

Принципы построения кроссплатформенного программно реализованного контроллера электроавтоматики систем ЧПУ высокотехнологичными производственными комплексами

The principles of a cross-platform software-based CNC controller electric automation systems, high-tech industrial complexes

Предложено архитектурное решение для построения программно-реализованного контроллера управления электроавтоматикой современных систем числового программного управления (ЧПУ). Раскрыты механизмы реализации среды проектирования на основе стандарта МЭК 61131 с возможностью реализации программы управления на базе редактора формул. Приведен пример описания работы систем управления электроавтоматикой на базе системы временных булевых функций. Рассмотрены практические аспекты программной реализации системы управления электроавтоматикой станков.

There is suggested an architectural design for constructing software-implemented controller for modern numerical control (NC) systems. There are revealed the mechanisms of the development environment realization based on the IEC 61131 standard with the possibility to create control programs with the help of formula editor. An example of the description of the work of Programmable Logic Control systems based on the system of Timed Boolean Functions is given. Practical aspects of program implementation of machine tools' Programmable Logic Control system are scrutinized.

Ключевые слова: программируемый логический контроллер, Soft PLC, среда программирования МЭК 61131, язык функциональных блоков.

Keywords: programmable logic controller, Soft PLC, development environment IEC 61131, functional block diagrams.

В условиях современного автоматизированного производства наметилась устойчивая тенденция решения логической задачи управления технологическим оборудованием в рамках общего программного обеспечения систем управления без привлечения дополнительной аппаратуры и системного программного обеспечения Программируемых Логических Контроллеров (ПЛК) [1]. Такой подход позволяет снизить стоимость системы управления и получить ряд преимуществ, в том числе: добавление новых функциональных возможностей и модернизация контроллера в короткие сроки; создание кроссплатформенного приложения зависящего от решаемой технологической задачи; возможность сокращения времени запуска в эксплуатацию; сохранение вложенных инвестиций в программное обеспечение и др. Данный подход получил название программно реализованный логический контроллер или Soft PLC [2].

Сегодня на рынке автоматизации, среди существующих программных средств управления технологическим оборудованием, по полноте реализации и масштабности применения выделяют: CoDeSys компании Smart Software Solutions, LabView компании National Instruments, ISaGRAF компании CJ Interna-

tional, SoftCONTROL фирмы SoftingGmbH и др. Однако ряд недостатков не позволяют их применять в кроссплатформенных проектах. В первую очередь это ориентация на конкретную операционную систему (чаще всего Windows), необходимость приобретения лицензии на коммерческое использование; закрытость системы ориентированная на комплексное решение, но без возможности его модернизации и доработки. К тому же ни один из указанных программных пакетов не поставляется с «открытым» программным кодом, что делает затруднительным их использование в качестве базы для программно реализованного ПЛК. В сложившейся ситуации, целесообразно разрабатывать собственный программно реализованный логический контроллер и редактор управляющих программ стандарта МЭК 61131.

Формирование требований к программно-реализованному контроллеру

Развитие компьютерных технологий позволило высвободить вычислительные ресурсы с управления в реальном времени и предоставить конечному пользователю дополнительные возможности (сервисы) в области диагностики оборудования, визуализации объек-

та, управления электроавтоматикой и др.[3,4]. При реализации к Soft PLC контроллеру предъявляются следующие требования:

1. Возможность проектирования и реализации программ в среде МЭК 611-31. [5] Это позволяет повысить скорость и качество разработки программ автоматизации и обеспечить соответствие программно реализованных контроллеров идеологии открытых систем. Применение стандартных языков программирования позволяет исключить этап дополнительного обучения при смене типа логического контроллера.

2. Кроссплатформенность реализации Soft PLC. Программно-реализованный контроллер должен быть ориентирован на работу в рамках различных операционных систем реального времени (Windows RTX, Windows CE, Linux и т.д.). Подход предполагает установку ядра Soft PLC на микроконтроллеры, поддерживающие работу операционной системы, что позволяет не привязываться к конкретному типу микроконтроллера.

3. Встроенный и автономный варианты реализации Soft PLC. Возможность установки Soft PLC как на базе персонального компьютера, например для встраиваемого в ЧПУ решения, так и на аппаратной базе однокристальных микропроцессоров для автономного решения.

4. Поддержка различных промышленных протоколов связи на уровне модулей ввода/вывода [6]. Поддержка стандартных промышленных протоколов обмена данными (CANbus, ModBus, Profibus, Sercos и т.д.) позволит подключать готовые модули ввода/вывода по стандартным физическим каналам связи: RS-232, RS-485, Ethernet и т.д.;

5. Возможность развития и модернизации существующего программного кода Soft PLC [7]. Т.к. ядро Soft PLC – это программой продукт, то его развитие и модернизация происходит постоянно и не требует серьезных капиталовложений, в отличие от аппаратного решения. Применение Soft PLC так же позволяет сократить конечную стоимость системы управления, за счет замены дорогостоящей аппаратной составляющей, на программное решение.

6. Сокращение времени пуско-наладочных работ. Время установки и пуско-наладочных работ программно-реализованного контроллера на технологическом оборудовании уменьшается, за счет применения программных решений требующих лишь инсталляции и настройки.

7. Предоставление механизма открытости для станкостроителя [8]. Применение Soft PLC позволяет создать удобный механизм для предоставления стан-

костроителю следующего функционала: подключение своих свободно программируемых клавиш на панели оператора, вывод информации на экран оператора, подключение специфичных для конкретного станка периферийных устройств.

8. Переносимость проекта с одного типа контроллера на другой. Эта возможность реализуется за счет применения стандарта PLCOpenXML, что позволяет сохранять все созданные программы электроавтоматики в рамках расширения языка разметки XML, ориентированного на работу в рамках среды программирования логических контроллеров.

9. Возможность повторного использования кода ПЛК программы. Это достигается за счет выделения часто применяемых фрагментов управляющих программ в отдельные библиотеки, которые можно подключать к любому проекту и использовать их для работы с однотипным электрооборудованием.

10. Простота языка программирования и разумное ограничение количества базовых элементов библиотеки. Это условие позволяет не предъявлять завышенных требований к разработчикам программ управления.

11. Возможность программирования управляющих программ на базе формульной записи. Это позволяет реализовывать логику работы электроавтоматики не только с помощью языков программирования стандарта МЭК 611-31, но и за счет формирования системы Булевых функций. В ходе работы над проектом, различные его модули могут быть реализованы как на базе языка программирования МЭК 611-31, так и на базе формульной записи.

Виртуальная модель программно-реализованного контроллера

Виртуальная модель программно-реализованного логического контроллера (Рис. 1), в независимости от способа его конкретной реализации, позволяет выделить вертикальные уровни и отношения между ними. На нижнем – аппаратном уровне располагается: стандартный компьютер, который может быть корпусного исполнения (персональный компьютер) или одноплатного исполнения; специальная аппаратура для осуществления ввода/вывода информации с контроллера. К такому виду аппаратного обеспечения относятся баскаптеры (buscouplers), поддерживающие один из стандартных промышленных протоколов. К возможностям такого рода модулей относятся удаленный ввод/вывод как дискретных, так и аналоговых сигналов. Выше расположен системный уровень, содержащий операционную систему реального времени (OS PB) и драйверы подключения аппаратуры..



Рис. 1 Виртуальная модель Soft PLC

На сегодняшний день реализована работа с операционными системами – Windows RTX и Linux, в каждой из которых реализован драйвер, использующий специфические системные ресурсы для подключения аппаратуры. Базовый уровень, содержит ключевые механизмы и объекты ядра, отвечающие за работу машины состояний. Следующим в иерархии реализован коммуникационный уровень, который позволяет связывать ядро с терминальной частью системы управления. На верхнем уровне расположен прикладной уровень, содержащий среду проектирования и разработки

управляющих программ SoftPLC, которая содержит механизмы позволяющие загружать и выполнять пользовательские программы электроавтоматики [9, 10].

Для выделения модулей и отношений между ними была разработана архитектурная модель программно реализованного логического контроллера Рис. 2. В данной модели можно выделить четыре составляющие: среда программирования, ядро SoftPLC, драйверы связи с внешними устройствами и непосредственно сами физические устройства.

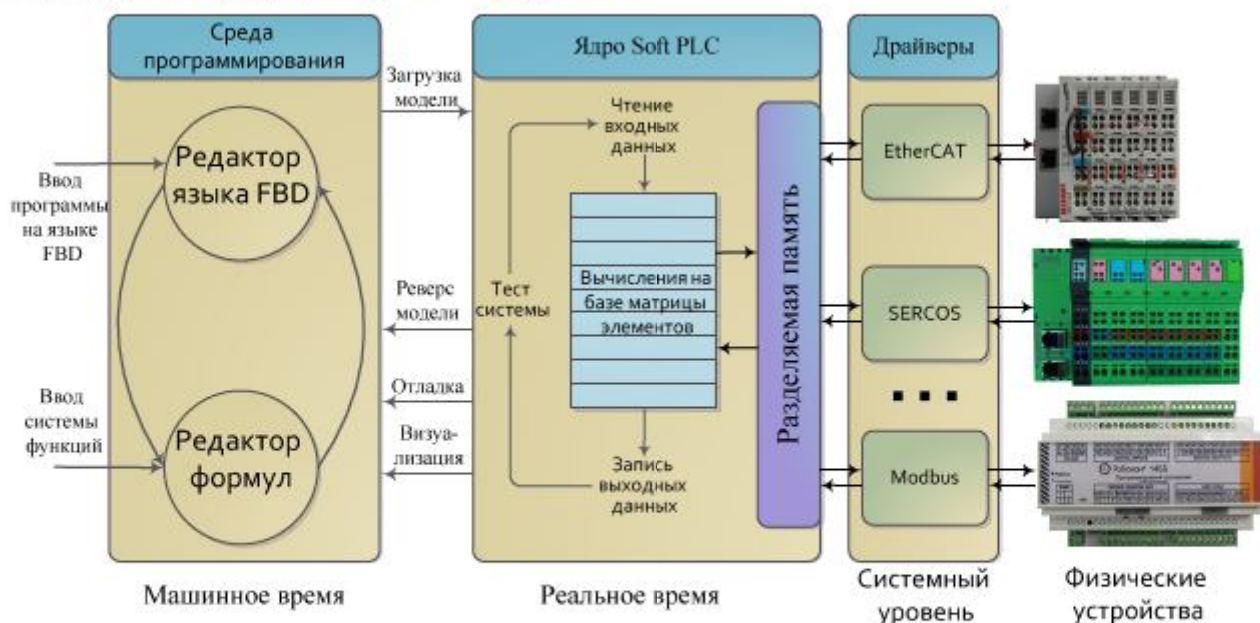


Рис. 2 Архитектура программно реализованного логического контроллера

Soft PLC комплектуется средой разработки, работающей в режиме машинного времени (OC Windows). Особенностью среды разработки является возможность работы как с редактором стандарта МЭК 61131, использующим язык функциональных блоков (FB), так и с редактором формул, который задает логику работы объекта управления посредством системы булевых функций. Среда разработки включает в себя: рабочую область, в которой происходит программирование логики системы управления с помощью задания функциональных блоков и связей между ними; панель инструментов, позволяющую произвести компиляцию, отправку в ядро контроллера и запуск программы на выполнение [11].

При использовании редактора формул необходи-

мо выделить ряд уравнений, которые описывают логику работы объекта управления. На Рис. 4 представлен вариант реализации управляющей программы на базе редактора формул. Логика работы системы управления для моделей изображенных на Рис. 3 и Рис. 4 одинаковая, она предполагает наличие двух цепей: первая цепь реализует операцию «И» для переменных X1 и X2, вторая цепь реализует операцию «ИЛИ» для переменных X2 и X3.

Формирование управляющей программы на базе редактора формул, осуществляется последовательным вводом функции в окне редактирования. После добавления новой функции, появляется окно ввода последующей в котором необходимо добавить поля ввода переменных и базовые операции из меню расположенного слева от окна редактирования [12].

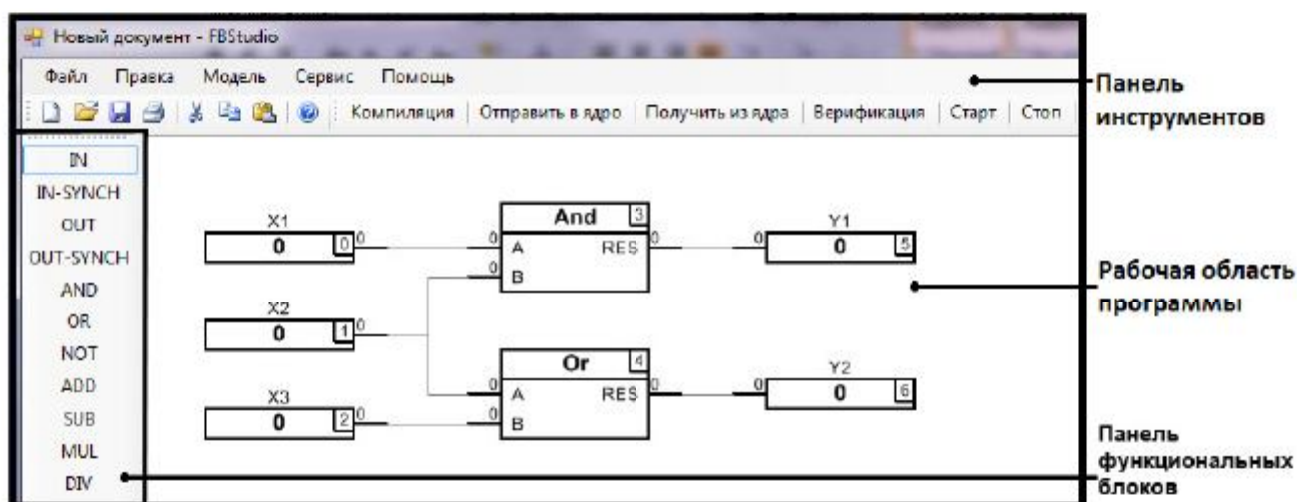


Рис. 3 Среда разработки управляющих программ SoftPLC

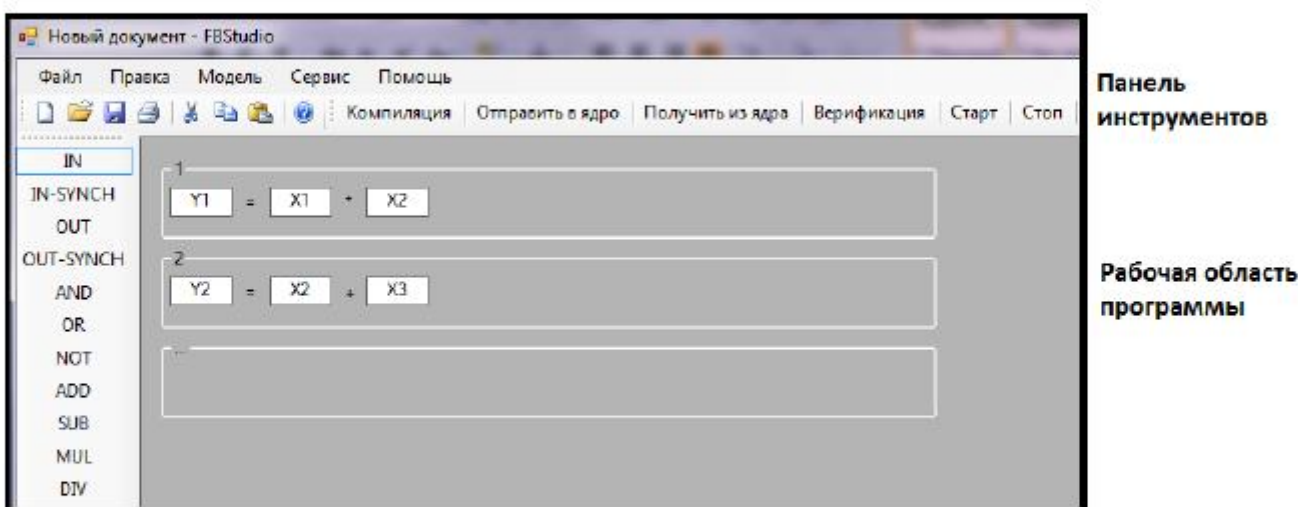


Рис. 4 Редактор формул SoftPLC

Программирование комбинационных схем в среде SoftPLC

Программирование Soft PLC контроллера использует Булеву алгебру, а так же построенные на её основе математические теории. Например, для программирования комбинационной логики без учета временных характеристик можно использовать теорию (n, m)-полюсников. Пусть имеется система *m* функций:

$$\begin{cases} \varphi_1(x_1, x_2, \dots, x_n) = y_1; \\ \varphi_2(x_1, x_2, \dots, x_n) = y_2; \\ \dots \dots \dots \\ \varphi_m(x_1, x_2, \dots, x_n) = y_m. \end{cases} \quad (1)$$

Требуется построить схему, в которой работа *i*-го выхода определялась бы функцией φ_i . Такие схемы, имеющие *n* входов и *m* выходов, называются (n, m)-полюсниками. Функция φ_i определяет некоторый (n,1)- полюсник. Схема для всей системы будет совокупность *m* таких (n,1)-полюсников.

Рассмотрим пример программирования некоторых функций электроавтоматики станка с ЧПУ СА-700, производства ОАО «САСТА»: функции аварийного останова $\varphi_{АВАР}$, функции подачи сигнала включения приводов φ_{ENABLE} и функции готовности гидравлической системы $\varphi_{ГИДР}$. Для программирования данного функционала нам понадобятся следующие сигналы, представленные в таблице 1.

Функция аварийного останова активируется и останавливает станок в тот момент, когда выполняется одно из условий: нажаты конечные выключатели $X_0, X_{MAX}, Y_0, Y_{MAX}$ или не закрыт защитный кожух, т.е. активен сигнал A_{DOOR} . Все эти условия можно записать следующей функцией:

$$\varphi_{АВАР} = (X_0 \vee X_{MAX})(Y_0 \vee Y_{MAX})A_{DOOR} \quad (2)$$

Таблица 1. Сигналы электроавтоматики (n, m)-полюсника

Обозначение сигнала	Описание сигнала
X_0	Аварийный конечный выключатель в начале оси X

Обозначение сигнала	Описание сигнала
X_{MAX}	Аварийный конечный выключатель в конце оси X
Y_0	Аварийный конечный выключатель в начале оси Y
Y_{MAX}	Аварийный конечный выключатель в конце оси Y
A_{DOOR}	Датчик закрытия защитного кожуха
$A_{ПИТ}$	Сигнал наличия питания на приводах
B_X	Сигнал готовности привода оси X
B_Y	Сигнал готовности привода оси Y
$A_{ДАВ}$	Сигнал наличия давления в гидравлической системе

Функция подачи сигнала включения приводов активна тогда, когда имеется сигнал готовности с обоих приводов B_X, B_Y и имеется сигнал наличия питания на приводах $A_{ПИТ}$. Данная функция может быть записана в следующем виде:

$$\varphi_{ENABLE} = A_{ПИТ}B_XB_Y \quad (3)$$

Функции готовности гидравлической системы зависят от двух сигналов: наличия питания $A_{ПИТ}$ и наличия необходимого давления $A_{ДАВ}$.

$$\varphi_{ГИДР} = A_{ПИТ}A_{ДАВ} \quad (4)$$

Объединив данные функции, получим систему следующего вида:

$$\begin{cases} \varphi_{АВАР} = (X_0 \vee X_{MAX})(Y_0 \vee Y_{MAX})A_{DOOR}; \\ \varphi_{ENABLE} = A_{ПИТ}B_XB_Y; \\ \varphi_{ГИДР} = A_{ПИТ}A_{ДАВ}. \end{cases} \quad (5)$$

Построив схему для каждой функции в отдельности в среде программирования Soft PLC, получим совокупность схем, показанную на Рис. 5. Такой синтез не всегда будет оптимальным, при наличии большого количества сигналов необходимо пользоваться методами простых импликант или каскада для синтеза (n, m)-полюсников, широко применяемыми в технике.

Программирование схем с учетом временных характеристик в среде SoftPLC

При программной реализации систем управления со сложной структурой и требующих работы с элементами задержки времени (таймеров) и счетчиками,

применение Булевой алгебры и теории (n,m)-полюсников невозможно. В таких случаях для программирования электроавтоматики применяют временные диаграммы (циклограммы) работы. Рассмотрим пример запуска системы со смазочно-охлаждающей жидкостью (СОЖ) на станке СА-700.

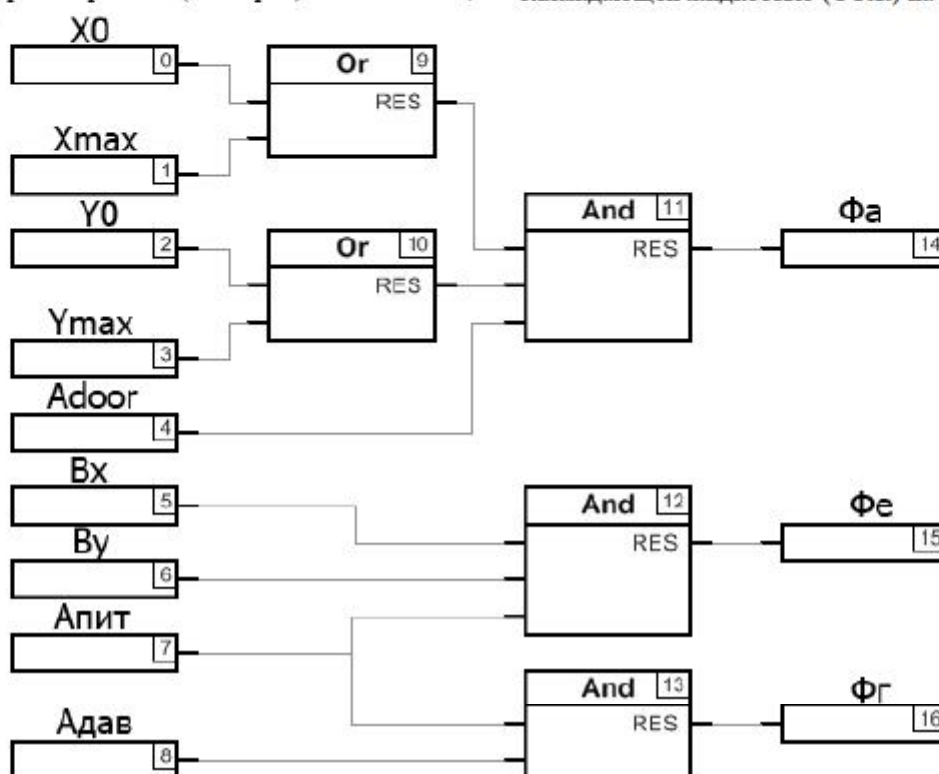


Рис. 5 Схема реализации (n,m)-полюсника в среде SoftPLC

Часть циклограммы работы станка представлена на Рис. 6.

Рассмотрим временную диаграмму работы системы СОЖ (позиция 9). Для написания управляющей программы необходимо определить зависимость функции включения узла СОЖ – $\Phi_{СОЖ}$ от других функций и сигналов, представленных на циклограмме.

Таблица 2 Сигналы электроавтоматики циклограммы работы СОЖ

Обозначение сигнала	Описание сигнала
$X_{СТАРТ}$	Сигнал начала работы
$X_{СТОП}$	Сигнал окончания работы
$\Phi_{СТАНКА}$	Функция работы станка
$X_{ОШИБКА}$	Сигнал возникновения ошибки
$X_{ШПИНДЕЛЬ}$	Сигнал запуска шпинделя
$\Phi_{СОЖ}$	Функция запуска системы СОЖ

Работа станка начинается после нажатия на кнопку СТАРТ – сигнал $X_{СТАРТ}$, при этом активируется реле с самоподхватом, отвечающее за работу станка – функция $\Phi_{СТАНКА}$. Для осуществления формообразования необходимо запустить шпиндель на станке, сигнал $X_{ШПИНДЕЛЬ}$, при активации данного сигнала автоматически должна запуститься система подачи СОЖ. При возникновении ошибки при работе оборудования появляется сигнал $X_{ОШИБКА}$, при этом будет отключено вращение шпинделя и подача СОЖ, после устранения ошибки работа станка продолжится, однако для вращения шпинделя необходимо будет активировать сигнал $X_{ШПИНДЕЛЬ}$. Подача СОЖ прекращается либо при возникновении ошибки, либо через 3 секунды после остановки шпинделя. Задержка в 3 секунды необходима для гарантированного окончания движения формообразования без нагрева

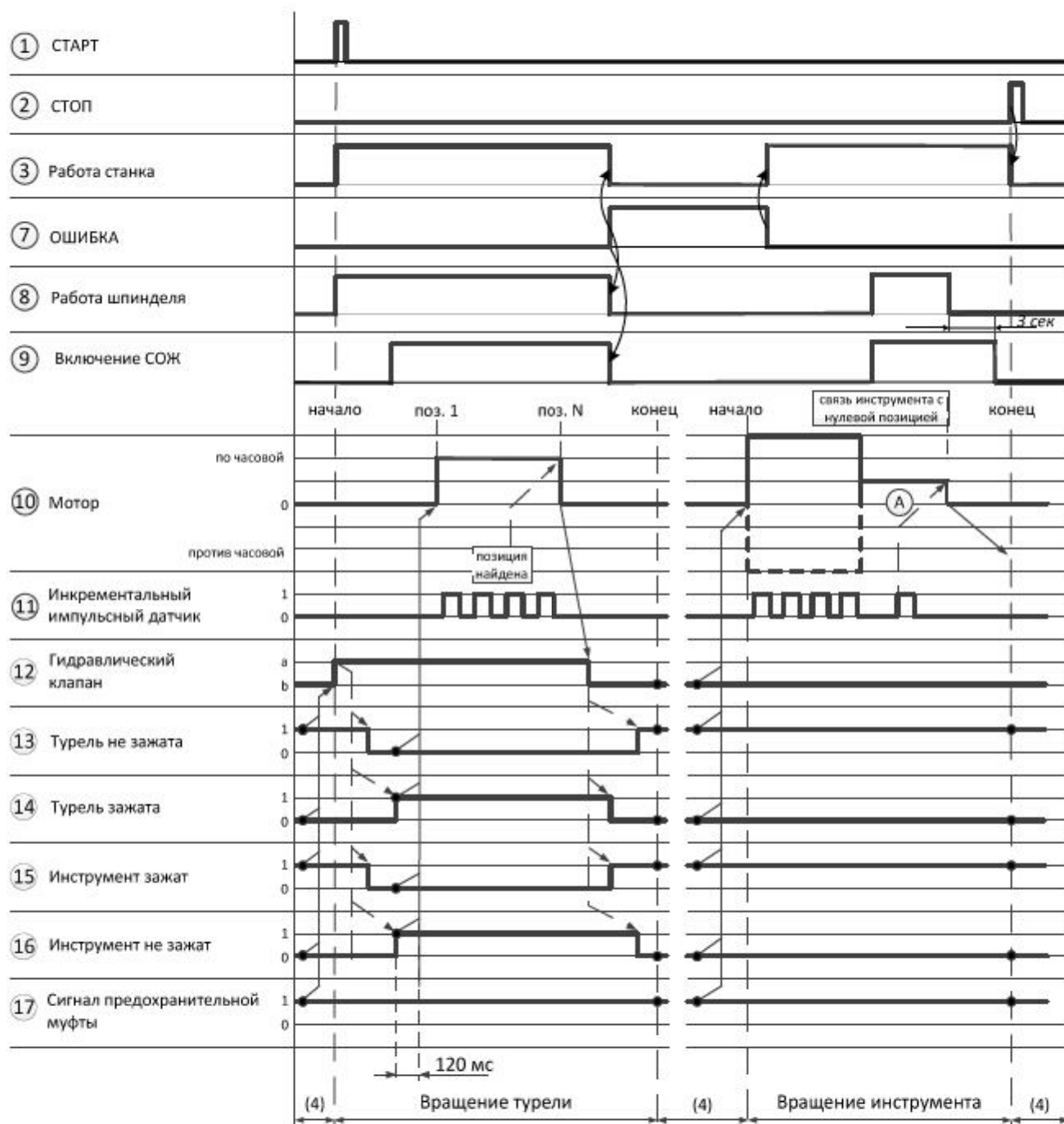


Рис. 6 Циклограмма работы станка СА - 700

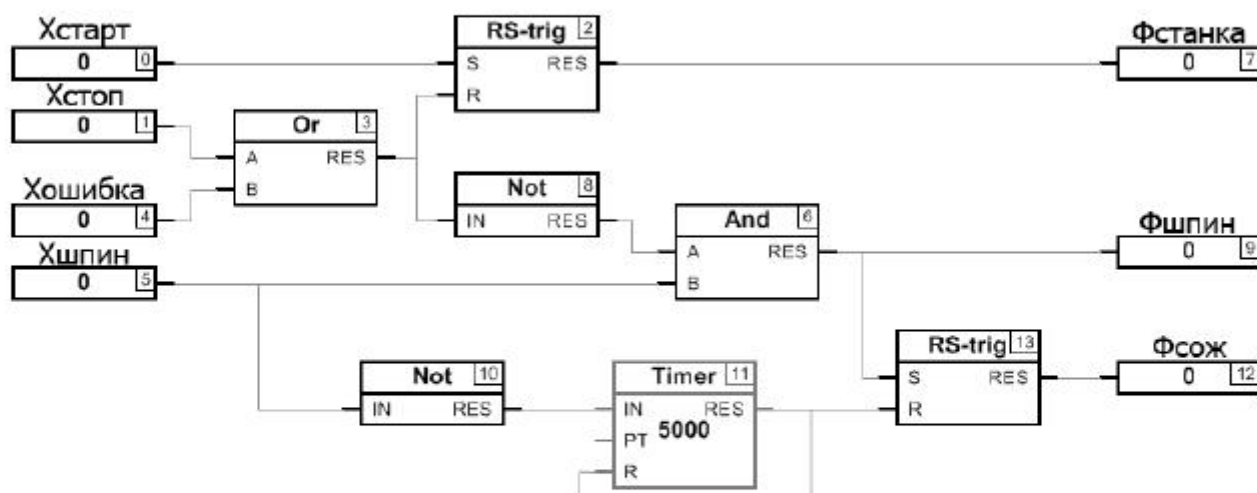


Рис. 7 Схема реализации включения СОЖ в среде SoftPLC

детали или инструмента. Программную реализацию работы циклограммы, выполненная в среде разработки Soft PLC представлена на Рис. 7.

Анализ представленной выше схемы и циклограммы целесообразно разделить на две составляющие: комбинационную и временную. К комбинационной составляющей будут относиться функции работы станка $\Phi_{\text{СТАНКА}}$ и шпинделя $\Phi_{\text{ШПИНДЕЛЯ}}$, т.к. их можно описать комбинационными схемами. Данное описание будет выглядеть в виде системы функций:

$$\begin{cases} \Phi_{\text{СТАНКА}} = (X_{\text{СТАРТ}} \vee \Phi_{\text{СТАНКА}}) \bar{X}_{\text{ОШИБКА}}; \\ \Phi_{\text{ШПИНДЕЛЬ}} = X_{\text{ШПИНДЕЛЬ}} \bar{X}_{\text{ОШИБКА}} \end{cases} \quad (6)$$

Для анализа временной составляющей системы целесообразно применять временные булевы функции. В этом случае используется дискретное время с различным интервалом дискретизации – секунда, минута, час, год и т.д. Единицу дискретного времени обозначим целыми положительными числами – t ($t = 0, 1, 2, \dots$).

Временная булева функция (ВБФ) при этом будет иметь вид:

$$y = f(x_1, x_2, \dots, x_n, t) \quad (7)$$

где x_i – двоичные аргументы; t – аргумент, принимающий значения $0, 1, 2, \dots, l$.

Переменная t рассматривается как дискретное время, и означать моменты времени, в которые про-

исходит смена режимов, переключение, переход в новое состояние и т. д. При фиксации времени t , ВБФ вырождается в статическую булеву функцию. Придавая t ряд последовательных значений от 0 до l , можно получить последовательность булевых функций y_0, y_1, \dots, y_l . Таким образом, каждой ВБФ соответствует определенная последовательность обычных булевых функций.

ВБФ можно описать с помощью таблицы истинности. Число строк в таблице должно быть $2^n \cdot (l+1)$. Табличная запись очень наглядна, однако очень громоздка, поэтому для анализа используем другой способ.

Введем специальную функцию, определенную соотношением:

$$t_\alpha = \begin{cases} 1, & t = \alpha; \\ 0, & t \neq \alpha. \end{cases} \quad (8)$$

Тогда ВБФ можно выразить следующим образом:

$$y = y_0 t_0 \vee y_1 t_1 \vee \dots \vee y_l t_l \quad (9)$$

В любой фиксированный момент времени $t = \alpha$ ($0 \leq \alpha \leq l$) $t_\alpha = 1$, все остальные $t_i = 0$. Функции y_0, y_1, \dots, y_l – обычные функции алгебры логики при $t = 0, t = 1, \dots, t = l$.

При выражении работы системы СОЖ с применением ВБФ получим следующие функции:

В момент времени t_0 функция будет иметь вид

$$y_0 = \varphi_{\text{ШПИНДЕЛЬ}} \overline{X_{\text{ОШИБКА}}}; \quad (10)$$

В момент времени t_1 функция будет иметь постоянное значение 1;

Объединив данные функции, получим ВБФ отвечающую за логику работы системы СОЖ:

$$y = y_0 t_0 \vee y_1 t_1 = \varphi_{\text{ШПИНДЕЛЬ}} \overline{X_{\text{ОШИБКА}}} t_0 \vee t_1 \quad (11)$$

Данная функция является периодической с периодом $q=2$, при этом будет иметь место соотношение $y_t = y_{t+q}$. Если объединить данную функцию с функциями, описывающими логику работы станка и шпинделя, то получим следующую систему описывающую логику работы представленной циклограммы в части включения станка и подачи СОЖ:

$$\begin{cases} \varphi_{\text{СТАНКА}} = (X_{\text{СТАРТ}} \vee \varphi_{\text{СТАНКА}}) \overline{X_{\text{ОШИБКА}}}; \\ \varphi_{\text{ШПИНДЕЛЬ}} = X_{\text{ШПИНДЕЛЬ}} \overline{X_{\text{ОШИБКА}}}; \\ y = \varphi_{\text{ШПИНДЕЛЬ}} \overline{X_{\text{ОШИБКА}}} t_0 \vee t_1. \end{cases} \quad (12)$$

Заключение

Предложенный подход построения кроссплатформенной системы управления электроавтоматикой технологического оборудования применим как для встроенное в систему ЧПУ решение, так и для автономного решения на базе микроконтроллера с операционной системой. Программная реализация контроллеров электроавтоматики предоставляет возможность добавления новых функциональных возможностей и модернизации контроллера в короткие сроки, без существенного изменения архитектуры систем управления. Реализованная в рамках работы среда программирования программно-реализованных контроллеров электроавтоматики соответствует требованиям стандарта МЭК 61131 и дает возможность оператору создавать программы различной степени сложности как на базе стандартных языков программирования, так и на базе редактора формул. Это позволяет применять в рамках одного проекта различные подходы к программированию электроавтоматики.

Библиографический список

1. Sosenkin V.L., Martinov G.M., Perepelkina M.M., The concept: Controlling the cycle logic of NC machine tools with the help of virtual controllers SoftPLC // *Pribory i Sistemy Upravleniya*, 2003, №7, p.5-10.
2. Xi Y., Yang C., Xu J., Study on elevator manufacturing industry with modeling and simulation of virtual elevator control system based on TwinCAT soft-PLC // 2013 3rd International Conference on Industry, Information System and Material Engineering, IISME 2013, p.51-54.
3. Quan Liang, Li Li, The Study of Soft PLC Running System // *Advanced in Control Engineering and Information Science, Procedia Engineering* 15 (2011) 1234 – 1238.
4. J. Borges PC vs PLC for machine and process control // *Real-time Magazine*, 1997;4; 71-72.
5. IEC61131-3 Programmable controller-part3: programming languages, International Electrotechnical Commission, 1993.
6. Мартинова Л.И., Козак Н.В., Нежметдинов Р.А., Пушков Р.Л. Реализация открытости управления электроавтоматикой станков в системе ЧПУ класса PCNC // *Приборы и системы. Управление, контроль, диагностика*. 2011. № 2. С. 11.
7. Мартинов Г.М., Нежметдинов Р.А., Козак Н.В., Пушков Р.Л. Прикладные решения в области управления электроавтоматикой станков с ЧПУ класса PCNC // *Промышленные АСУ и контроллеры*. 2011. № 4. С. 48.
8. Нежметдинов Р.А., Соколов С.В., Обухов А.И., Григорьев А.С. Расширение функциональных возможностей систем ЧПУ для управления механолазерной обработкой // *Автоматизация в промышленности*. 2011. № 5. С. 49-53.
9. Шемелин В.К., Нежметдинов Р.А. Применение технологии клиент-сервер при проектировании контроллера типа Soft PLC для решения логической задачи в рамках систем ЧПУ // *Автоматизация и современные технологии*. 2010. №3. С. 31.
10. Мартинова Л.И., Козак Н.В., Нежметдинов Р.А., Пушков Р.Л., Обухов А.И. Практические аспекты применения отечественной системы ЧПУ «АксиОМАКонтрол» // *Автоматизация в промышленности*. 2012. № 5. С. 36-40.

11. **Мартинов Г.М., Нежметдинов Р.А., Соколов С.В.** Способ построения инструментария систем мониторинга и настройки параметров мехатронного технологического оборудования на основе специализированных программных средств // Мехатроника, автоматизация, управление. 2012. № 7. С. 45-50.

12. **Мартинов Г.М., Нежметдинов Р.А.** Кроссплатформенный программно-реализованный логический контроллер управления электроавтоматикой станков с ЧПУ // Автоматизация и современные технологии. 2013. № 1. С. 15-23.

Мартинов Георги Мартинов – д-р техн. наук, профессор, зав. кафедрой компьютерных систем управления МГТУ «СТАНКИН».

Тел.: 8(499) 972-94-40.

Нежметдинов Рамиль Амирович – канд. техн. наук, доцент кафедры компьютерных систем управления МГТУ «СТАНКИН».

Тел.: 8(499) 972-94-40. E-mail: neramil@ncsystems.ru

Емельянов Алексей Сергеевич – заместитель директора по науке ООО «Центр системной безопасности «ЩИТ-ИНФОРМ», г. Курск.

Martinov Georgi Martinov – D. Sc. in Engineering, head of the sub-department “Computer-architecture Control System” of MSTU “STANKIN”.

Tel.: +7(499) 972-94-40. E-mail: martinov@ncsystems.ru

Nezhmetdinov Ramil Amirovich - Candidate Sc. in Engineering, sub-department “Computer-architecture Control System” of MSTU “STANKIN”.

Tel.: +7(499) 972-94-40. E-mail: neramil@ncsystems.ru

Emel'yanov Aleksey Sergeevich – Deputy Director of Research ООО “Center of system safety “SHIELD-Inform”, Kursk.

