Р13-2006-41

Н. В. Астахова*, Л. Г. Бордюгов**, А. В. Герасимов***,
Н. Д. Дикусар, Г. И. Еремин***, А. И. Иванов**,
Ю. С. Крюков***, Н. Г. Мазный**, О. В. Рябчун***,
И. М. Саламатин

РАСПРЕДЕЛЕННАЯ БЕСПРОВОДНАЯ СИСТЕМА РЕГИСТРАЦИИ С СИНХРОНИЗАЦИЕЙ ПОТОКОВ ДАННЫХ

Направлено в журнал «Приборы и техника эксперимента»

* ФГУП НИИ «Атолл», Дубна
** ЗАО НПЦ «Аспект», Дубна
*** ФГУП «НИИПА», Дубна

Астахова Н. В. и др. Р13-2006-41
Распределенная беспроводная система регистрации с синхронизацией потоков данных

Развивается способ обеспечения преемственности программного кода при изменении класса задач и способ интегрирования в прикладную систему драйверов специальных устройств. Разработанные принципы и методы построения систем автоматизации использованы при построении беспроводной распределенной системы, предназначенной для регистрации характеристик импульсных процессов с синхронизацией потоков данных, передаваемых по радиоканалу. Точность синхронизации при применении оборудования с частотой дискретизации 20 кГц составила $\pm$50 мкс. Замена части используемого оборудования (частоты дискретизации) позволяет повысить точность до 0,1 мкс. Полученные результаты могут быть использованы при разработке систем мониторинга объектов, систем автоматизации экспериментов, АСУТП.

Работа выполнена в Лаборатории нейтронной физики им. И. М. Франка ОИЯИ.

Препринт Объединенного института ядерных исследований. Дубна, 2006

Astakhova N. V. et al. P13-2006-41
Distributed Wireless Data Acquisition System with Synchronized Data Flow

New methods to provide succession of computer codes under changes of the class of problems and to integrate the drivers of special-purpose devices into application are devised. The worked out scheme and methods for constructing automation systems are used to elaborate a distributed wireless system intended for registration of the characteristics of pulse processes with synchronized data flow, transmitted over a radio channel. The equipment with a sampling frequency of 20 kHz allowed us to achieve a synchronization accuracy of up to $\pm$ 50 μs. Modification of part of the equipment (sampling frequency) permits one to improve the accuracy up to 0.1 μs. The obtained results can be applied to develop systems for monitoring various objects, as well as automation systems for experiments and automated process control systems.

The investigation has been performed at the Frank Laboratory of Neutron Physics, JINR.

Preprint of the Joint Institute for Nuclear Research. Dubna, 2006

ВВЕДЕНИЕ

В системах мониторинга дорогостоящих или потенциально опасных объектов используются подсистемы регистрации сигнальной информации, датчики которых охватывают контролируемый объект. В зависимости от геометрических (географических) характеристик объекта конфигурация системы мониторинга может включать ряд модулей регистрации, в том числе не имеющих электрической связи с центром управления. Такие системы, помимо фиксации штатной информации о поведении объекта, используются для регистрации дополнительной информации с целью обнаружения источников сигналов (например, выбросов загрязнений, грозовых явлений, лавин, взрывов), определения их координат, прогнозирования развития состояния объекта и т. п. Для решения этих задач требуется синхронизация потоков данных от используемых модулей регистрации, средства динамического управления объемом регистрируемой информации в зависимости от состояния объекта и др.

Эффективность практического использования систем зачастую зависит от сроков разработки, наличия средств динамического изменения их конфигурации, возможности установки системы на объекте значительной протяженности (десятки км) и в условиях отсутствия готовых средств коммуникации (например, Интернета или телефонной сети).

Так как разработка программного обеспечения таких систем достаточно трудоемка, то возможность настройки ранее разработанного программного кода в рамках новой системы является весьма актуальной. В данной работе на основе результатов, изложенных в [1–3], *продолжено* развитие методов интегрирования программных модулей в систему с целью обеспечения настройки программного кода на *новый класс задач* с изменяемым составом драйверов. Разработанные принципы и методы использованы при построении беспроводной распределенной системы, предназначенной для регистрации характеристик импульсных процессов с синхронизацией регистрируемых потоков данных.

1. СТРУКТУРА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

В работах [1, 2] сформулирована концепция системы автоматизации экспериментов, в соответствии с которой для *определенного класса задач* программный код системы инвариантен относительно изменений методики экспе-

римента. Функциональные возможности конкретной экспериментальной установки определяются составом оборудования и реализуются драйверами, управляющими исполнительными устройствами. В работе [3] предложен вариант способа интеграции в систему драйверного слоя программ, обеспечивающий преемственность кода системы при изменении состава драйверов. Область применения системы [1, 2] при заданном составе драйверов определяется алгоритмом управляющей программы.

Отметим ограничения, существующие в работах [1–3]:

• изменение состава драйверов возможно только после остановки прикладной системы, не введено использование горячего резерва;

• объем управляющей программы, определяющей состав методик, обслуживаемых программным комплексом, не оптимизирован: включены процедуры (например, модуль формирования команд), не зависящие от методики решения задачи, место которых в составе базовых модулей;

• интерфейс пользователя объединен с кодом реализации алгоритма;

• модули драйверного слоя являются серверами, что усложняет алгоритм программы формирования команд (SETVECTOR в [3]) и процедуру динамического конфигурирования системы.

Эти ограничения устраняются при дальнейшем развитии структуры программного обеспечения (ПО). С этой целью в структуру ПО введены *сервер приложения* и *драйвер приложения*, как это представлено на рис. 1. *Сервер приложения* динамически регистрирует конфигурацию прикладной системы.

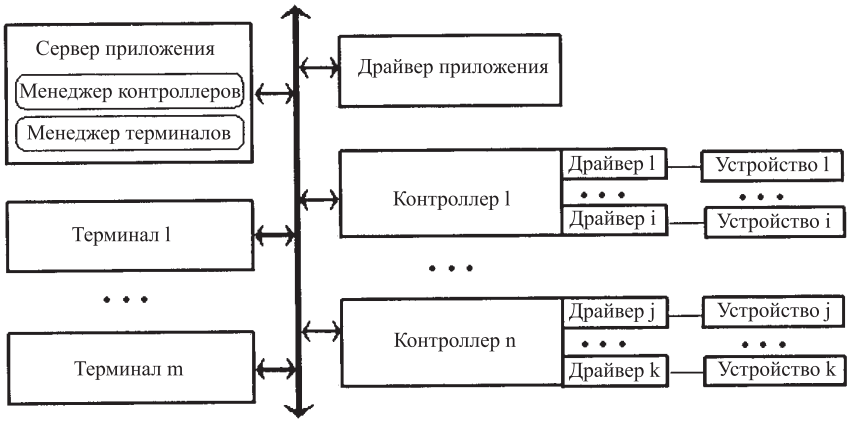

Рис. 1. Структура программного обеспечения системы

Эта централизованная информация используется им для транспортировки команд между компонентами системы. Программа управления [1, 2] заменена

двумя программами — *драйвером приложения*, реализующим алгоритм управления, и программой *терминал*, которая является интерфейсом пользователя. В результате, для изменения класса задач, решаемых таким программно-аппаратным комплексом, достаточно изменить только *драйвер приложения* и (или) соответствующий ему интерфейс. Для упрощения программ динамического формирования прикладной системы модули драйверного слоя оформлены как сетевые клиенты *сервера приложения*.

Данная структура предполагает использование нескольких логических ЭВМ, имеющих следующее назначение:

- управляющая ЭВМ (возможно несколько), на которой работает программа *терминал* (тонкий клиент);
- ЭВМ *сервера приложения*;
- группа *контроллеров*, работающих с оборудованием.

Физические ЭВМ системы, в зависимости от загруженного в них программного обеспечения, могут выполнять функции нескольких логических.

Программный комплекс включает следующие программы, работающие в локальной сети: *терминал, драйвер приложения, сервер приложения* и *контроллер*.

2. ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ЭВМ КОНТРОЛЛЕРА

ЭВМ контроллера предназначена для управления специальным оборудованием, обеспечивающим регистрацию штатной информации о состоянии объекта и системы, а также характеристик различных сигналов в течение заданного интервала времени при возникновении инициирующего события. Программное обеспечение *контроллер* содержит процедуры обслуживания канала связи (сетевой клиент), интерпретатор команд и группу драйверов. При включении ЭВМ контроллера его программа загружается в память, читает файлы прежних значений параметров и с заданным периодом повторяет попытки подключиться к *серверу приложения*. Установив соединение с сервером, шлет ему свое логическое имя с требованием включить его в состав системы и принимает текущие значения параметров. В дальнейшем работает интерпретатор команд, поступающих через *сервер приложения*. Интерпретатор исполняет служебные команды:

- принять и запомнить новые значения параметров;
- прервать исполнение текущей команды;
- перезагрузить операционную систему;
- выполнить прикладную команду.

Прикладные команды выполняются драйверами. В данной работе для краткости мы рассмотрим работу только драйвера АЦП, определяющего основ-

ные функциональные возможности системы. Этот драйвер выполняет команды:

• включить программу обнаружения стартового события;

• регистрировать поток данных в течение заданного времени;

• передать результаты регистрации на *сервер приложения*.

В случае разрыва соединения с сервером программа останавливает исполнение текущей команды и возвращается к попыткам установить соединение. В процессе эксплуатации системы не предполагается работа пользователя с программой *контроллер* непосредственно. Необходимые тесты могут быть выполнены по запросу программы *терминал*.

Наиболее важными в программном обеспечении контроллера являются алгоритмы драйвера АЦП, обеспечивающие обнаружение события, регистрацию и синхронизацию данных. Рассмотрим их более подробно.

2.1. Программа регистрации данных. Подсистема регистрации работает в следующих режимах:

• ожидание команды;

• автоматическое обнаружение стартового события, инициирующего начало регистрации потока данных;

• регистрация данных.

Основное состояние драйвера АЦП — режим ожидания. В режиме ожидания данные от датчиков последовательно поступают в два кольцевых буфера — встроенный в АЦП (например, 1024 байта для PCI-1711L [4]) и программный буфер, длина которого параметризуется (см. рис. 2). По заполнении любой половины программного буфера вырабатывается сигнал, который

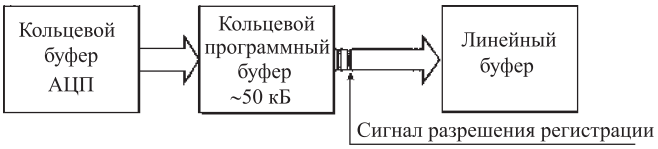

Рис. 2. Схема работы подсистемы регистрации данных

используется программой регистрации данных, а также программой обнаружения события. В режиме ожидания данные от датчиков сохраняются только в кольцевых буферах. После включения ЭВМ контроллера через время

$$t = 2L/(nf),$$

где t — время (в секундах); L — длина программного буфера (в байтах); n — число используемых входов АЦП; f — частота дискретизации сигналов (Гц), система располагает описанием предыстории ожидаемого события, максимальный объем которой определяется длиной программного буфера L. В реальных условиях $t = \sim$1с.

Предыстория используется для определения фоновых условий, предшествовавших событию (для программ математической обработки), а также для ретроспективного формирования начального участка синхронизированного потока данных от датчиков.

Изменение режима работы драйвера происходит по команде. Команда может быть подана оператором или выработана самой системой при автоматическом способе обнаружения события. При получении команды заказанное действие выполняется одновременно с приемом данных в кольцевые буфера.

По команде «*обнаружить событие*» включается программа обнаружения стартового события. Если событие обнаружено, то последует команда всем контроллерам регистрировать сигналы датчиков. Если в течение заданного интервала времени событие не обнаружено, данная команда будет повторена.

При получении команды «*регистрировать сигналы датчиков*» в каждом контроллере начинается процесс переноса данных из кольцевого программного буфера в линейный буфер (поток данных в памяти), длина которого зависит от заданной продолжительности регистрации (см. рис. 2). Одновременно с этой командой *драйвер приложения* передает контроллеру данные для определения локального времени возникновения события, вычисленные программой обнаружения стартового события. По заполнении линейного буфера программа передает *драйверу приложения* сигнал готовности данных и переходит в режим ожидания команды.

По команде «*передать данные*» каждый контроллер выполняет следующее:

• в потоке зарегистрированных данных вычисляет позицию, соответствующую времени события;

• начиная с времени возникновения события выделяет из зарегистрированного потока участок заказанной длины;

• передает на ЭВМ *сервера приложения* зарегистрированные данные, начало которых соответствует времени события, и протокол работы.

2.2. Программа обнаружения стартового события. Для автоматического обнаружения события контроллер оснащается специальным оборудованием (например, оптическим датчиком, датчиком гамма-излучения и т. д.), сигнал которого подается на один из входов АЦП — служебный вход. Контроллер, обнаруживший событие, считается «активным контроллером», остальные — пассивными. Номер служебного входа известен программе обнаружения события. При получении команды «*обнаружить событие*» программа начинает просматривать кольцевой программный буфер. Событие считается обнаруженным, если в потоке данных встретится заданное количество идущих подряд значений, превышающих указанный порог.

Все контроллеры оборудованы датчиками GPS [5]. Сигнал pps (импульс в секунду) этого датчика подключается к нулевому входу АЦП и используется программой синхронизации потоков данных.

В случае обнаружения события эта процедура вычисляет дистанцию от момента возникновения события до появления очередного сигнала GPS и через *сервер приложения* передает эту информацию *драйверу приложения*, который немедленно транслирует ее всем контроллерам с командой начала регистрации.

При работе в режиме включения регистрации по команде оператора получение команды оператора считается стартовым событием, остальная часть сценария работы не меняется.

2.3. Алгоритм синхронизации потоков данных. С момента включения контроллера должно пройти некоторое время (∼30 с) для того, чтобы датчик GPS настроился на видимые спутники и начал принимать сигналы синхронизации. Программе синхронизации требуется зарегистрировать хотя бы один импульс GPS для настройки процедуры, вычисляющей положение сигналов GPS в кольцевом программном буфере, после чего система готова к работе.

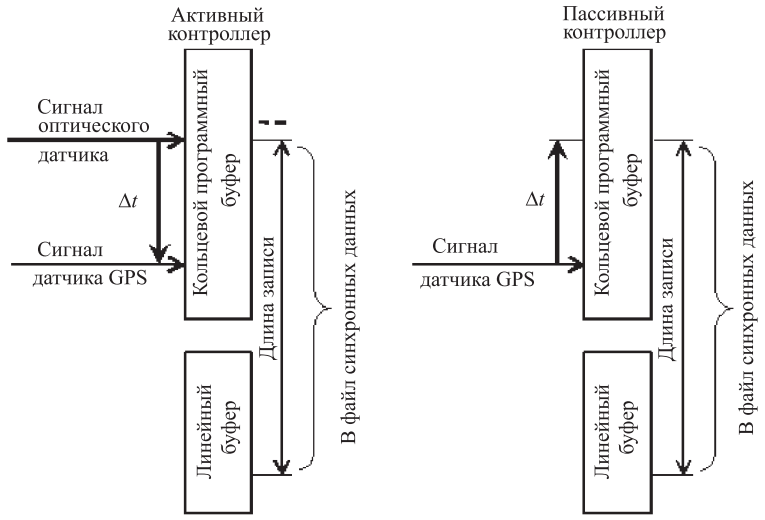

Рис. 3. Схема способа синхронизации данных

Как отмечено выше, хотя бы к одному из контроллеров системы должен быть подключен датчик, позволяющий идентифицировать факт возникновения события. При получении команды ожидания события драйвер начинает просматривать поток данных (кольцевой программный буфер) в поисках признака наличия события. При обнаружении события программа по фронтам импульсов сигналов датчика и сигнала GPS определяет дистанцию до последующего сигнала синхронизации времени (разность индексов соответствующих дискретных отсчетов в потоке данных) и транслирует это значение всем контроллерам транзитной командой через *сервер приложения*. Каждый контрол-

лер благодаря наличию зарегистрированной предыстории состояния объекта отсчитывает от соответствующего синхроимпульса нужное число отсчетов из предыстории и формирует поток данных, начинающийся с времени возникновения события, как это показано на рис. 3. По желанию в начале этого потока может быть помещена дополнительная часть, содержащая информацию для вычисления фоновых условий, предшествовавших событию.

3. ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ЭВМ СЕРВЕРА ПРИЛОЖЕНИЯ

Программное обеспечение ЭВМ *сервера приложения* содержит процедуры обслуживания канала связи, интерпретатор команд и *драйвер приложения*. При включении ЭВМ сервера программа загружается в память, читает файлы прежних значений параметров и слушает порт сервера. При поступлении запроса на соединение регистрирует логическое имя клиента, сообщает об этом на *терминалы* и передает клиенту текущие значения параметров. В дальнейшем работает интерпретатор команд, поступающих от *терминала* (команды пользователя) с учетом состояния функциональных устройств, о чем информация поступает от контроллеров. Программа *сервера приложения* является коммуникационным модулем исполняющей системы, не зависящим от методики решения прикладной задачи. Для обеспечения такой независимости функциональная часть прикладной задачи вынесена на драйверный уровень. Помимо драйверов подсистемы регистрации разработан *драйвер приложения*, осуществляющий интерпретацию команд *терминала* и их выполнение. Интерпретатор сервера исполняет служебные команды с *терминала*:

- принять и запомнить новые значения параметров; передать новые значения параметров всем клиентам (контроллерам и дополнительным *терминалам*);
- всем контроллерам прервать исполнение текущей команды;
- прервать исполнение текущей команды на сервере;
- всем контроллерам перезагрузить операционную систему;
- перезагрузить операционную систему на ретрансляторе;
- выполнить прикладную команду (команда *драйверу приложения*).

Драйвер приложения выполняет заложенную в него программу, используя следующие команды:

- включить программу обнаружения стартового события;
- всем контроллерам регистрировать поток данных;
- всем контроллерам передать результаты регистрации.

Помимо этого введены некоторые тестовые команды. После выполнения прикладных команд *драйвер приложения* шлет подтверждение на *терминал*.

В случае разрыва соединения с клиентом (контроллером или *терминалом*) программа исключает его из состава системы, информирует об этом работающие *терминалы* и продолжает слушать порт сервера.

В процессе эксплуатации системы не предполагается работа пользователя с этой программой непосредственно. Необходимые тесты могут быть выполнены по запросу программы *терминал*.

4. ПРОГРАММА ТЕРМИНАЛ

Данная программа является оконным интерфейсом пользователя для управления работой системы (см. рис. 4). Интерфейс программы *терминал* узкоспециализирован на передачу исполняющей системе определенного состава команд. Пользователю предоставлены следующие возможности:

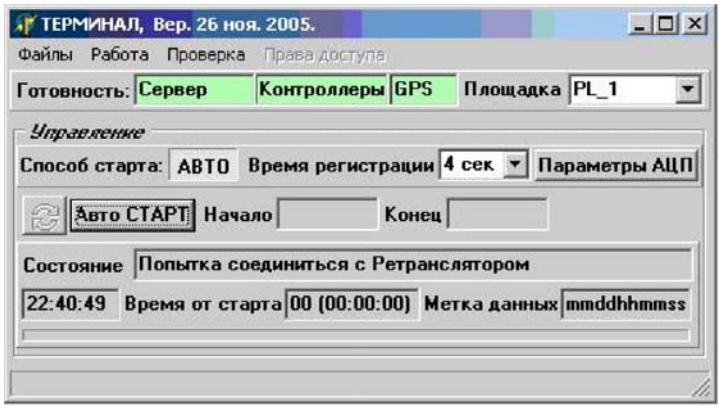

Рис. 4. Окно программы *терминал* — интерфейс пользователя

• изменить значения ряда параметров, определяющих условия работы системы; в числе таких параметров — способ начала регистрации (автоматический при обнаружении стартового события или по команде оператора), время ожидания стартового события, продолжительность регистрации и др.;

• включить или остановить работу в условиях автоматического обнаружения события;

• включить или остановить регистрацию данных;

• выполнить ряд служебных операций — перезагрузить ОС на ЭВМ сервера, контроллера, выполнить тесты и др.

Программа *терминал* содержит процедуры обслуживания канала связи, чтения результатов измерений и ряд диалоговых процедур. При включении ЭВМ программа загружается в память, читает файл предыдущих значений параметров и с заданным периодом повторяет попытки подключиться к *серверу приложения*. Установив канал связи с сервером, шлет ему свое логическое имя с требованием включить в состав системы и принимает текущие значения

параметров. В дальнейшем работают диалоговые процедуры, вызываемые из меню или при нажатии соответствующих клавиш. Эти процедуры формируют команды, которые передаются на сервер и исполняются там *драйвером приложения.*

5. ОБЕСПЕЧЕНИЕ НАДЕЖНОСТИ

С целью повышения надежности работы комплекса и удобства его эксплуатации

• обеспечена автоматическая регистрация и взаимное сетевое соединение компонентов системы при любой последовательности их включения. Работоспособность комплекса не теряется при временном отключении части контроллеров, при восстановлении состава контроллеров взаимодействие с ними восстанавливается автоматически;

• введены операции дистанционной перезагрузки операционной системы на сервере и контроллере с автоматическим восстановлением работоспособности;

• обеспечено централизованное изменение значений и автоматическая поддержка использования последних заданных пользователем значений изменяемых параметров всеми компонентами комплекса независимо от последовательности их включения и возможных операций перезагрузки;

• на ЭВМ *терминала* ведутся индивидуальные протоколы работы всех компонентов системы, введена индикация готовности к работе сервера и контроллеров, а также быстрый доступ к информации о временно отсутствующих компонентах;

• приняты меры к защите данных от потери путем автоматического присвоения названий файлов.

6. КОНКРЕТНОЕ ПРИЛОЖЕНИЕ — АКУСТИЧЕСКАЯ СИСТЕМА

6.1. Схема системы в общем виде показана на рис. 5. Локальная сеть построена на основе радиомодемов фирмы «Alvarion» [6]. Это оборудование позволяет устойчиво работать при расстояниях между компонентами сети до 25 км.

Для **стационарного *терминала*** нами использовалась серийная персональная ЭВМ, а для переносного — мобильный компьютер с ОС Windows.

Сервер приложения собран из модулей фирмы «Advantech» [4], объем памяти 256 Мбайт, объем диска (flash) 1 Гбайт. Для связи с *терминалом* и группой контроллеров использованы отдельные радиоканалы.

Рис. 5. Схема системы

Контроллер также собран из модулей фирмы «Advantech». В отличие от ЭВМ *сервера приложения*, в его конфигурацию включены АЦП и датчик GPS, объем диска уменьшен до 256 Мбайт. Нулевой вход АЦП использовался для синхронизации данных. К нему подключался датчик сигналов GPS [5], обеспечивающий аппаратную поддержку алгоритма синхронизации потоков данных, хотя могут использоваться иные источники сигналов синхронизации. К остальным 15 входам могут подключаться датчики, регистрирующие характеристики исследуемых процессов. Состав и сочетание датчиков определяется конкретной решаемой задачей. К входу АЦП одного из контроллеров должен быть подключен датчик, сигналы которого позволяют идентифицировать факт возникновения тревожного события, например, датчик светового, гамма-, нейтронного излучения и т. п. Количество используемых контроллеров и состав регистрируемой информации определяется характером решаемой задачи и условиями применения.

В случае доступности локальной сети с кабельным соединением (а также на этапе проверки) система может быть использована без изменения программ.

6.2. Описанная система применяется для исследования распространения акустических сигналов.

Для испытания была построена система, включающая два контроллера. Драйверам системы регистрации были заданы следующие параметры: количе-

ство датчиков — четыре; частота дискретизации сигналов АЦП — 20000 Гц; продолжительность регистрации — 4 с; сигнал синхронизации — от датчика GPS; датчик обнаружения события — оптический; остальные датчики — акустические. Акустические датчики обоих контроллеров располагались на одинаковом расстоянии от источника сигналов. В качестве имитатора источника акустических и оптических сигналов использовался заряд КВВ массой 3 г в тротиловом эквиваленте.

Графики сигналов GPS, зарегистрированные двумя контроллерами системы, наложены друг на друга и представлены на рис. 6. Из рисунка видно совпадение фронтов сигналов GPS. На рис. 7 показаны сигналы акустических датчиков двух разных контроллеров.

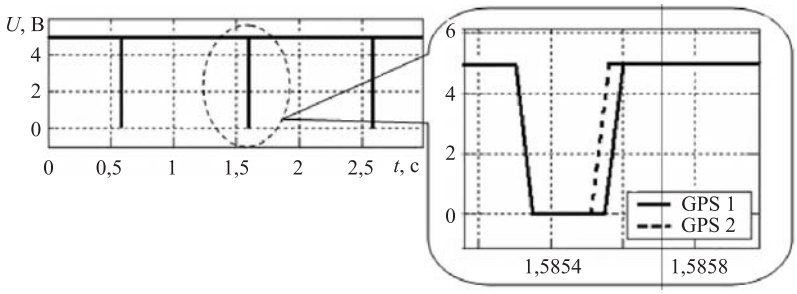

Рис. 6. Совпадение сигналов датчиков GPS двух контроллеров

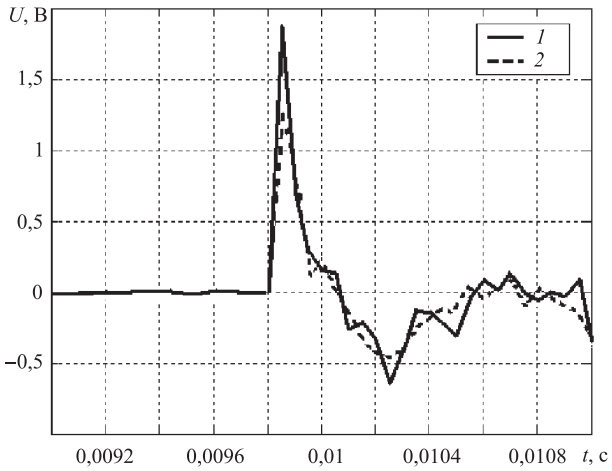

Рис. 7. Сигналы акустических датчиков двух контроллеров: *1* — контроллер 1; *2* — контроллер 2

При дистанции между акустическими датчиками и источником сигналов порядка 100–500 м данная точность синхронизации позволяет определить положение источника сигнала с погрешностью $\sigma \leqslant 1$ м.

Минимальная возможная погрешность синхронизации определяется характеристиками сигнала датчика GPS и составляет 0,1 мкс. Это значение огрубляется используемым АЦП и равно $1/f$, где f — частота дискретизации событий.

7. РЕЗУЛЬТАТЫ

В отличие от [1–3], в данной работе выделен универсальный *сервер приложения*, к которому через каналы клиентов подключается драйверный слой программ. Драйверный (функциональный) слой программ подключается универсальной клиентской программой. Структура драйверов подобна описанной в [3]. Адаптация системы для конкретного приложения (или группы задач) обеспечивается специально разработанным интерфейсом пользователя (клиент–*терминал*) в сочетании с соответствующим *драйвером приложения*. Опыт разработки и проведенные в 2005 г. эксперименты показывают, что использованная схема интеграции компонентов

- обеспечивает возможность изменения области приложения путем замены только двух компонентов — оконного интерфейса пользователя и соответствующего ему драйвера приложения;
- существенно упрощает программирование распределенной системы и ее масштабирование;
- облегчает реализацию динамического изменения конфигурации и состава регистрируемых данных;
- повышает надежность системы в целом за счет повторного использования ранее разработанных компонентов без изменения.

Помимо задач мониторинга компоненты данной системы могут быть использованы при разработке систем автоматизации экспериментов и АСУТП. Для систем контроля распространения выбросов радиационных, химических загрязнений и ряда других точность синхронизации, обеспечиваемая датчиками GPS, не требуется и может быть использовано более дешевое оборудование.

Работа выполнена при поддержке РФФИ (грант № 04-07-90256).

ЛИТЕРАТУРА

1. *Астахова Н. В., Саламатин И. М., Швецов В. Н.* Программный комплекс АС (автоматизация спектрометрии). 1. Концепция программной системы, инвариантной по отношению к изменениям методики эксперимента // ПТЭ. 2004. № 5. С. 56–61.

2. *Астахова Н. В. и др.* Программный комплекс АС (автоматизация спектрометрии). 2. Интерфейс пользователя системы автоматизации эксперимента // Там же. С. 62–68.

3. *Астахова Н. В. и др.* Программный комплекс АС (автоматизация спектрометрии). 3. Методика управления окружением образца и применение ее в программе интерактивного управления спектрометром // ПТЭ. 2005. № 5. С. 36–43.

4. http://www.advantech.com

5. http://www.gps.ru

6. http://www.Alvarion.com

Получено 30 марта 2006 г.

Редактор *Е. В. Сабаева*

Подписано в печать 30.06.2006.
Формат 60 × 90/16. Бумага офсетная. Печать офсетная.
Усл. печ. л. 0,94. Уч.-изд. л. 1,14. Тираж 325 экз. Заказ № 55396.

Издательский отдел Объединенного института ядерных исследований
141980, г. Дубна, Московская обл., ул. Жолио-Кюри, 6.
E-mail: publish@jinr.ru
www.jinr.ru/publish/