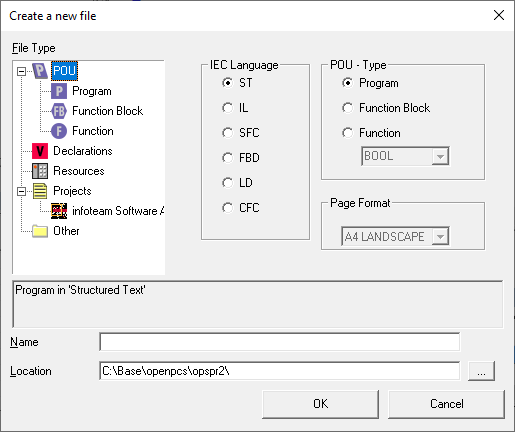
Система программирования Open PCS и другие,. Языки программирования. Конфигурирование модулей ввода/ вывода ПЛК.

**Система программирования ПЛК и ПР.**

1. Step 7
2. openPCS
3. owen logic
4. logo soft comfort
5. codesys
6. ispsoft
7. wplsoft

Подключение и конфигурация модулей ПЛК и ПЛК согласно документации по программному обеспечению.

**Будет рассмотрено в практиках.**

** openPCS предложение по обеспечению выбора как и что.**

**Система программирования ПЛК OpenPCS**

**Цель работы:** Создание ресурса, задач, программ на языках стандарта IEC 6 1131-3 и их отладка в PLC-симуляторе OpenPCS 2004. Все программы выполняют одну и ту же задачу, хорошо известную по лабораторным работам верхнего уровня SCADA-системы – реализация алгоритма «Пуск-Стоп».

**О стандарте IEC 6 1131-3**

Стандарт IEC 6 1131-3 описывает синтаксис и семантику пяти языков программирования ПЛК, - языков, ставших широко известными за более чем 30-летнюю историю их применения в области автоматизации промышленных объектов:

**1. SFC (Sequential Function Chart)** - графический язык, используемый для описания алгоритма в виде набора связанных пар: шаг (step) и переход (transition). Шаг представляет собой набор операций над переменными. Переход - набор условных логических выражений, определяющий передачу управления к следующей паре шаг-переход. По внешнему виду описание на языке SFC напоминает хорошо известные логические блок-схемы алгоритмов. SFC имеет возможность распараллеливания алгоритма. Однако SFC не имеет средств для описания шагов и переходов, которые могут быть выражены только средствами других языков стандарта. Происхождение: Grafcet (Telemechanique-Groupe Schneider).

**2. LD (Ladder Diagram)** - графический язык программирования, являющийся стандартизованным вариантом класса языков релейно-контактных схем. Логические выражения на этом языке описываются в виде реле, которые широко применялись в области автоматизации в 60-х годах. Ввиду своих ограниченных возможностей язык дополнен привнесенными средствами: таймерами, счетчиками и т.п. Происхождение: различные варианты языка релейно-контактных схем (Allen-Bradley, AEG Schneider Automation, GE-Fanuc, Siemens).

Получить выполненную работу или консультацию специалиста по вашему учебному проекту

**3. FBD (Functional Block Diagram)**- графический язык по своей сути похожий на LD. Вместо реле в этом языке используются функциональные блоки, по внешнему виду - микросхемы. Алгоритм работы некоторого устройства на этом языке выглядит как функциональна схема электронного устройства: элементы типа "логическое И", "логическое ИЛИ" и т.п., соединенные линиями. Корни языка выяснить сложно, однако большинство специалистов сходятся во мнении, что это не что иное, как перенос идей языка релейно-контактных схем на другую элементную базу.

**4. ST (Structured Text)** - текстовый высокоуровневый язык общего назначения, по синтаксису ориентированный на Паскаль. Происхождение: Grafcet (Telemechanique-Groupe Schneider).

**5. IL (Instruction List)** - текстовый язык низкого уровня.

Выглядит как типичный язык Ассемблера, что объясняется его

происхождением: для некоторых моделей ПЛК фирмы Siemens является языком Ассемблера. В рамках стандарта IEC 6 1131-3 к архитектуре конкретного процессора не привязан. Происхождение - STEP 5 (Siemens).

Перечисленные языки IEC 6 1131-3 используются ведущими фирмами изготовителями ПЛК, имеют длительную историю применения, достаточно распространены и известны пользователям по тем или иным модификациям. Несмотря на то, что во многих случаях такие модификации несущественны, это влечет определенные неудобства при работе с ПЛК различных фирм-изготовителей. С этой точки зрения, стандарт IEC 6 1131-3, несомненно, прогрессивен, поскольку позволяет привести бесчисленное число различных вариантов и интерпретаций языков ПЛК к единому знаменателю. OpenPCS представлен в виде двух частей: набора средств разработки и исполняемого на целевом ПЛК ядра-интерпретатора. Набор средств разработки исполняется на компьютере проектировщика, например, компьютере типа IBM PC, и состоит из редактора, отладчика и препроцессора, который подготавливает описанный проектировщиком алгоритм к формату, "понятному" ядру-интерпретатору. Этот набор имеет современный пользовательский интерфейс, позволяет тестировать алгоритм в режиме эмуляции и получать листинг алгоритма на языках его описания. После создания, пользовательская программа совместно с ядром-интерпретатором загружается в целевой ПЛК для исполнения. Ядро-интерпретатор, как следует уже из его названия, транслирует пользовательский алгоритм во время исполнения. Это позволяет сконцентрировать машинно-зависимый код и таким образом снизить накладные расходы при переходе на другой ПЛК. Неплохой подход, однако, сразу необходимо отметить, что интерпретационная модель имеет недостаток - она всегда снижает показатели эффективности исполнения программы.

