АВТОМАТИЗАЦИЯ ПРОИЗВОДСТВЕННЫХ ПРОЦЕССОВ

ЛЕКЦИЯ 14

4. ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ РСУ

Программа является неотъемлемой частью любого цифрового устройства. Без программы ПЛК – это просто «железо», которое нельзя ни для чего приспособить. И только поместив в него программу, мы оживляем это «железо», теперь оно способно выполнять многие удивительные функции: производить безошибочно сложнейшие вычисления, обобщать и систематизировать данные, управлять сложными процессами в реальном масштабе времени, предупреждать об опасных ситуациях, предоставлять информацию оператору в удобном виде и выполнять многие другие действия.

Однако все это совместно и хорошо работает при условии, что используются нужные программы и отдельные элементы РСУ «понимают» друг друга. РСУ содержит множество территориально распределенных ПЛК, МСД и других цифровых устройств, выполняющих функции сбора и обработки данных. Каждое из этих устройств работает по своим программам. В то же время все эти устройства составляют единое целое и должны работать слаженно для достижения общей цели управления. Следовательно, работа всех программ РСУ должна быть согласованной и эти программы должны быть приспособлены для обработки одних и тех же данных. Рассмотрим коротко понятия аппаратного и программного обеспечения.

15 Понятия аппаратного и программного обеспечения вычислительной системы

Состав цифровой вычислительной системы называется *конфигурацией*. Приборы и устройства вычислительной системы называют *аппаратной конфигурацией, или аппаратным обеспечением*. Совокупность и состав программ этой системы образуют *программную конфигурацию*, или составляют *программное обеспечение*. Ясно, что программы могут быть реализованы аппаратными средствами, кроме того, конечная цель любой программы – управление аппаратными средствами. Поэтому программное и аппаратное обеспечение в вычислительной системе работают в неразрывной взаимосвязи и в непрерывном взаимодействии. Тем не менее аппаратные и программные средства по принципу своей работы сильно отличаются, поэтому их принято рассматривать отдельно.

Аппаратное обеспечение. Современные цифровые приборы и вычислительные комплексы имеют блочно-модульную конструкцию, поэтому аппаратную конфигурацию, необходимую для исполнения конкретных видов работ, можно собирать из готовых узлов и блоков. По способу расположения устройств относительно центрального процессорного устройства (ЦПУ) различают *внутренне* и *внешние* устройства. К внешним относятся устройства ввода-вывода данных (периферийные устройства) и устройства для длительного хранения данных. Согласование между отдельными узлами и блоками выполняют с помощью переходных аппаратно-логических устройств, называемых *аппаратными интерфейсами*. Стандарты на аппаратные интерфейсы в вычислительной технике называют *протоколами*. Итак, *протокол – это совокупность технических условий, которые должны быть обеспечены разработчиками устройств для успешного согласования их работы с другими устройствами*. Многочисленные аппаратные интерфейсы условно разделяются на *параллельные* и *последовательные*. Передача параллельным интерфейсом по восьми каналам (проводникам) производится байтами, поэтому скорость передачи – байт/секунду (байт/с, Кбайт/с, Мбайт/с). Передача последовательным интерфейсом производится по одному каналу битами, здесь скорость передачи – бит/секунду (бит/с, Кбит/с, Мбит/с).

Программное обеспечение. *Компьютерная программа – это набор упорядоченных команд и данных, которые описывают операции в форме, приемлемой для их выполнения*

компьютером. Каждый компьютер выполняет свою, уникальную задачу и для ее решения нужна программа. Гипотетически возможно создание для каждого вычислительного устройства своей, уникальной единой программы работы этого устройства для решения данной задачи. Ясно, что из-за сложности задачи создания такой программы это невозможно сделать. Как обычно, здесь используется декомпозиция – разделение общей сложной задачи на ряд более простых, частных задач и раздельное их решение. Решения затем объединяются (композиция), что позволяет решить общую задачу достаточно простыми средствами. Естественным и экономным направлением декомпозиции при создании программ является такое разделение, при котором для решения частных задач создаются универсальные подпрограммы, пригодные для решения частных задач широкого класса. В итоге после объединения подпрограмм создается общая программа, позволяющая решать данную задачу, состоящая из взаимодействующих между собой подпрограмм.

На практике нашло применение разделение программного обеспечения (ПО) на уровни пирамидальной конструкции, при котором каждый следующий уровень опирается на программное обеспечение предыдущего уровня. Можно выделить четыре уровня: базовый, системный, служебный и прикладной (рисунок 15.1).

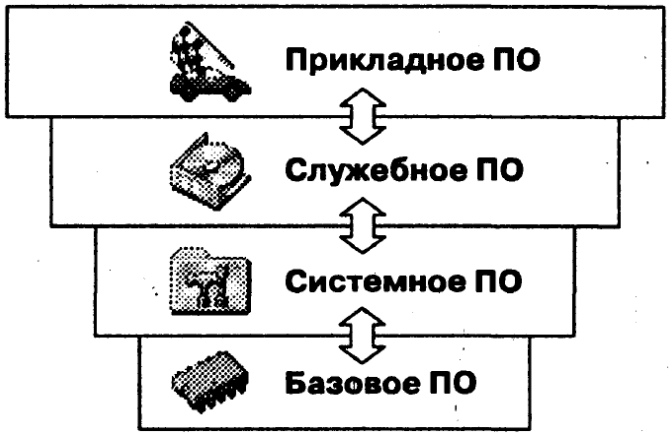

Рисунок 15.1 – Разделение программного обеспечения (ПО) вычислительного устройства на уровни

Базовое ПО отвечает за взаимодействие с базовыми аппаратными средствами. Как правило, это ПО непосредственно входит в состав базового оборудования и записывается в свое постоянное запоминающее устройство (ПЗУ) на этапе производства этого оборудования.

Системное ПО обеспечивает взаимодействие прочих программ вычислительной системы с программами базового уровня и аппаратным обеспечением, т.е. выполняет «посреднические» функции. Совокупность программ системного уровня образуют ядро *операционной системы* (ОС). Программы ОС, отвечающие за взаимодействие с конкретным устройством, называется *драйвером* устройства, отвечающие за взаимодействие с пользователем – называются *средствами обеспечения пользовательского интерфейса*. Вычислительная система, в которой установлена ОС, готова для установки прикладного ПО. Элементы системного ПО являются универсальными и при создании данной вычислительной системы выбираются из готового набора. Следует заметить, что от выбора системного ПО во многом зависят показатели работы вычислительной системы. В РСУ используются ОС реального времени (ОС РВ). Мы уже говорили о том, что при управлении динамическими процессами нужно опрашивать управляемые переменные и выдавать управляющие воздействия через фиксированные моменты времени. ОС РВ как раз обеспечивает это требование. Наиболее распространены ОС РВ Windows CE, QNX, Neutrino и OS-9.

Служебное ПО предназначено для организации работ по проверке, наладке и настройке вычислительной системы. Программы этого уровня называются *утилитами*. Они служат для расширения функций ОС.

Прикладное ПО – это комплекс прикладных программ, предназначенных для выполнения конкретных заданий вычислительной системы. Как раз прикладные программы обеспечивают весь обширный спектр задач, решаемых этой системой.

При таком разделении общего ПО задача программирования конкретной вычислительной системы переносится на задачу создания прикладного ПО для этой системы. На начальном этапе применения ЭВМ в системах автоматизации использовалась одна центральная ЭВМ с устройствами связи с объектом (УСО), к которой подключались датчики, ИУ и устройства связи с оператором. Прикладное ПО для ЭВМ на этом этапе писали профессиональные программисты на языках высокого уровня. Однако практика показала, что при таком подходе прикладное ПО оказывалось очень дорогим и малоэффективным. Технологам трудно было заранее точно сформулировать требования к программам с учетом особенностей составления этих программ, в итоге после создании программы требовался длительный процесс «доводки» и отладки этих программы. Естественным выходом является участие технологов в процессе программирования, но для этого нужно было упростить процесс программирования. С расширением применения РСУ возникла также проблема согласования работы программ разных блоков и модулей РСУ, что еще усложняло задачу. Выходом из первого затруднения, как и ранее, явилась декомпозиция, из второго – стандартизация программных интерфейсов и применение открытых систем. Декомпозиция заключалась в разработке специализированных средств программирования для систем автоматизации.

Рассмотрим более подробно основные причины необходимости разработки специализированных средств программирования для систем автоматизации:

1) Требование надежности прикладного ПО. ПО, написанная целиком на алгоритмическом языке для конкретного заказа, содержала слишком много программного кода, на тщательную разработку и тестирование которого не хватало времени.

2) Сжатые сроки внедрения и ограниченная стоимость работ. Для создания системы в короткий срок при ограниченном бюджете требовалось большое количество готовых программных компонентов, уже написанных и тщательно оттестированных.

3) Необходимость модификации системы в процессе ее эксплуатации. Внести изменения в в специализированную программу мог только программист, ее написавший и работающий уже на другом предприятии. Поэтому вместо модификации приходится писать программу заново.

4) Требование совместимости с другими системами автоматизации, работающими на том же предприятии. Для этого необходимы стандартные интерфейсы между программами, созданными разными производителями на разных аппаратно-программных платформах.

5) Высокие требования к качеству пользовательского интерфейса. Ограниченный бюджет времени и финансовых ресурсов не позволял разработать достаточно хороший программный интерфейс на универсальных языках.

Перечисленные причины привели к следующему разделению труда по созданию программных средств для систем автоматизации: фирмы, специализирующиеся на прикладном ПО создают универсальные системы программирования задач автоматизации (SCADA-пакеты и стандартные средства программирования контроллеров), а инжиниринговые фирмы (системные интеграторы) адаптируют эти средства к нуждам конкретного заказчика. SCADA – аббревиатура Supervisory Control And Data Acquisition – диспетчерское управление и сбор данных, о SCADA – системах речь пойдет ниже. В результате достигается решение всех перечисленных выше проблем. Теперь благодаря существенному упрощению процесса

программирования изменения в алгоритмы управления могут быть внесены технологами эксплуатирующей организации без привлечения системных интеграторов или программистов.

В настоящее время заказные программы для сложных систем естественным путем вытеснены с рынка промышленной автоматизации SCADA-пакетами и аналогичными универсальными средствами, а также средствами программирования контроллеров на языках стандарта МЭК 61131-3.

Для простых систем автоматизации SCADA-пакеты часто оказываются все-таки дорогими. Простой пример: нужно автоматизировать процесс сжигания топлива в небольшой котельной. Нужно создать 2-3 контура регулирования и вывести информацию по не более 10-и переменным. Приобретать SCADA-пакет, осваивать его и адаптировать к данной задаче слишком долго и дорого. Здесь по-прежнему оказывается выгодным создание специализированных программ. Экономически целесообразно также разрабатывать специализированные программы для серийно тиражируемых однотипных систем автоматизации. Тем не менее специализированные программы содержат в своем составе универсальные подпрограммы, настроенные на выполнение конкретной задачи.

16 Понятие открытой системы

При делении РСУ на модули открывается возможность выбирать эти модули из числа выпускаемых промышленностью недорогих универсальных модулей. Однако при этом возникает проблема аппаратной и программной совместимости этих модулей. Для достижения совместимости интерфейс, конструктив и выполняемые функции модулей должны быть стандартизованы. Системы, удовлетворяющие такому требованию, называются *открытыми системами*.

Открытую систему сегодня определяют как «исчерпывающий и согласованный набор международных стандартов (официальных и общепринятых (де-факто)) на информационные технологии и профили функциональных стандартов, которые реализуют открытые спецификации на интерфейсы, службы и поддерживающие их форматы, чтобы обеспечить взаимодействие (интероперабельность) и мобильность программных приложений, данных и персонала». Это определение дано специалистами IEEE. Интероперабельность – способность продукта или системы, интерфейсы которых полностью открыты, взаимодействовать и функционировать с другими продуктами или системами без каких-либо ограничений доступа и реализации. Другое, более понятное, но более узкое определение: «Открытая система - это система, которая состоит из компонентов, взаимодействующих друг с другом через стандартные интерфейсы».

В открытой системе допускается замена любого модуля на аналогичный модуль другого производителя, *имеющийся в свободной продаже*, а интеграция системы с другими системами выполняется легко. В вычислительной системе предметами стандартизации могут быть [4]:

1) физические интерфейсы, протоколы обмена, методы контроля ошибок, системы адресации, форматы данных, типы организации сети, интерфейсы между программами, диапазон изменения аналоговых сигналов.

2) Пользовательские интерфейсы, языки программирования контроллеров, управляющие команды ввода-вывода, языки управления базами данных, ОС, средства связи аппаратуры с ПО.

3) Конструктивные элементы шкафов, стоек, корпусов, разъемов, крепежных элементов.

Для SCADA – системы открытость означает

1) совместимость со стандартом OPC (о стандарте ОПС см. ниже);

2) совместимость с устройствами с широко доступными ОС;

3) совместимость с ActiveX, COM и DDL компонентами других производителей;

4) поддержка языков стандарта МЭК 61131-3 программирования контроллеров;

5) наличие встроенного стандартного алгоритмического языка, например, Visual Basic;

6) возможность работы с разным количеством тегов (тег – это канал ввода или вывода SCADA – системы, содержащий какое-то значение).

Одним из основных составляющих открытых систем является стандарт OPC. Коротко рассмотрим, что это такое.

17 Стандарт OPC

OPC (Open Platform Communications, открытая платформа связи) – семейство программных технологий, предоставляющих единый интерфейс для управления объектами автоматизации и технологическими процессами. Стандарт OPC разработан международной организацией OPC Foundation, членами которой являются ведущие в мире фирмы в области автоматизации. Главной целью этого стандарта явилось обеспечение возможности совместной работы (интероперабельность) средств автоматизации, в разных промышленных сетях и на разных аппаратных платформах, производимых разными фирмами. После появления стандарта OPC практически все SCADA-пакеты были представлены, как OPC-клиенты, а каждый производитель аппаратного обеспечения стал снабжать свои ПЛК, МСД, интеллектуальные датчики и ИУ стандартными OPC-серверами. Применение OPC-сервера при разработке заказных программ позволяет предоставлять разработчику простой и удобный метод доступа к аппаратуре.

Стандарт OPC состоит из нескольких частей [4]:

1) OPC DA спецификация для обмена данными между клиентом (например SCADA) и аппаратурой (контроллерами, МСД, и др.) в реальном времени;

2) OPC Alarms & Evets (A&E) – спецификация для уведомления клиента о событиях и сигналах тревоги. Этот сервер пересылает аварийные сигналы, действия оператора, информационные сообщения, результаты контроля состояния системы;

3) OPC HDA (Historial Data Assess) – спецификация для доступа к предыстории процесса (к сохраненным в архиве данным). Сервер обеспечивает унифицированный способ доступа с помощью DCOM-технологии. Обеспечивает чтение, запись и изменение данных;

4) OPC Batch – спецификация для особых физико-химических технологий, которые не являются непрерывными;

5) OPC Data eXchange – спецификация для обмена данными между двумя OPS DA-серверами через сеть Ethernet;

6) OPC Security – спецификация, которая определяет методы доступа клиентов к серверу, обеспечивающие защиту важной информации;

7) OPC XML-DA – набор гибких, согласующихся друг с другом правил и форматов для представления первичных данных;

8) OPC Complex Data – дополнительные спецификации для OPS DA и OPS XML-DA;

9) OPC Commands – набор программных интерфейсов для работы с командами.

Из перечисленных спецификаций в России широко используются только две: OPC DA и OPC HDA. Более подробно со спецификациями можно ознакомиться по литературе [4] и др.

Наряду с успешным применением OPC технологии практика выявила ее серьезные недостатки. В 2006 году OPC Foundation предложил новую стандартную спецификацию, получившую название «OPC Unified Architecture» – «OPC с унифицированной архитектурой»,

которая рассматривается как OPC-стандарт нового поколения. Более подробно с OPC Unified Architecture также можно ознакомиться по литературе [4] и др.