УДК 621.382:004.358

Н.В. Голян, Ю.П. Шабанов-Кушнаренко

Харьковский национальный университет радиоэлектроники

ПРОГРАММНЫЕ СРЕДСТВА АВТОМАТИЗИРОВАННОГО ПРОЕКТИРОВАНИЯ ПРОПОРЦИОНАЛЬНО-ИНТЕГРАЛЬНО-ДИФФЕРЕНЦИАЛЬНЫХ РЕГУЛЯТОРОВ

Разработан программный продукт, позволяющий автоматически формировать: коэффициенты пропорционально-интегрально-дифференциальных (ПИД) регуляторов и журнал-ведомость работы установки на каждом этапе оптимизации.

Ключевые слова: *пропорционально-интегрально-дифференциальные регуляторы, автоматизация рабочего места, база данных.*

Введение

Использование моделирования для построения имитаторов электронных устройств является сегодня все более актуальным в связи с все большей потребностью автоматизации производства.

В связи с этим системы автоматизированного проектирования используют при прогнозировании поведения моделируемых объектов. Например, характеристики современной лазерной установки могут быть определены и предсказаны средствами моделирования. На моделях выполняют контролируемые эксперименты в тех случаях, когда экспериментирование на реальных объектах практически невозможно из-за отсутствия последних или возникающей во время экспериментов опасности (сети энергоснабжения, химические производства) [1].

Разработка автоматических систем регулирования технологических процессов состоит из ряда стадий, в ходе выполнения которых формируется техническое задание, осуществляются эскизное проектирование, изучение объекта регулирования и составление его математической модели, техническое и рабочее проектирование, изготовление спроектированной системы и, наконец, ее внедрение и оценка эффективности [2].

Из сказанного следует, что задача построения математической модели объекта является системной задачей, требующей для своего решения системного подхода. Это значит, что выбор критерия приближения модели объекта к реальному объекту должен зависеть от алгоритма функционирования регулятора, для отыскания которого и строится модель объекта. Таким образом, задача построения модели объекта оказывается противоречивой уже в своей постановке: для построения модели объекта требуется знать алгоритм функционирования регулятора, для определения которого и нужна модель.

Постановка задачи. Задачей данной работы является оптимизирующая настройка ПИД-регуляторов, регулирующих подачу компонентов газовой смеси в ГДК. Для выполнения поставленной задачи необходимо установить необходимые коэффициенты K_p передачи ПИД-регуляторов. Коэффициенты передачи определяются на основе анализа данных о ходе эксплуатации регуляторов.

Как следствие необходимо организовать сбор и сохранение экспериментальной информации о работе системы для дальнейшего использования. Кроме того, необходимо реализовать математический метод анализа полученной информации с целью определения необходимых оптимизационных коэффициентов K_p. Обмен информацией с регулятором возможен посредством последовательного порта с интерфейсом RS-232, который имеется в наличии и у устройства управления, и у устройства оптимизационного регулирования ПК. Для плодотворного использования последовательный порт необходимо настроить соответствующим образом.

Информацию, полученную в ходе эксперимента, целесообразно сохранять в базе данных для облегчения ее анализа при подсчете оптимизационных коэффициентов и в иных инженерных нуждах. [3, 4]. Кроме того, целесообразно будет организовать визуальное информирование пользователя о ходе процессов, происходящих в лазерной установке во время ее эксплуатации.

Программная часть алгоритма регулирования ЭУ

Для наглядности рассмотрим алгоритм функционирования разрабатываемой программной системы представленной на рисунке «Алгоритм работы программы» (рис. 1).

Для реализации поставленных задач программная система должна работать таким образом [5]:

– после запуска организация передачи данных между ПК и УСО;

– после завершения сбора информации производится подсчет оптимизационных коэффициентов;

– на заключительном этапе работы программной системы оптимизационные коэффициенты передают-

ся посредством последовательного порта УСО;

– также после завершения сбора информации возможен доступ пользователей к накопленной информации с целью ее анализа.

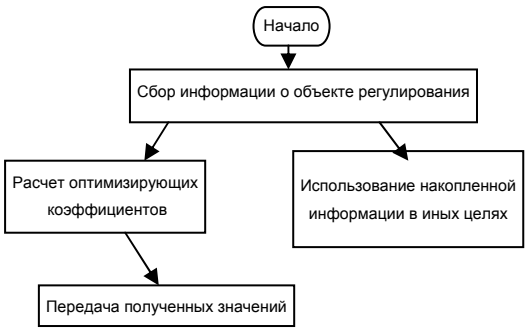

Рис. 1. Алгоритм работы программы

Для построении программной системы необходимо разбить блок сбор информации о системе на более мелкие. Такие как:

– организация передачи данных между ПК и УСО;

– сохранение полученных от УСО экспериментальных данных в «журнале испытаний».

В разработанном программном продукте, следуя принципам ООП, поставленные задачи следует решать построением отдельных модулей. Рассмотрим каждый модуль в отдельности.

Модуль «Организация передачи данных между ПК и УСО» активизируется при запуске программного продукта и состоит из:

– открытие порта. Открывается COM2 порт как файл с помощью функций WinAPI, так как организация работы с портом в OS Windows другими способами практически тяжело реализуема. Кроме того функции WinAPI дают возможность сравнительно быстрой и удобной для программиста настройки параметров порта.

Непосредственно при открытии порта мы можем задать только весьма небольшую часть из списка свойств, доступ к которым предоставляют функции WinAPI. При открытии можно задать только наиболее масштабные свойства порта, такие как: синхронная или асинхронная организация передачи данных, вид доступа к файлу, чтение, запись, чтение и запись, и другие;

– настройка порта. Так как после открытия настройки порта имеют параметры, по умолчанию во многом не удовлетворяющие требованиям, предъявляемым разрабатываемой программной системой, для адаптации настроек последовательного порта производим его настройку.

Операция настройки представляет собой запись необходимых управляющих параметров в основную структуру настройки работы с портами, структуру DCB. Также для организации обмена данными необходимо записать информацию в структуру COMMTIMEOUTS относительно временных параметров работы порта. Настроенную информацию проверяем в структуре COMMPROP.

Модуль «Сохранение полученных от УСО экспериментальных данных в «журнале испытаний»» активизируется пользователем при нажатии на кнопку «Старт». При этом информация, которая поступает в буфер порта от УСО, записывается в, так называемый, журнал наблюдений. Журнал наблюдений представляет собой базу данных, построенную в «MySQl». Этот факт позволяет упростить программу, возложив значительную часть математических операций на SQL-запросы, выполняющиеся базой данных. Кроме того, модуль «Построение журнала наблюдений» обязан самостоятельно завершать заполнение базы данных при наличии достаточного количества информации о системе. Для реализации задачи самостоятельного завершения заполнения базы данных в программу введен таймер, который при прохождении заданного пользователем или заданного по умолчанию промежутка времени завершает процесс сбора информации.

Модуль «Расчет оптимизационных коэффициентов» производит расчет оптимизационных коэффициентов следуя математической модели, приведенной в подпункте постановке задачи. После подсчета оптимизационных коэффициентов программа выводит их в компоненты TEdit, расположенные на форме, и запрашивает разрешения у пользователя на установку подсчитанных коэффициентов. При разрешении пользователя эти коэффициенты передаются через COM2 порт к УСО.

Описание разработанной программной системы

Обоснование выбора среды разработки. Современные средства разработки ПО характеризуются большим разнообразием критериев, используя которые разработчик имеет возможность автоматизировать процесс разработки приложений. Так, в настоящее время инструментальные средства позволяют:

– создавать интерфейс, используя стандартные компоненты;

– передавать управление различным процессам, в зависимости от состояния системы;

– создавать оболочки для баз данных, как и сами базы данных;

– разрабатывать более надежное ПО, путем обработки исключительных ситуаций, возникающих при некорректной работе ПО.

Современные средства разработки характеризуются следующими параметрами:

– поддержка объектно-ориентированного стиля программирования;

– возможность использования CASE-технологий, как для проектирования разрабатываемой системы, так и для разработки моделей реляци-

онных баз данных;

– использование визуальных компонент для наглядного проектирования интерфейса;

– поддержка БД;

– возможность использования алгоритмов реляционной алгебры для управления реляционными базами данных;

– возможность синхронизации составных частей проекта (предоставляется при разработке больших программных комплексов).

Широкий спектр современных языков программирования представлен языками высокого уровня, то есть языками, ориентированными под выполнение определенного круга задач. К примеру, Delphi лучше использовать для написания баз данных, JavaScript для создания Web-приложений. Для разработки программных продуктов, управляющих аппаратными средствами ЭВМ, зачастую используются языки Ассемблер и C++.

Язык Ассемблер является языком более низкого уровня в сравнении с языком C++. Код программ, написанных на Ассемблере, выполняется процессором значительно быстрее, однако код получается громоздким и его написание требует больших временных затрат. Поэтому, там где к программе не предъявлены требования высокой скорости выполнения, целесообразнее использовать язык C++.

В качестве программной среды реализации была выбрана среда Borland C++ Builder версии 6.0. Данный выбор обусловлен следующими факторами, которые выдвигают Borland C++ Builder 6.0 на авангардные позиции систем объектно-ориентированного программирования для быстрой разработки современного математического обеспечения персональных компьютеров:

– язык C++ – один из самых быстродействующих прикладных алгоритмических языков;

– высокопроизводительный 32-разрядный оптимизирующий компилятор;

– наличие большого объема обучающего и справочного материала;

– возможность полного доступа к Windows API;

– широко развита система создания классов, объектов, библиотек;

– поддержка объектно-ориентированного программирования;

– широкая библиотека визуальных компонент;

– наличие интегрированной среды разработки;

– существование механизмов двунаправленной разработки;

– наличие визуального наследования форм;

– быстрый инкрементальный компоновщик приложений;

– удобная визуальная среда разработки (Builder, Visual Studio).

Компилятор в машинный код. Компилятор, встроенный в C++Builder, обеспечивает высокую производительность. Этот компилятор в настоящее время является самым быстрым в мире (неофициальные данные). Он предлагает легкость разработки и быстрое время проверки готового программного блока, характерного для языков четвертого поколения (4GL), и в то же время обеспечивает качество кода, характерного для компилятора 3GL.

В процессе построения приложения разработчик выбирает из набора компонент готовые компоненты подобно тому, как это делается при конструировании любых форм, состоящих из типовых объектов. В этом смысле проектирование в C++Builder мало чем отличается от проектирования в интерпретирующей среде, однако после выполнения компиляции мы получаем машинный код, в то время как существуют компиляторы (Visual Basic, например, или Power Builder), превращающие программу в так называемый p-код, который затем интерпретируется виртуальной p-машиной. Это не может не сказаться на фактическом быстродействии готового приложения.

Объектно-ориентированные модули и компоненты. При построении готовых форм из типовых объектов, которые вкладываются друг в друга и образуют желаемую программно-архитектурную форму, разработчик работает с моделью подобно конструктору-дизайнеру. Программный код всегда генерируется автоматически при выборе, настройке и удалении объектов.

Никаких ограничений по типам объектов, которые могут создавать разработчики, не существует. Действительно, все в C++Builder написано на нем же, поэтому разработчики имеют доступ к тем же объектам и инструментам, которые использовались для создания среды разработки. В результате нет никакой разницы между объектами, поставляемыми Borland или третьими фирмами, и объектами, которые можно создать.

В стандартную поставку C++Builder входят основные объекты, которые образуют удачно подобранную иерархию из 270 базовых классов [4]. Среда C++Builder включает в себя полный набор визуальных инструментов для скоростной разработки приложений (RAD – rapid application development), поддерживающей разработку пользовательского интерфейса и подключение к корпоративным базам данных. VCL – библиотека визуальных компонент. Она включает в себя стандартные объекты построения пользовательского интерфейса, объекты управления данными, графические объекты, объекты мультимедиа, диалоги и огромное количество свободно распространяемых freeware-компонент, которыми можно надстраивать C++Builder.

В C++Builder осуществлена поддержка OLE-управляющих компонент, которые могут добавляться посредством инсталляционной опции в меню Component. Например, с помощью этих функций очень легко можно написать код, который откроет

Word, Excel и другие компоненты, создаст документ и впишет в него информацию. Естественно, что в C++Builder хорошо проработана и событийная модель. С каждым объектом может быть связан целый ряд событий, которые программируются и задаются в специальном окне стандартного Инспектора объектов. Событийная модель в Windows всегда была сложна для понимания и отладки. Но именно разработка интерфейса в C++Builder является самой простой задачей для программиста.

Немалое значение при выборе Borland C++ Builder 6.0 в качестве средства для разработки играет возможность использования большого количества встроенных визуальных компонент, как для разработки интерфейса, так и для других необходимых функций при проектировании.

При создании программного продукта главным критерием выбора программных средств разработки являлись:

- скорость разработки приложений;
- возможность быстрого внесения изменений в программу;
- возможность редактирования и просмотра БД, используя средства разработки.

Как дополнение к перечисленному выше, можно указать, что время разработки зависит от: поддержки выбранным инструментарием ОС, аппаратной поддержки, необходимой для их оптимального функционирования; наличия предварительного опыта у разработчиков в использования соответствующих программных средств. Обеспечить минимальное время разработки можно только при выполнении этих условий.

Исходя из приведенных требований, выделим следующие характеристики средств разработки программного обеспечения:

- наличие опыта разработки с использованием данного программного продукта;
- требования по ресурсам;
- наглядность разработки интерфейса;
- скорость работы разработанного программного обеспечения;
- обработка исключительных ситуаций;
- время создания разработанного программного обеспечения;
- удобство эксплуатации.

В результате выполненного анализа инструментальных средств был окончательно решён вопрос по выбору средства разработки в пользу Borland C++ Builder 6.0, как наиболее оптимального средства разработки с точки зрения разработчика.

Структурная схема программной системы. Разрабатываемая программная система предназначена для выполнения ряда задач производства, которые направлены на увеличение экономического эффекта от эксплуатации лазерного оборудования в производственных условиях.

Безусловно, для реализации поставленной задачи главным является расчет оптимизационных коэффициентов на основе собранной информации о протекании процессов во время эксплуатации ЛУ. Но не следует также упускать из вида тот факт, что программа накапливает экспериментальную информацию о процессах происходящих в ЛУ. Что может быть использовано в дальнейшем для разработки новых типов лазерного оборудования.

Программную систему можно представить в виде таких взаимосвязанных функций:

– функция обмена информацией с УСО ЛУ;

– накопление получаемой информации;

– обработка полученной в ходе эксперимента информации для получения оптимизирующих коэффициентов;

– передача устройству связи с объектом полученных в ходе обработки экспериментальной информации, оптимизационных коэффициентов.

– организация интерфейса роботы пользователя с накопленной экспериментальной информацией.

При этом функции обмена информацией с УСО ЛУ и накопления получаемой информации работают циклически, вызывая друг-друга на первом этапе выполнения программы. При их завершении вызывается функция-обработчик накопленной информации. Эта функция, согласно выбранному методу оптимизации, производит расчет величины целесообразных оптимизационных коэффициентов, которые будут переданы УСО ЛУ по окончании их расчета, посредством функции обмена информации с УСО ЛУ.

В свою очередь организация интерфейса работы пользователя с накопленной экспериментальной информацией реализована в виде отдельной функции, которая доступна для выполнения, когда программная система не производит сбор информации, но имеет при этом информацию, накопленную в ходе предыдущих опытов.

Описание работы программы. Доступ к составным частям программы производится переключением панелей. На форме реализовано три панели: «Расчет», «База данных», «Справка». Представлено на рис. 2.

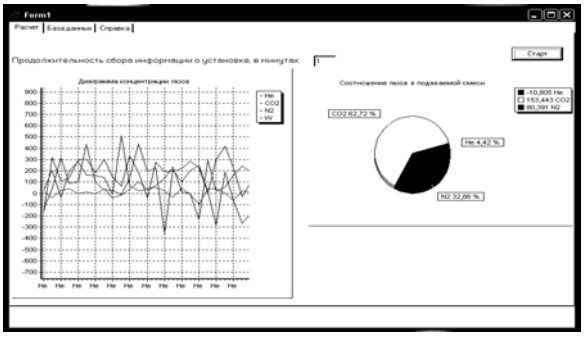

Рис. 2. Внешний вид разработанной программной системы

Панель «Расчет» (рис. 2) предусмотрена для задания времени сбора информации об управляемом

объекте, запуска процесса регулирования, а также информирования пользователя о тенденции изменения входящей информации в графической форме.

Панель «База данных» предусмотрена для просмотра пользователем информации о работе системы в табличной форме, представлена на рис. 3.

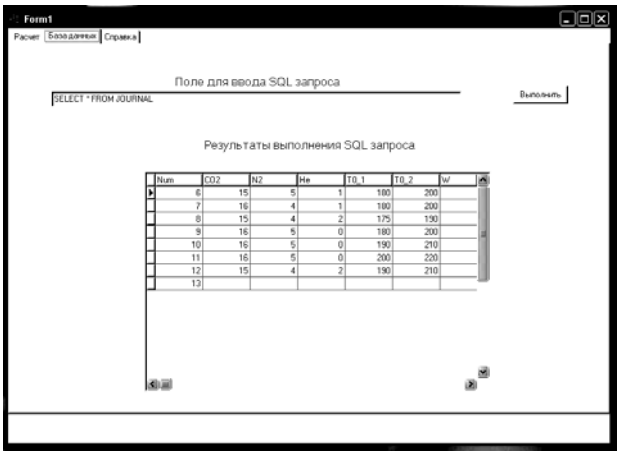

Рис. 3. Внешний вид панели «База данных»

Панель «Справка» используется для информирования пользователя программы о ее названии, версии, а также о разработчиках данного программного продукта.

При запуске программы и наступлении события «OnCreate» производится открытие порта COM2 для доступа к УСО, а также производится настройка параметров порта. Хотя во время сбора информации о системе порт COM2 используется исключительно для чтения, порт открывается для чтения и записи, чтобы избежать необходимости повторного открытия порта для записи коэффициентов оптимизации.

Пользователь на панели «Расчет» в поле «Продолжительность процесса в минутах» задает продолжительность процесса сбора информации о системе, по умолчанию этот параметр равняется 120 минут. Этот параметр передается таймеру. По истечении заложенного интервала времени происходит событие «OnTimer». Обработчик этого события перехватывает нить программы. То есть прекращает сбор информации о системе и, обрабатывая собранную базу данных, подсчитывает необходимые оптимизационные коэффициенты, и выводит их на экран. При согласии пользователя коэффициенты передаются УСО. Начало процесса сбора информации о системе, а также инициализации и запуска таймера, происходит при нажатии пользователем кнопки «Старт».

На панели «База данных» пользователю предоставляется возможность получения информации из собранной программой базы данных путем ввода SQL запроса в поле «SQL запрос» и нажатии кнопки «Выполнить». Полученная в результате запроса информация будет отображаться в компоненте «DBGrid», который занимает все свободное пространство данной панели.

Описание системы защиты информации

Необходимость создания системы защиты информации. Уже в первых работах по защите информации были изложены основные постулаты, которые не утратили своей актуальности и по сей день [4]: абсолютную защиту создать нельзя, система защиты информации должна быть комплексной, СЗИ должна быть адаптируемой к изменяющимся условиям. К этим аксиомам нужно добавить и другие суждения. Во-первых, СЗИ должна быть именно системой, а не простым, во многом случайным и хаотичным набором некоторых технических средств и организационных мероприятий, как это зачастую наблюдается на практике. Во-вторых, системный подход к защите информации должен применяться, начиная с подготовки технического задания и заканчивая оценкой эффективности и качества СЗИ в процессе ее эксплуатации.

Массовое создание, внедрение и эксплуатация информационных систем привели к возникновению спектра новых проблем в сфере безопасности личности, общества и государства. Потребность в обеспечении безопасности связана с тем, что существует множество субъектов и структур, весьма заинтересованных в чужой информации и готовых заплатить за это высокую цену.

Результативное решение задач анализа и синтеза СЗИ не может быть обеспечено одними лишь способами умозрительного описания их поведения в различных условиях – системотехника выдвигает проблемы, требующие количественной оценки характеристик. Такие данные, полученные экспериментально или путем математического моделирования, должны раскрывать свойства СЗИ. Основным из них является эффективность, понимается степень соответствия результатов защиты информации поставленной цели [5]. Последняя, в зависимости от имеющихся ресурсов, знаний разработчиков и других факторов, может быть достигнута в той или иной мере, при этом возможны альтернативные пути ее реализации. Эффективность имеет непосредственную связь с другими системными свойствами, в том числе качеством, надежностью, управляемостью, помехозащищенностью, устойчивостью. Поэтому количественная оценка эффективности позволяет измерять и объективно анализировать основные свойства систем на всех стадиях их жизненного цикла, начиная с этапа формирования требований и эскизного проектирования.

Нормативные документы по оценке безопасности ИТ практически не содержат конкретных методик, в результате чего величина разрыва между общими декларациями и конкретным инструментарием по реализации и контролю их положений является недопустимой. Исходя же из своего предназначения, методическая база должна охватывать все критически важные аспекты обеспечения и проверки выполнения требований, предъявляемых к инфор-

мационной безопасности. Объективным видом оценки эффективности СЗИ является функциональное тестирование, предназначенное для проверки фактической работоспособности реализованных механизмов безопасности и их соответствия предъявленным требованиям, а также обеспечивающее получение статистических данных. В силу того, что средства безопасности обладают ограниченными возможностями по противодействию угрозам, всегда существует вероятность нарушения защиты, даже если во время тестирования механизмы безопасности не были обойдены или блокированы.

Тестирование по методу "черного ящика" предполагает отсутствие у тестирующей стороны каких-либо специальных знаний о конфигурации и внутренней структуре объекта испытаний. При этом против объекта испытаний реализуются все известные типы атак, и проверяется устойчивость системы защиты в отношении этих атак. Используемые методы тестирования эмулируют действия потенциальных злоумышленников, пытающихся взломать систему защиты. Метод "белого ящика" предполагает составление программы тестирования на основании знаний о структуре и конфигурации объекта испытаний.

В ходе тестирования проверяется наличие и работоспособность механизмов безопасности, соответствие состава и конфигурации системы защиты требованиям безопасности и существующим рисками. Выводы о наличие уязвимостей делаются на основании анализа конфигурации используемых средств защиты и системного ПО, а затем проверяются на практике. Основным инструментом анализа в данном случае являются программные агенты средств анализа защищенности системного уровня, рассматриваемые ниже.

Реализация системы защиты информации. В программной среде оптимизации регулирования газового контура лазерной установки основным объектом внимания злоумышленников является база данных, построенная программой в процессе сбора информации о системе. Такое внимание к базе данных объясняется дефицитом практической информации о функционировании ЛУ. В связи с этим вход в базу данных закрыт паролем. Перед началом сеанса пользователь должен непременно ввести пароль, блокирующий базу данных, в противном случае он не сможет работать даже с главной формой программного продукта, так как она использует доступ к базе данных.

Выводы

В рамках данной работы была разработана программная система, накапливающая информацию о характеристиках имитатора лазерной установки в ходе процесса эксплуатации, а также производящая необходимые манипуляции с полученной информацией, в частности получение оптимизационных коэффициентов. Получаемые таким образом оптимизационные коэффициенты являются более точными.

Новизна данной работы состоит в создании системы, позволяющей оптимизировать конкретную установку в ходе ее нормальной эксплуатации с учетом индивидуальных особенностей конкретной установки.

Научная ценность проведенной работы: накопленная в ходе проведения ряда экспериментов, практическая информация может быть использована в дальнейшем для конструирования более совершенной электронной техники.

Актуальность разработки обуславливается значительный экономический эффект который может быть достигнут при внедрении данного программного продукта в производственных условиях.

Список литературы

1. Федеральный портал «Инженерное образование». – [Электронный ресурс]. – Режим доступа к документу: http://techno.stack.net/.

2. Архангельский А.Я. C++Builder 6 : cСправочное пособие / А.Я. Архангельский. – М.: Бином-Пресс, 2002. – Книга 1. Язык C++. – 544 с.

3. Коннолли Т. Базы данных: проектирование, реализация, сопровождение. Теория и практика: пер. с англ. / Т. Коннолли, К. Бег, А. Страчан. – 2-е изд. – М.: Издательский дом «Вильямс», 2000. – 1120 с.

4. Хансен Г.Д. Базы данных: разработка и управление / Г.Д. Хансен, Д.А. Хансен. – М.: БИНОМ, 1999. – 704 с.

5. Автоматизация настройки систем управления / В.Я. Ротач, В.Ф. Кузищин, А.С. Клюев и др.; под ред. В.Я. Ротача. – М.: Энергоатомиздат, 1984. – 367 с.

Поступила в редколлегию 9.12.2008

Рецензент: д-р техн. наук, проф. С.Ю. Шабанов-Кушнаренко, Харьковский национальный университет радиоэлектроники, Харьков.

ПРОГРАМНІ ЗАСОБИ АВТОМАТИЗОВАНОГО ПРОЕКТУВАННЯ ПРОПОРЦІЙНО-ІНТЕГРАЛЬНО-ДИФЕРЕНЦІАЛЬНИХ РЕГУЛЯТОРІВ

Н.В. Голян, Ю.П. Шабанов-Кушнаренко

Розроблено програмний продукт, що дозволяє автоматично формувати: коефіцієнти пропорційно-інтегрально-диференціальних регуляторів і журнал-відомість роботи установки на кожному етапі оптимізації.

Ключові слова: *пропорційний-інтегральний-диференціальні регулятори, автоматизація робочого місця, база даних.*

PROGRAMMATIC FACILITIES OF THE AUTOMATED PLANNING OF PROPORTIONAL-INTEGRAL-DIFFERENTIAL REGULATORS

N.V. Golyan, Yu.P. Shabanov-Kushnarenko

A software product, allowing automatically to form, is developed: coefficients of proportional-integral-differential regulators and magazine-list of work of setting on every stage of optimization.

Keywords: *proportional-integral-differential regulators, automation of workplace, database.*