИЯИ, Дубна [2];
- АКУЛИНА – тритиевая мишень для исследования нейтронно-избыточных систем на циклотроне ЛЯР ОИЯИ, Дубна [3].

Накопленный позволяет выявить ряд характерных особенности систем автоматизации для исследовательских установок. Так, измеряемые при исследованиях величины отличаются большим разнообразием и широким диапазоном. Например, на установках ТРИТОН и АКУЛИНА приходится измерять температуры в диапазоне от 10 К (жидкий водород) до 1500 °С, давления – от глубокого вакуума до тысяч атмосфер, объемные активности от 10^{-7} до 10^{4} Ки/л. К этому добавляется спектрометрия, хроматография, ядерно-физические измерения и т.д.

Разнообразие измеряемых величин вынуждает использовать оборудование разных фирм, а также специально созданное под данную задачу оборудование, так как ни одна фирма не может обеспечить требуемый набор измерительных средств. Поэтому практически все исследовательские установки по составу гибридные, например, на установках ТРИТОН и ПРОМЕТЕЙ используется продукция ICP-DAS, Balzers, Advantech, а также оригинальные устройства разработки ВНИИЭФ.

Как правило, аппаратное и программное обеспечение модифицируется под каждый конкретный эксперимент или серию измерений. Это связано с самим характером исследовательских задач – исследователь старается охватить как можно более широкий круг явлений, к тому же круг задач меняется в результате самого процесса исследования.

При решении исследовательских задач управляющий алгоритм часто заранее не известен, а вырабатывается постепенно в процессе эксплуатации. С этим связана существенно более высокая трудоемкость создания программного обеспечения исследовательских установок по сравнению с другими типами программного обеспечения.

Для исследовательских установок характерно, что аппаратурные драйверы пишутся не под устройство, а под конкретную задачу. Типовые драйверы, даже для стандартных “фирменных” устройств, могут обеспечить только самые простые виды измерений, что не всегда подходит. Например, типовые драйверы АЦП PCL-818L могут обеспечить высокоскоростной буферизованный ввод данных. Если же по результатам измерений надо, например, выработать мгновенную реакцию, выдаваемую на исполнительное устройство при помощи ЦАП, придется писать драйвер и для АЦП и для ЦАП специально под данную задачу.

Использование специализированного оборудования и необходимость частой корректировки алгоритмов предъявляет к программному обеспечению повышенные требования с точки зрения возможности быстрой и качественной модификации программ. Поэтому авторы считают, что для решения исследовательских задач практически обязательным является наличие открытых исходных кодов программ, а также крайне желательно наличие встроенных языков программирования для быстрого создания управляющих алгоритмов.

Оборудование в исследовательских установках часто работает в экстремальных и, иногда, опасных условиях. Поэтому предъявляются высокие требования к надежности и предсказуемости работы как аппаратного, так и программного обеспечения системы автоматизации.

Малые исследовательские коллективы не имеют возможности специализированного обучения программистов, что совершенно необходимо для создания предсказуемых и безопасных систем на основе закрытого коммерческого программного обеспечения. Это тоже весомый аргумент в пользу открытого программного кода, который, в конце концов, всегда можно прочитать и понять. Это исключительно полезно даже в том случае, если не

ставится задача его модификации. Например, наличие полных исходных кодов Runtime библиотек Borland Pascal, Delphi позволило существенно повысить надежность работы программного обеспечения. При этом никаких коррекций в “фирменный” код не вносилось, он использовался только как наиболее полное справочное пособие по программированию.

Авторы скептически относятся к применению в задачах автоматизации технологий типа “черного ящика”. Такие технологии преследуют, прежде всего, коммерческие цели, позволяя производителям сохранять в тайне реализацию программ, предоставляя только интерфейс. Это, наверное, приемлемое решение для офисных приложений, но не для задач автоматизации, где детали реализации, например, нежелательные задержки, могут иметь решающее значение. Сокрытие информации о реализации (основная идея “черного ящика”) никак не способствует повышению предсказуемости работы программ.

Юридические вопросы, связанные с сертификацией, не играют решающей роли в исследовательской области. Достоверность научных результатов принято подтверждать рациональными аргументами и результатами калибровочных измерений, а не наличием сертификата на применяемое оборудование и программное обеспечение.

Применение коммерческого программного обеспечения не приводит автоматически к возможности сократить число программистов и связанных с ними расходов. Все равно на практике каждая исследовательская команда обязательно включает специалиста по программному обеспечению, так как характер исследовательских задач не позволяет сделать установку “раз и навсегда”, а требует тесного взаимодействия исследователя с программистом.

В силу перечисленных обстоятельств, применение стандартных пакетов промышленной автоматизации для исследовательских установок представляется авторам не целесообразным. Эти пакеты дороги, ориентированы на стандартное оборудование, закрыты с точки зрения исходного кода. Авторы считают оптимальным для исследовательских установок вариантом использование открытого программного обеспечения, либо создание собственного программного обеспечения.

2. Принципы построения программного обеспечения исследовательских установок.

Приняв решение использовать для задач автоматизации установок собственное программное обеспечение, авторы видели два пути – создавать “с нуля” небольшие программы под каждую установку отдельно, либо написать специализированный пакет, который можно использовать на всех установках подобного класса, дополняя его относительно небольшим набором прикладных программ, индивидуальным для каждой установки. Оба подхода имеют свои плюсы и минусы. Программу под конкретную установку можно сделать очень эффективной и удобной, но это займет много времени, сопровождать ее будет сложно, причем проблемы сопровождения возрастают с ростом числа установок. Специализированный пакет требует больших начальных затрат, но потом программы для конкретных установок легче создавать и сопровождать. На практике использовались оба подхода, но предпочтение все же отдавалось решению задач в рамках специализированного пакета.

В результате в течение ряда лет, в процессе решения практических измерительных задач, авторы разработали специализированный пакет для автоматизации исследовательских физических установок, названный CRW-DAQ**.** С использованием этого пакета и было автоматизировано значительное число исследовательских установок. Только в ряде особых случаев программное обеспечение разрабатывалось “с нуля” под конкретный прибор или эксперимент. В настоящее время пакет CRW-DAQ существует в двух версиях - CRW16 для DPMI и CRW32 для Windows-95/98/NT/2K/XP. Архитектура пакета позволяет довольно легко реализовать его в любой другой операционной системе, хотя практической необходимости в этом пока не возникало.

В основу программного обеспечения, созданного авторами для автоматизации исследовательских физических установок, положены следующие принципы.

Разделение программного обеспечения на системное и прикладное. Системное программное обеспечение, общее для всех установок, предоставляет графический интерфейс, языковые средства для прикладного программирования и другие общие сервисы. Прикладное программное обеспечение, индивидуальное для каждой установки, использует предоставленные средства для решения конкретной задачи. Такое разделение имеет следующие преимущества:

- Системное программное обеспечение одинаково на всех установках. Его становится легче поддерживать и обновлять, что позволяет сосредоточить усилия на повышении его надежност,. В качестве системного обеспечения выступает собственно пакет CRW-DAQ.
- Системное программное обеспечение предоставляет прикладным программам встроенные языки программирования с хорошо защищенным программным интерфейсом. Это снижает влияние возможных ошибок в прикладном программном обеспечении на надежность системы в целом.
- Прикладное программное обеспечение, включающее конфигурационные файлы и прикладные программы на встроенных языках, обеспечивает необходимую гибкость программного обеспечения. Так как оно написано на встроенном языке программирования, его легко модифицировать. Более того, система позволяет модифицировать его прямо в процессе измерений.

Приоритетная многопоточность. Измерительные программы построены в виде нескольких параллельно выполняемых программных потоков, приоритеты которых могут различаться. При правильном применении система приоритетов позволяет гарантировать выделение положенного кванта времени критически важным потокам, даже при высокой загрузке процессора.

Высокая степень параллелизма. В типичной измерительной системе CRW-DAQ может работать несколько десятков программных потоков. За счет высокой степени параллелизма каждая из параллельно выполняемых прикладных задач может быть сделана относительно простой. Поначалу были опасения, что большое число потоков приведет к снижению производительности, однако практика показывает, что это не так. Учитывая тенденции развития современной вычислительной техники и операционных систем (многопроцессорные системы, быстрое переключение контекста потоков, наличие внутреннего параллелизма на уровне процессора), а также самого характера измерительных задач (в измерениях чаще всего надо следить параллельно за многими параметрами) авторы считают, что это стратегически правильный путь.

База данных реального времени. Это набор общедоступных тегов (скалярных переменных) и кривых (динамических массивов), которые хранят состояние прикладных программ и измеряемые данные. Поскольку каждая прикладная программа выполняется в своем потоке и имеет логически изолированное адресное пространство, коммуникации между прикладными программами осуществляются в основном через базу данных, не считая обмена сообщениями. Обобщенно можно представить себе работу измерительных систем под управлением CRW-DAQ как набор параллельно запущенных прикладных программ, которые заполняют и модифицируют базу данных тегов и кривых.

Например, регулирование температуры печи может быть реализовано следующим образом. Драйвер АЦП измеряет сигнал датчика и помещает его в базу данных. Параллельно работающая программа калибровки извлекает сигнал датчика из базы данных и переводит его по калибровке в температуру. Параллельно работающая программа регулирования извлекает температуру из базы данных и формирует сигнал для управления печью. Наконец, драйвер ЦАП берет из базы данных управляющий сигнал и передает его на исполнительное устройство.

Такая структура прикладных программ придает системе большую гибкость. В приведенном примере легко, например, заменить один тип драйвера АЦП на другой, лишь бы его воздействие на базу данных было таким же. Это позволяет также использовать одни и те же прикладные программы во многих измерительных системах, что сильно сокращает время разработки.

Многооконный графический интерфейс реального времени. Позволяет наблюдать и управлять процессом измерений в реальном времени, в процессе измерений. Наблюдаемые данные располагаются в окнах в виде мнемосхем, графиков, текстов, спектров или поверхностей. При этом графический интерфейс удовлетворяет специальным требованиям.

Для измерительной системы важнее всего, чтобы графический интерфейс ни в коем случае не препятствовал параллельно выполняемым измерениям. Для этого сделано разделение потока интерфейса пользователя и потоков управления. Все измерения выполняются в отдельных программных потоках с высоким приоритетом. Обновление изображения и прорисовка идет по таймеру в потоке с нормальным приоритетом. Поток прорисовки, имея относительно низкий приоритет, всегда может быть вытеснен измерительными потоками, то есть не препятствует решению измерительных задач.

Система отображения в реальном времени основана на следящем мониторе. Это значит, что подавляющее число прорисовок система выполняет сама, без участия прикладных программ, которые чаще всего вообще не заботятся о рисовании и даже “не знают” о нем. Прикладные программы работают только с тегами, а монитор сам следит за состоянием отображаемых тегов и ,по мере необходимости и возможности, обновляет изображение на экране. За счет этого прикладные программы становятся существенно проще и понятнее.

Программный интерфейс прикладных программ не содержат функций рисования, которые могут блокировать выполнение потока на длительное время. Все функции рисования реализованы через посылку сообщений без блокировки измерительного потока. Процедуры прорисовки также тщательно проработаны с той целью, чтобы время блокировки общих ресурсов было минимальным. Это уменьшает вероятность приостановки измерительных потоков из-за блокировки общих ресурсов.

Асинхронная система ввода - вывода. Большинство функций ввода - вывода буферизовано с использованием FIFO и выполняется асинхронно. Это значит, что вызов функций ввода-вывода не приводит к блокировке измерительного потока. Как правило, отдельные подсистемы ввода – вывода (консольное окно, COM-порты, звуковые сообщения и т.д.) реализованы в виде серверных потоков, которые принимают команды и выдают результаты через FIFO.

Отказ от использования блокирующих вызовов. Блокирующие вызовы – это вызовы, приостанавливающие выполнение программного потока до наступления какого - то внешнего события. Например, в Windows функции SendMessage, WaitForSingleObject и т.д. относятся к блокирующим вызовам. Эти методы применяются для снижения загрузки процессора при ожидании различных событий. Опасность заключается в том, что заблокированный поток по каким-то причинам может так и не дождаться ожидаемого события, что равносильно его повисанию. Поэтому при создании измерительного пакета предпочтение всегда отдавалось периодической проверке статуса событий без ожидания и без блокировки. Например, для посылки сообщений всегда использовался не блокирующий вызов PostMessage , а для временной блокировки предпочтение всегда отдавалось вызову Sleep, который не может вызвать повисания. Эти меры, может быть, не оптимальны с точки зрения использования процессора, но они повышают надежность работы, что для измерительных систем намного важнее. Блокировки (чаще всего критические секции) использовались в основном для защиты общих ресурсов, при этом тщательно отслеживалось, чтобы время блокировки было минимальным.

Наличие встроенных языков программирования. Пакет CRW-DAQ имеет три интегрированных языка:

- Daq Script – интерпретатор выражений, нечто среднее между интерпретатором C и Basic. Этот простой интерпретатор служит для тех случаев, когда программа не может быть формализована на этапе компиляции. Например, если требуется обеспечить регулирование температуры по заданному пользователем закону, пользователь может ввести закон в виде формулы в поле ввода, а интерпретатор его выполнит.
- Daq Pascal – компилятор упрощенного языка Pascal, генерирующий код некоторой виртуальной машины, плюс интерпретатор кода этой виртуальной машины. Задача этого языка – создание хорошо защищенного кода обычных прикладных программ. Высокая степень защиты получается за счет того, что интерпретатор кода виртуальной машины хорошо защищен, хотя выполняется, конечно, несколько медленнее естественного машинного кода. На практике самая большая часть прикладного кода измерительных систем написана именно на языке Daq Pascal.
- Object Pascal – компилятор объектно-ориентированного языка Pascal в естественный для данной системы машинный код. Для CRW16 это Borland Pascal 7.0, для CRW32 – это Delphi 5.0. Встроенный компилятор позволяет создавать, редактировать, компилировать и тут же выполнять подпрограммы, выполненные в виде DLL-библиотек, которые дают доступ ко всем мощностям и возможностям машины. DLL-программы не конкурируют с программами Daq Pascal, они создаются только для специальных целей, когда средств или скорости Daq Pascal не хватает. Например, драйверы высокоскоростных измерительных устройств или трудные математические алгоритмы можно реализовать в виде DLL.

Каждая из прикладных программ, создаваемых на одном из встроенных языков программирования, выполняется в отдельном потоке и имеет логически изолированное адресное пространство. Для коммуникации между собой прикладные программы могут использовать базу данных реального времени, а также посылку сообщений. При этом, прямо в процессе измерений, каждая из прикладных программ может быть отредактирована и откомпилирована (при этом поток прикладной программы временно приостанавливается). Другие параллельно выполняемые прикладные программы при этом продолжают нормальную работу. Это свойство пакета CRW-DAQ позволяет разрабатывать или модифицировать прикладные программы, не прерывая измерений. Немногие измерительные системы имеют такие возможности.

Практика показала, что наличие встроенной многоуровневой языковой поддержки не является роскошью или излишеством. Это позволяет решать задачи в более короткие сроки, повысить гибкость и в то же время надежность прикладных программ. Для каждой частной задачи выбирается язык, позволяющий решить ее проще и быстрее.

Совмещенная среда разработки и исполнения для измерительных систем. В большинстве коммерческих систем среда разработки отделена от исполнительной системы, так что для модификации управляющей программы надо остановить измерения, перекомпилировать проект и запустить измерения вновь. Для исследовательских установок, когда алгоритмы управления часто “нащупываются” в процессе исследования, прямо во время измерений, это не очень удобно, так как после прекращения измерений восстановить прежние условия бывает трудно или долго. Система CRW-DAQ, за счет высокой степени параллелизма, а также совмещения среды разработки и исполнения позволяет разрабатывать и отлаживать прикладные программы, не прерывая процесса измерений. Например, можно запустить вакуумные измерения и параллельно отлаживать драйвер масс-спектрометра. Поскольку каждая Daq-программа выполняется в своем программном потоке, перекомпиляция драйвера масс-спектрометра напрямую не влияет на работу подсистемы вакуумных измерений. Таким образом, совмещение среды

разработки и исполнения позволяет ускорить процесс разработки и модернизации измерительных систем.

Другой причиной создания совмещенной среды разработки и исполнения явилось стремление сделать пакет “самодостаточным”. Это связано с тем, что многие исследовательские работы ведутся в командировках, в других институтах или на удаленных площадках, иногда на "слабых" машинах. В этих условиях зачастую нет возможности развернуть большую систему разработки, подобную Delphi, Visual C++ или LabView. Пакет CRW-DAQ, при объеме порядка 20 МБ, позволяет полностью реализовать весь цикл разработки измерительных систем, включая создание интерфейса, кода прикладных программ и драйверов, калибровок, справочной системы и т.д. Полное “развертывание” системы CRW-DAQ занимает несколько минут, не требуя инсталляции каких-либо дополнительных пакетов или драйверов.

Развитые средства диагностики и отладки для задач реального времени. К ним следует отнести монитор ресурсов, “сторож”, отладочные файлы, консольные окна. Монитор ресурсов – консольное окно, которое позволяет наблюдать загрузку процессора для каждой из выполняемых прикладных программ, частоту их опроса, использование физической и виртуальной памяти и другую информацию, связанную с использованием вычислительных ресурсов. Сторож – это монитор, который отслеживает активность запущенных программных потоков и выдает предупреждение, если какие-то потоки долго не проявляют “признаков жизни”. Это, фактически, программная диагностика повисания прикладных программ. Отладочные файлы и консольные окна – буферизованные средства асинхронного вывода текстовых сообщений в файл или в окно. Следует сказать, что встроенные в CRW-DAQ языки не имеют отладчика и трассировки. Для задач реального времени трассировка практически не имеет смысла, так как для этих задач важно не только что выполняется, но и когда это происходит. Трассировка же приостанавливает выполнение программы, нарушая при этом временные характеристики процесса. Поэтому наиболее полезным средством отладки в реальном времени является асинхронный вывод отладочных сообщений в файл или консольное окно, так как при этом временные характеристики процесса практически не нарушаются. Например, можно направить поток обмена сообщениями по COM-порту в файл, чтобы затем проанализировать работу подключенных к порту устройств.

Поддержка стандартных интерфейсов связи. Чаще всего сложные исследовательские установки реализуются в виде распределенной сети, содержащей один или несколько управляющих компьютеров, а также удаленных измерительных устройств, соединенных через Ethernet, RS-232, RS-485. Во всех случаях, когда это было технически возможно, предпочтение отдавалось внешним устройствам, подключенным через стандартный интерфейс, например, RS-232. Во-первых, это повышает гибкость систем, позволяет легко наращивать число каналов и устройств. Во-вторых, это связано с быстрым развитием вычислительной техники. В среднем каждые 2-3 года происходит смена поколений компьютеров и операционных систем. Для внешних устройств, подключенных через стандартный интерфейс, эта замена проходит обычно безболезненно, чего не скажешь об устройствах на внутренних шинах (ISA,PCI). Программный интерфейс встроенных языков пакета содержит функции, необходимые для работы с устройствами RS-232, RS-422/485, GPIB, ISA, PCI, LPT, Ethernet. Это позволяет реализовать значительную часть драйверов устройств в виде прикладных программ на встроенных языках, не перегружая ими ядро пакета.

Библиотека встроенных драйверов. Драйверы наиболее часто используемых на практике устройств были включены в ядро пакета. Это, например, устройства серий ADAM-4000, ICP-DAS 7000, 87000, Balzers, ряд измерительных карт Advantech. Значительная часть драйверов устройств в ядро пакета сознательно не включалась. Это связано с тем, что пакет позволяет реализовать драйверы в виде прикладных программ на встроенных языках (Daq Pascal, Object Pascal). В тех случаях, когда эти средства

позволяли решить поставленные задачи, драйверы в пакет не включались, а оставались прикладными программами. За счет этого ядро пакета стало компактнее и надежнее, модифицируется оно довольно редко.

Встроенная система архивирования накопленных данных. Данные, поступающие во время измерений, могут сохраняться для последующей offline обработки. Для сохранения предусмотрено несколько форматов. Чаще всего используется внутренний двоичный формат архивирования, позволяющий сохранять и загружать окна со всем их содержимым (текстом, набором кривых и т.д.). Обмен данными с другими приложениями реализуется обычно через текстовые таблицы. В настоящее время доступ к стандартным базам данных не реализован, так как в сфере научных исследований они редко используются. При необходимости импорта или экспорта данных в другие форматы пишется DLL библиотека на встроенном языке Object Pascal, которая и выполняет требуемые действия.

Встроенные средства обработки и первичного анализа данных. Наличие этих средств позволяет частично или полностью анализировать поступающие в процессе измерений данные в режиме online. Например, в процессе измерений можно сгладить зашумленную кривую, оценить ее производную, вычесть фон и т.д. Большинство функций обработки данных реализуется в виде DLL программ, написанных на встроенном языке Object Pascal. Обработка в основном ориентирована на графические окна: входные данные в виде набора графиков кривых надо поместить в окно, играющее роль аргумента, затем вызвать нужную программу DLL. Результат обработки появляется в новом окне, тоже в виде графиков кривых. Этот стиль работы сильно отличается от таблично-ориентированных пакетов, таких как Excel, Origin и т.д. Авторам представляется, что для online анализа экспериментальных данных оперирование с графиками более естественно, чем использование таблиц. Графики позволяют быстро, одним взглядом, оценить характер зависимости в целом, чего нельзя сказать о таблице. Во время эксперимента это играет важную роль, так как исследователь часто должен принимать оперативные решения на основании анализа поступающих данных.

Встроенная справочная служба. Пакет содержит в себе обширную справочную службу, содержащую сведения, полезные как для программистов, так и для конечных пользователей. Справочная служба содержит также разнообразную информацию, прямо не относящуюся к пакету, но часто используемую на практике, например, спецификации наиболее часто используемого оборудования, таблицы физических констант и т.д.

Заключение

Созданный авторами программный пакет CRW-DAQ позволил успешно автоматизировать целый ряд исследовательских установок. Он содержит в себе большой потенциал для дальнейшего развития, так как имеет интегрированные средства разработки измерительных систем и программ обработки данных, развитый графический интерфейс пользователя.

Кроме перечисленных выше задач программный пакет CRW-DAQ успешно использовался при создании систем охлаждения и термостабилизации спектрометра фотонов PHOS [4, 5], разрабатываемого для эксперимента ALICE в ЦЕРН.

Список литературы

1. Ю.И.Виноградов, А.В.Курякин, А.А.Юхимчук и др. Автоматизированная система контроля, управления и сбора данных стенда "Прометей". Материаловедение, №1, 2002, с.46-50.
2. Ю.И.Виноградов. В.С.Арюткин, В.А.Курякин и др. Автоматизированная система контроля и управления комплексом подготовки газовой смеси для экспериментального

исследования мюонного катализа ядерных реакций синтеза. Препринт РФЯЦ-ВНИИЭФ, 82-2002, Саров, 2002, 26 с.

3. Виноградов Ю.И., Арюткин В.С, Курякин А.В. Система контроля и управления комплекса тритиевой мишени для исследования экзотических нейтронно-избыточных ядер. ВАНиТ, Серия: Физика ядерных реакторов, выпуск ½, 2002, Материалы 51 Международной конференции по ядерной спектроскопии и структуре атомного ядра. с.197-200.
4. Technical Design Report of the Photon Spectrometer (PHOS), CERN/LHCC 99-4, ALICE TDR 2, 5 March 1999.
5. Басманов В.Ф., Блик А.М., Боголюбский М.Ю. и др. Испытания 64-канального прототипа фотонного спектрометра для эксперимента ALICE. ВАНиТ, Серия: Физика ядерных реакторов, выпуск ½, 2002, Материалы 51 Международной конференции по ядерной спектроскопии и структуре атомного ядра, с.204-207.