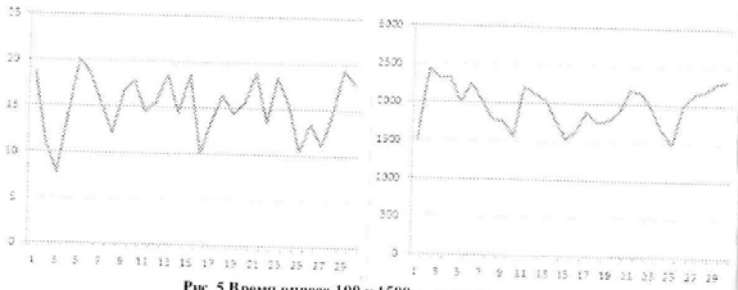

**Рис. 5 Время опроса 100 и 1500 элементов**

Как видно из графиков, если для 100 элементов среднее время опроса составляло 15,26 мс, то для 1500 элементов это значение было уже 1971,5мс. Понятно, что если запрашивать каждый элемент по отдельности, то получается слишком затратные нагрузки на систему.

При анализе передаваемых данных, было выявлено, что можно ускорить обновление за счет оптимизация передаваемых данных и сокращение информации в пакете, а так же запросе только тех объектов, которые нуждаются в обновлении.

При начале цикла обновления, в модуль SoftPlc приходят Id и ParentId (необходим при идентификации объекта при многовложенности) объекта, по которым формируются список объектов обновления. Таким образом если, допустим, всего объектов в управляющей программе 2500, то для текущего обновления может потребоваться не более 50-150. Такой эффект достигается так же за счет запроса элементов, которые находятся в области окна FBEditor. При перемещении окна области видимости элементов, или открытии вложенного блока происходит переформирование пакета обновляемых объектов, что позволяет получать корректную информацию при различных манипуляциях в редакторе.

Размер пакета одного объекта был сокращен с 450 до 75 байт, за счет того, что осталась информация о Id объектов, их типах, количествах контактов и их значениях, вся другая информация, такая как описание объектов, контактов и т.д. была убрана.

Получается, что в один пакет при обмене по TCP/IP может поместиться в среднем около 18 объектов, при этом реальные значения (за счет мало количества контактов элементов, их вложенности и прочего) составляют порядка 30-35 объектов за один запрос. При превышении в какой-то момент заполняемого буфера для отправки, последний заполняемый объект из временного буфера не попадает в основной буфер отправки данных, что позволят избежать различных ошибок при фрагментации пакетов. Не попавший в предыдущую отправку объект, будет отправлен в следующей. Обмен происходит до тех пор, пока не будет получена команда на остановку или не завершиться выполнение всей программы.

При масштабировании рабочей области в редакторе управляющих программ FBEditor, количество видимых элементов в среднем составляет порядка 300 элементов (при большем скроллинге объекты уже плохо различимы и в принципе их обновления не требуется).

Таким образом получается, что при всех максимальных условиях, опрос всех элементов займет порядка 50мс (среднее же время при данной реализации составляет порядка 30-35 мс), что является более чем достаточным условием, и возможно даже увеличение этого времени, для того, чтобы избежать чрезмерной нагрузки сети. Засчет же алгоритма поиска и последующего формирования отдельного списка элементов для обмена происходит опрос не всех объектов в модулеSoftPlc, а только необходимых в текущий момент времени.



**Библиографический список:**

1.Сосонкин В.Л, Мартинов Г.М. Системы числового программного управления: Учеб. Пособие. – М.: Логос, 2005.-296с.

2.Мартинов Г.М., Любимов А.Б., Бондаренко А.И., Сорокоумов А.Е., Ковалев И.А. Подход к построению мультипротокольной системы ЧПУ // Автоматизация в промышленности. 2012. №5. с.8-11.

3.Ковалёв И.А. Разработка библиотеки логических элементов для программно-реализованного контроллера электроавтоматики.Материалы студенческой научно-практической конференции "Автоматизация и информационные технологии (АИТ-2012)". Первый тур. Сборник докладов. - М.: ФГБОУ ВПО МГТУ "Станкин", 2012. - С.19-22.

## СПЕЦИФИКА ПРОВЕДЕНИЯ ТЕСТИРОВАНИЯ ИСПЫТАТЕЛЬНЫХ СТЕНДОВ МНОГОТЕРМИНАЛЬНЫХ СИСТЕМ ЧПУ

***Комаров А.В.***

***Научный руководитель: к.т.н., доц. Мартинова Л.И.***

***Кафедра: «Компьютерные системы управления» ФГБОУ ВПО МГТУ «СТАНКИН»***

Производители станков с ЧПУ оснащают их большим количеством современных узлов и элементов, от корректного и надежного функционирования которых в значительной степени зависит точность исполнения заданной программы и, соответственно, качество изготовляемого изделия. Архитектура автоматизированной распределенной компьютерной системы, предназначенной для контроля и организации входящих в нее программно-аппаратных средств, имеет специфические особенности, на которых основаны способы организации в ней методов управления информацией. Широкий спектр задач, решаемых в рамках дистрибутивной системы, не может быть полностью охвачен одним программным продуктом, в связи с чем, система должна быть гетерогенной [1]. При этом структура системы управления должна соответствовать структуре самого объекта автоматизации [2]. Необходимо, также, создавать открытые системы управления, способные интегрироваться как в уже существующие, так и создающиеся, в том числе и специфические или независимые приложения заказчика в единое решение [3].

Открытость архитектуры подразумевает то, что разработчик использует такой способ построения, который регламентирует и стандартизирует описание принципа действия системы и ее конфигурации. Это позволяет собирать ее из отдельных узлов и элементов, построенных независимыми фирмами-производителями, что дает возможность строить, улучшать и расширять системы наиболее экономичным способом. Протоколы передачи данных между такими системами и аппаратные реализации, базирующиеся на общепринятых стандартах с опубликованными спецификациями, и сами системы создаются на основе разных технических средств, а работают они на различных платформах [4].

Открытая модульная архитектуры автоматизированной системы управления позволяет адаптировать ее для различных типов технологического оборудования и различных технологических задач, расширять ее функциональные возможности за счет простой интеграции новых программно-аппаратных решений. Опыт практического применения многофункциональной системы ЧПУ"АксиОМА Контрол" для станков разного типа подтвердил реальную возможность конфигурирования систем под конкретные технологические задачи. Применение конфигурируемой многофункциональной системы ЧПУ для оснащения станков на предприятиях позводит значительно сократить средства на наладку технологических комплексов,

подключение к цеховым сетям, обслуживание, оборудования, а также на обучение персонала[5].

Важным этапом разработки и создания испытательного стенда является планирование архитектуры двухкомпьютерной системы. Двухкомпьютерная модель предполагает размещение PC-подсистемы на одном компьютере, а NC-подсистемы на другом [6]. В PC-подсистеме целесообразно использовать операционную систему Windowsc расширением реального времени, а вNC-подсистеме – LinuxReal-Time. Двухкомпьютерная архитектура позволяет ускорить процесс обработки данных и, тем самым, оптимизировать проведение испытаний. На рисунке 1 представлена архитектура многотерминального испытательного стенда, разработанная инженерами кафедры «КСУ». Такой архитектурный вариант дает общее представление о принципах открытой распределенной системы применительно к ЧПУ [7].


**Рис. 1. Архитектура многотерминального испытательного стенда**

Архитектура тестируемых многотерминальных систем может различаться в зависимости от задач автоматизации, но испытательный стенд должен позволять проводить оценку системы по определенному набору параметров. Отладка и тестирование проводятся в лабораторных условиях в целях выявления проблем в работе терминалов, станочных панелей, контроллеров, и их взаимодействий между собой, а также в механизмах передачи данных. Результаты таких испытаний заносятся в протокол тестирования и размещаются в системах отслеживания ошибок («баг-трекеры»), основной задачей которых является сбор данных онайденных «багах» в программном обеспечении и отслеживание историй их возникновений вплоть до полного устранения ошибки.

Обобщенная схема проведения испытаний программного продукта представлена на рисунке 2. Ключевым звеном в ней является руководитель группы тестирования, отвечающий за весь цикл разработки программного обеспечения. Основные обязанности руководителя включают в себя: организация процесса тестирования, наблюдение за проведением тестирования, поддержка связи между группами разработчиков и тестировщиков, а также своевременная загрузка исходного кода в репозиторий.

В ходе работы группа тестирования должна фиксировать поведение программного обеспечения, проводить анализ и оценку системы, а также записывать поведение программного обеспечения при заданных входных данных и настройках. После завершения работы, тестировщик записывает найденные ошибки, с подробным



описание их воспроизведения, в «баг-трекер». После чего, группа разработчиков может приступать к устранению неполадок в программном коде.

Помимо программных «багов», присутствуют также и аппаратные ошибки. Так, в ходе тестирования, проверяется не только программная часть системы ЧПУ, но сама аппаратура. В ходе исследования на предмет тестирования многотерминального стенда, такими проблемами стали:

– Работоспособность функциональных и машинных клавиш;

– Работоспособность маховиков, переключателей, кнопок и элементов станочной панели;

– Работоспособность пульта ручного управления;

– Работоспособность приводов и двигателей;

– Работоспособность слотов расширения бас-каплер, плат расширения МОХА;

– Отсутствие коллизий и сбоев при передаче и обработки информации по шине данных.


**Рис. 2. Концептуальная схема разработки ПО**

Все практические аспекты построения конфигурации испытательных стендов, начиная от электрических схем соединения и заканчивая многоуровневыми программными алгоритмами, проверяются в лабораторных условиях, в прямом взаимодействии с разработчиками программного обеспечения. Такой подход к созданию испытательных стендов перспективен для дальнейшего роста, в частности, применения нового оборудования, расширения конфигурации и т.д.

**Библиографический список:**

1 Григорьев С.Н., Мартинов Г.М. Перспективы развития распределенных гетерогенных систем ЧПУ децентрализованными производствами // Автоматизация в промышленности. 2010. №5. С. 4-8.

2 Денисенко В.В. Компьютерное управление технологическим процессом, экспериментом, оборудованием. – М: Горячая линия – Телеком, 2009. – 608 с., ил.

3 Григорьев С.Н., Андреев А.Г., Мартинов Г.М. Перспективы развития кроссплатформенных компьютерных систем числового программного управления высокотехнологичного оборудования // Автоматизация в промышленности. 2011. №5.

4 Мартинов Г.М., Мартинова Л.И., Григорьев А.С. Специфика разработки программного обеспечения для систем управления технологическим оборудованием в реальном времени // Спецвыпуск I-Comm, июль 2009. С. 121-124.

5.Мартинова Л.И., Козак Н.В., Нежметдинов Р.А., Пушков Р.Л., Обухов А.И. Практические аспекты применения отечественной многофункциональной системы ЧПУ ""АксиОМА Контрол" // Автоматизация в промышленности. 2012. №5. с.36-40.

6.Сосонкин В.Л., Мартинов Г.М. Системы числового программного управления: Учеб. Пособие. – М.: Логос, 2005 – 296 стр., ил.

7.Мартинова Л.И., Мартинов Г.М. Организация межмодульного взаимодействия в распределенных системах ЧПУ. Модели и алгоритмы реализации // Мехатроника, автоматизация, управление. - 2010. - N 11 (116). - С. 50-55.

## РАЗРАБОТКА ИНСТРУМЕНТА ПО СОЗДАНИЮ И ОТЛАДКЕ УПРАВЛЯЮЩИХ ПРОГРАММ ДЛЯ ПРОГРАММНО-РЕАЛИЗОВАННОГО КОНТРОЛЛЕРА, ИНТЕГРИРОВАННОГО В СИСТЕМУ ЧПУ

***Никишечкин П.А.***

***Научный руководитель: д.т.н., проф. Мартинов Г.М.***

***Кафедра «Компьютерные системы управления» ФГБОУ ВПО МГТУ «СТАНКИН»***

Программируемые логические контроллеры (ПЛК) на сегодняшний день являются базовыми элементами систем промышленной автоматики. На их основе построены все АСУ ТП, системы мониторинга, контроля функционирования, телеметрии, обеспечения безопасности и многие другие.

В последнее время, новым направлением в области автоматизации стало развитие программно-реализованного логического контроллера (SoftPLC). Применение SoftPLC подразумевает замену классического аппаратного ПЛК на исключительно программно-математическое обеспечение. Такое решение обуславливается тем, что мощность и ресурсы вычислительного ядра персональных и промышленных компьютеров не имеют существенных ограничений и позволяют решать сложные вычислительные задачи. Данный подход позволяет снизить стоимость системы в целом, добиться возможности простой модернизации контроллера в кратчайшие сроки, реализовать поддержку различных протоколов связи на уровне модулей ввода/вывода, а также обеспечить кросплатформенность работы контроллера. [1]

Огромное значение ПЛК занимают в системах ЧПУ. На сегодняшний день, на кафедре «Компьютерные Системы Управления» производится разработка программно-реализованного контроллера, работающего в одной операционной среде с программным обеспечением ЧПУ. Такое решение позволяет сделать систему ЧПУ более независимой от аппаратного обеспечения, снизить стоимость системы и повысить ее надежность. [2]

На сегодняшний день, на рынке существует большое разнообразие систем программирования контроллеров, такие, как CoDeSys, Simatic STEP7, LabView, ISaGRAF, и др. Однако, вышеперечисленные продукты являются объемными и сложными средами, требующими высокой специализации сотрудников, а также наличие платной лицензии на их использование. Таким образом, просматривается необходимость в создании универсального средства по созданию, визуализации и отладки управляющих программ для SoftPLC. Можно выделить следующие основные требования для разрабатываемого инструментария: отсутствие платной лицензии, универсальность и простота работы, стандартизированный язык программирования, наличие режима эмуляции, и возможность использования в учебных целях. [3]

При разработке инструмента за прототип был взят вышеупомянутый CoDeSys, как один из наиболее известных универсальных инструментов МЭК программирования для ПЛК и промышленных компьютеров. Разрабатываемый продукт проектируется с использованием технологий .NET, XML и представляет собой приложение, позволяющее осуществлять проектирование программ, их



взаимодействие с ядром системы управления, а также верификацию и отладку для программно-реализованного контроллера, функционирующего в ядре системы ЧПУ.

В качестве основного инструмента создания управляющих программ был выбран графический язык функциональных блоков FBD (Functional Block Diagram). Выбранный язык является распространенным и входит в состав международного стандарта МЭК 611-31. Программа на языке FBD внешне напоминает функциональную схему логического устройства – совокупность элементов (блоков), входы и выходы которых соединены линиями связи. Среди основных преимуществ FBD можно отметить его визуальную наглядность, поскольку данный язык является полностью графическим.

Основной интерфейс прикладной среды разработки управляющих программ (рис. 1) позволяет производить редактирование управляющей программы, производить настройку параметров функциональных блоков при помощи панели настроек, а также производить отладку управляющей программы. [4]

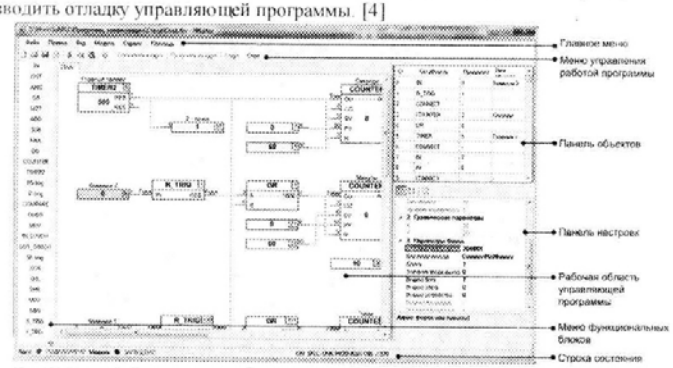


**Рис. 2. Интерфейс программной среды разработки УП для программно-реализованного контроллера, интегрированного в систему ЧПУ**

Интерфейс редактора включает в себя следующие основные компоненты для управления программой и ее работой: главное меню, позволяющее управлять документированием разработанной программы (сохранением/загрузкой), ее работой, а также настройками редактора; меню управления работой программы, – позволяет производить запуск/останов управляющей программы, а также ее отправку и получение из ядра системы управления; меню функциональных блоков, – содержит визуальные компоненты, соответствующие различным функциональным блокам, которыми можно оперировать при создании управляющей программы; рабочая область управляющей программы, – содержит визуальное представление управляющей программы; панель объектов, – содержит список всех функциональных блоков и связей, содержащихся в управляющей программе; панель настроек объектов, – служит для конфигурирования функциональных блоков, а также объектов входов/выходов; строка состояния, – отображает текущее состояние подключения к ядру системы управления, а также статус работы управляющей программы.

При создании управляющих программ в разработанном редакторе, пользователь может оперировать тремя видами функциональных блоков:

- блоки входов/выходов – базовые объекты, содержащие основную информацию о состоянии объекта управления. Как правило, каждый такой объект привязан к входам/выходов аппаратных модулей, либо используется для определения каких-либо константных значений;
- стандартные функциональные блоки – объекты математических, логических операций, таймеры, счетчики, триггеры, а также блоки для операций с байтами. Вся