Моделирование и анализ информационных систем. Т. 23, № 2 (2016), с. 173–184
Modeling and Analysis of Information Systems. Vol. 23, No 2 (2016), pp. 173–184

© Рябухин Д. А., Кузьмин Е. В., Соколов В. А., 2016

DOI: 10.18255/1818-1015-2016-2-173-184

УДК 517.9

Построение CFC-программ ПЛК по LTL-спецификации

Рябухин Д. А., Кузьмин Е. В., Соколов В. А.

получена 1 марта 2016

Аннотация. Статья продолжает серию работ, посвященных подходу к построению и верификации «дискретных» программ логических контроллеров (ПЛК) по LTL-спецификации. Этот подход обеспечивает возможность анализа корректности программ логических контроллеров с помощью метода проверки модели (Model Checking). В рамках подхода в качестве языка спецификации программного поведения используется язык темпоральной логики LTL. Анализ корректности LTL-спецификации относительно программных свойств производится автоматически с помощью программного средства символьной проверки модели Cadence SMV.

Ранее было показано, каким образом по корректной (проверенной на корректность относительно программных свойств) LTL-спецификации происходит построение ST-, LD- и IL-программ. В этой статье описывается технология построения CFC-программ по LTL-спецификации. Язык непрерывных функциональных схем CFC (Continuous Function Chart) представляет собой разновидность языка FBD (Function Block Diagram) — графического языка диаграмм принципиальных схем электронных устройств на микросхемах. В отличие от FBD язык CFC обеспечивает возможность свободного размещения программных компонентов и их соединений на экране. Технология построения CFC-программ демонстрируется на примере.

Представление ПЛК-программы на языке CFC в рамках подхода к программированию по LTL-спецификации в отличие от представлений на других стандартных языках дает возможность наглядного графического изображения потока данных от входных переменных к выходам ПЛК. Явным образом демонстрируется зависимость и влияние значений одних переменных на значения других переменных во время исполнения программы за один проход рабочего цикла ПЛК. Фактически CFC-программа представляет собой схему потоков данных ПЛК-программы.

Ключевые слова: программируемые логические контроллеры (ПЛК), построение и верификация ПЛК-программ, LTL-спецификация, CFC-диаграммы

Для цитирования: Рябухин Д. А., Кузьмин Е. В., Соколов В. А., "Построение CFC-программ ПЛК по LTL-спецификации", *Моделирование и анализ информационных систем*, **23**:2 (2016), 173–184.

Об авторах:
Рябухин Дмитрий Александрович, orcid.org/0000-0002-0799-8868, аспирант,
Ярославский государственный университет им. П.Г. Демидова,
ул. Советская, 14, г. Ярославль, 150000 Россия, e-mail: dmitriy_ryabukhin@mail.ru

Кузьмин Егор Владимирович, orcid.org/0000-0003-0500-306X, доктор физ.-мат. наук, доцент, профессор кафедры теоретической информатики, Ярославский государственный университет им. П.Г. Демидова,
ул. Советская, 14, г. Ярославль, 150000 Россия, e-mail: kuzmin@uniyar.ac.ru

Соколов Валерий Анатольевич, orcid.org/0000-0003-1427-4937, доктор физ.-мат. наук, профессор, зав. кафедрой теоретической информатики, Ярославский государственный университет им. П.Г. Демидова,
ул. Советская, 14, г. Ярославль, 150000 Россия, e-mail: sokolov@uniyar.ac.ru

Благодарности:
Работа проводилась при финансовой поддержке РФФИ, грант №12-01-00281-а.

Введение

Статья продолжает серию работ [1–6, 10–13], посвященных подходу к построению и верификации «дискретных» программ логических контроллеров по LTL-спецификации. Этот подход обеспечивает возможность анализа корректности программ логических контроллеров с помощью метода проверки модели (Model Checking) [7, 8]. В рамках подхода в качестве языка спецификации программного поведения используется язык темпоральной логики LTL. Анализ корректности LTL-спецификации относительно программных свойств производится автоматически с помощью программного средства символьной проверки модели Cadence SMV [14].

Программируемый логический контроллер (ПЛК) — «реагирующая» система, имеющая множество входов, подключаемых посредством датчиков к объекту управления, и множество выходов, подключаемых к исполнительным устройствам [18,19]. Программа ПЛК выполняется в рабочем цикле: считывание входов, выполнение программы и выставление выходов. Применение ПЛК в системах управления сложными производственными процессами предъявляет строгие требования корректности к программам ПЛК.

Языки программирования ПЛК определяются стандартом МЭК 61131-3. Стандарт включает в себя описание пяти языков: SFC, IL, ST, LD и FBD/CFC. Ранее в статьях [3–5] показывалось, каким образом по корректной (проверенной на корректность относительно программных свойств) LTL-спецификации происходит построение ST-, LD- и IL-программ. В этой статье описывается технология построения CFC-программ по LTL-спецификации. Язык непрерывных функциональных схем CFC (Continuous Function Chart) представляет собой разновидность языка FBD (Function Block Diagram) — графического языка диаграмм принципиальных схем электронных устройств на микросхемах. В отличие от FBD язык CFC обеспечивает возможность свободного размещения программных компонентов и их соединений на экране. Технология построения CFC-программ демонстрируется на примере программы ПЛК управления установкой для приготовления смесей.

Информация о других подходах к построению и анализу FBD/CFC-программ ПЛК может быть найдена, например, в работах [15–17].

1. Построение программ по LTL-спецификации

Смысл концепции программирования ПЛК по LTL-спецификации [1–6, 10–13] состоит в обеспечении возможности анализа корректности ПЛК-программ методом проверки модели [7, 8]. В соответствии с этой концепцией значение каждой переменной должно изменяться только в одном месте программы и не более одного раза за одно полное выполнение программы при прохождении рабочего цикла ПЛК. Поэтому изменение значения каждой программной переменной задаётся с помощью пары явных LTL-формул и одной неявной. Первая LTL-формула описывает ситуации, при которых происходит возрастание значения соответствующей переменной, вторая LTL-формула определяет условия, приводящие к уменьшению значения переменной. Третья LTL-формула в явном виде не прописывается, поскольку всегда имеет жёсткую конструкцию, составленную из элементов первых двух формул. Она описывает условия, при которых переменная не меняет своего значения за один проход

рабочего цикла ПЛК. Рассматриваемые для спецификации поведения переменных LTL-формулы являются конструктивными в том смысле, что по ним производится построение ПЛК-программы, которая соответствует темпоральным свойствам, выраженным этими формулами. Таким образом, программирование ПЛК сводится к построению LTL-спецификации поведения каждой программной переменной.

Для описания ситуаций, которые приводят к увеличению и уменьшению значения целочисленной переменной V, используются LTL-формулы вида

$$\mathbf{G}\,\mathbf{X}(\, V{>}_V \Rightarrow \mathit{OldValCond} \,\wedge \mathit{FiringCond} \,\wedge V = \mathit{NewValExpr}\,); \tag{1}$$

$$\mathbf{G}\,\mathbf{X}(\, V{<}_V \Rightarrow \mathit{OldValCond}' \wedge \mathit{FiringCond}' \wedge V = \mathit{NewValExpr}'\,). \tag{2}$$

Символ лидирующего подчеркивания «_» в обозначении переменной $_V$ воспринимается как псевдооператор, позволяющий обратиться к значению переменной V, которое она имела в предыдущем состоянии (после последнего полного прохода рабочего цикла ПЛК). При этом псевдооператор может использоваться только под действием темпорального оператора $\mathbf{X}$.

Условия *FiringCond* и *OldValCond* являются логическими выражениями над программными переменными и константами, которые строятся с применением операторов сравнения, логических и арифметических операторов и псевдооператора «_» (который по определению может быть применим только к переменным). Выражение *FiringCond* описывает ситуации, при которых возникает необходимость изменения значения переменной V, если это, конечно, допускается условием *OldValCond*. Выражение *NewValExpr*, определяющее новое значение переменной V, строится с помощью переменных и констант, операторов сравнения, логических, арифметических операторов и псевдооператора «_».

Для описания всех возможных ситуаций, при которых происходит возрастание значения переменной V, в формуле (1) после символа оператора импликации $\Rightarrow$ может потребоваться несколько наборов рассмотренных конъюнктивных членов $\mathit{OldValCond}_i \wedge \mathit{FiringCond}_i \wedge V = \mathit{NewValExpr}_i$ объединенных в дизъюнкцию.

Аналогичный смысл имеют выражения $\mathit{FiringCond}'$, $\mathit{OldValCond}'$ и $\mathit{NewValExpr}'$.

В случае с переменной логического типа данных для спецификации ее поведения используются более простые LTL-формулы:

$$\mathbf{G}\,\mathbf{X}(\, \neg_V \wedge \quad V \Rightarrow \mathit{FiringCond}\,);$$

$$\mathbf{G}\,\mathbf{X}(\quad _V \wedge \neg V \Rightarrow \mathit{FiringCond}'\,);$$

которые означают, что всякий раз, когда новое значение переменной V оказывается больше или меньше ее предыдущего значения, записанного в переменной $_V$, из этого следует, что было выполнено условие соответствующего внешнего воздействия *FiringCond* или $\mathit{FiringCond}'$. В случае наличия равенства $\mathit{FiringCond} = \neg\mathit{FiringCond}'$ эта спецификация может быть заменена на одну LTL-формулу вида

$$\mathbf{G}\,\mathbf{X}(\, V = \mathit{FiringCond}\,).$$

Неявная LTL-формула сохранения прежнего значения переменной V имеет вид

$$\mathbf{G}\,\mathbf{X}(\, V{=}_V \Rightarrow \neg(\mathit{OldValCond} \wedge \mathit{FiringCond}) \wedge \neg(\mathit{OldValCond}' \wedge \mathit{FiringCond}')\,). \tag{3}$$

Для логической переменной V эта неявная LTL-формула выглядит как

$$\mathbf{G}\,\mathbf{X}(\, V{=}_V \Rightarrow \neg(\neg_V \wedge \mathit{FiringCond}) \wedge \neg(_V \wedge \mathit{FiringCond}')\,).$$

При построении спецификации важно учитывать то, в каком порядке распола-

гаются темпоральные формулы, описывающие поведение переменных. Некоторая переменная без псевдооператора «_» может быть задействована в спецификации поведения другой переменной, только если спецификация ее поведения уже произведена и находится выше по тексту.

В спецификации блок темпоральных формул, описывающий поведение некоторой переменной, по необходимости сопровождается указанием начального значения переменной, которое она получает при инициализации. Для этого задействуется ключевое слово Init. Например, $\mathrm{Init}(V) = 1$ означает, что при инициализации переменная V получает значение 1. Если в спецификации явно не указывается начальное значение переменной, то считается, что это значение равняется нулю.

2. Построение CFC-программ ПЛК по LTL-спецификации

В этом разделе рассматривается способ построения программного CFC-кода по конструктивной LTL-спецификации поведения программных переменных. В общем виде схема трансляции LTL-формул в программный код на языке CFC следующая.

Двум явным темпоральным формулам переменной V (1) и (2) (и неявной темпоральной формуле (3)) ставится в соответствие графический блок на языке CFC. Общий вид CFC-блока для целочисленной и логической переменных представлен на рис. 1 в правом нижнем углу. Возможные реализации этого блока также приведены на рис. 1. Порядок выполнения и расположения блоков устанавливается в соответствии с порядком LTL-формул в спецификации. Блоки соединяются между собой по одноименным маркерам виртуальной или реальной линией. Линии проводятся также в соответствии с порядком следования формул в спецификации.

Представление ПЛК-программы на языке CFC в рамках подхода к программированию по LTL-спецификации в отличие от представлений на других стандартных языках дает возможность наглядного графического изображения потока данных от входных переменных к выходам ПЛК. Явным образом демонстрируется зависимость и влияние значений одних переменных на значения других переменных во время исполнения программы за один проход рабочего цикла ПЛК. Фактически CFC-программа представляет собой схему потоков данных ПЛК-программы.

Отметим, что каждая программная переменная должна быть определена в разделе (локальном или глобальном) описания переменных и проинициализирована в соответствии со спецификацией. В среде разработки CoDeSys [9] по умолчанию все переменные инициализируются нулем.

Кроме того, необходимо реализовать идею псевдооператора лидирующего подчеркивания «_». Для этого последними действиями в программе для каждой переменной V выходное значение соответствующего ей CFC-блока направляется также и в переменную-выход $_V$, которая на следующем рабочем цикле ПЛК будет выступать в качестве входной переменной. При этом $_V$ определяют в разделе описания переменных с такой же инициализацией, как и для переменной V.

В следующем разделе идея построения CFC-программ по LTL-спецификации наглядно демонстрируется на примере ПЛК-программы управления установкой для приготовления смесей.

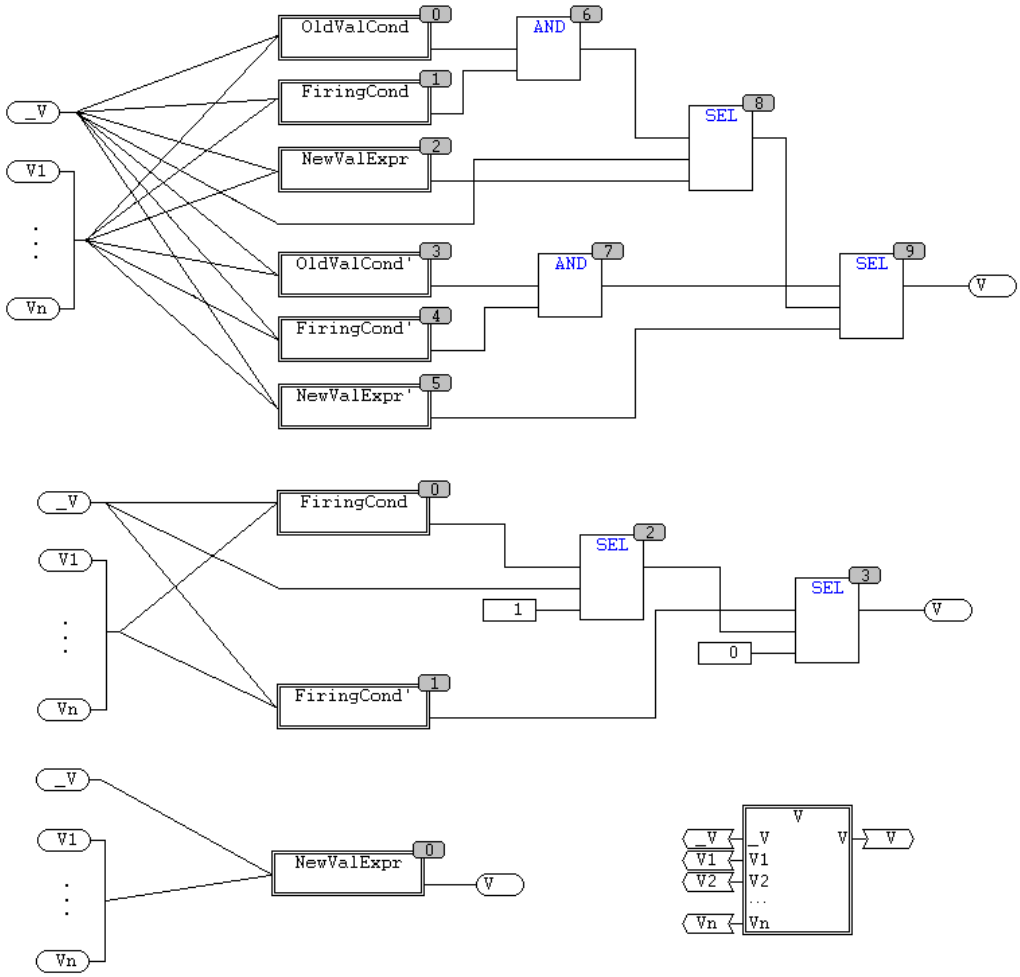

Рис. 1. Общий вид CFC-блока для целочисленной и логической переменных
Fig. 1. General form of CFC-block for integer and boolean variables

3. Установка для приготовления смесей

Установка, схема которой представлена на рис. 2, предназначена для приготовления смесей из двух компонентов. Первый смешиваемый компонент подается из бака 1 посредством клапана «Кл1» в изначально пустой (нет сигнала от датчика «ДУ0») резервуар до тех пор, пока не сработает датчик уровня «ДУ1». По срабатыванию датчика «ДУ1» клапан «Кл1» закрывается. Затем открывается «Кл2» и из бака 2 подается второй смешиваемый компонент до тех пор, пока не сработает датчик уровня «ДУ2», после чего клапан «Кл2» закрывается. Далее в течение T секунд с помощью мешалки в резервуаре происходит перемешивание компонентов при определённой температуре, которая поддерживается нагревателем. По истечении этого времени открывается клапан «РКл» и происходит выгрузка готовой смеси. Выгрузка заканчивается, т. е. клапан «РКл» закрывается, как только пропадает сигнал от датчика уровня «ДУ0». Клапан «АКл» используется для принудительного слива смеси из резервуара. Наличие смешиваемых компонентов в баках определяется по соответствующим датчикам уровня «ДБ1» и «ДБ2». Функционирование мешалки определяется

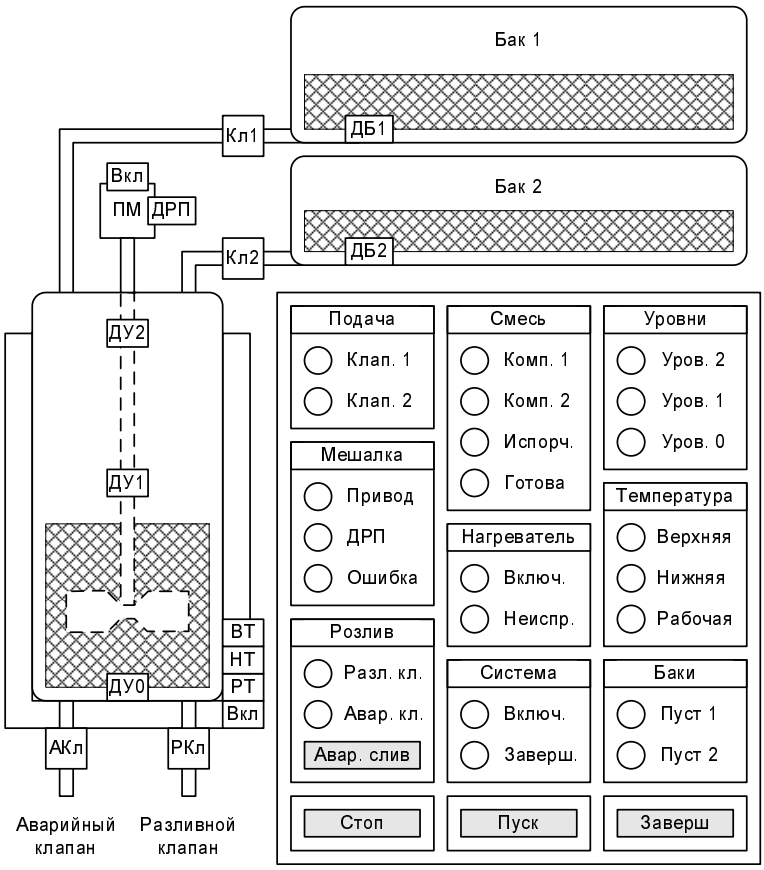

Рис. 2. Схема и панель управления установки для приготовления смесей
Fig. 2. Scheme and control panel of mixing plant

по датчику работы привода мешалки «ДРП». Температура нагревательного элемента отслеживается с помощью датчика верхней температуры «ВТ», датчика нижней температуры «НТ» и датчика рабочей температуры «РТ». Если после запуска установки нагреватель имеет температуру меньше нижней температуры, то он включается и начинает нагреваться. Как только температура нагревателя достигает заданного верхнего значения, то нагреватель выключается и постепенно остывает. Задача нагревателя состоит в поддержании некоторой рабочей температуры, необходимой для смешивания компонентов. Рабочая температура устанавливается на несколько градусов меньше нижней температуры. Запуск, экстренная остановка и плавное завершение работы установки осуществляется соответствующими кнопками панели управления «Пуск», «Стоп» и «Заверш». Под плавным завершением работы понимается выключение установки после того, как будет приготовлена текущая порция смеси и полностью выдана через клапан «РКл».

Установка по приготовлению смесей является автоматической. Управление установкой осуществляется с помощью ПЛК, получающего входные сигналы от датчиков установки и переключателей панели управления оператора и подающего выходные сигналы на приводы установки и лампы панели управления (см. рис. 2).

Задача состоит в написании программы для ПЛК с 13 входами и 14 выходами, предназначенного для управления установкой. Интерфейс ПЛК управления уста-

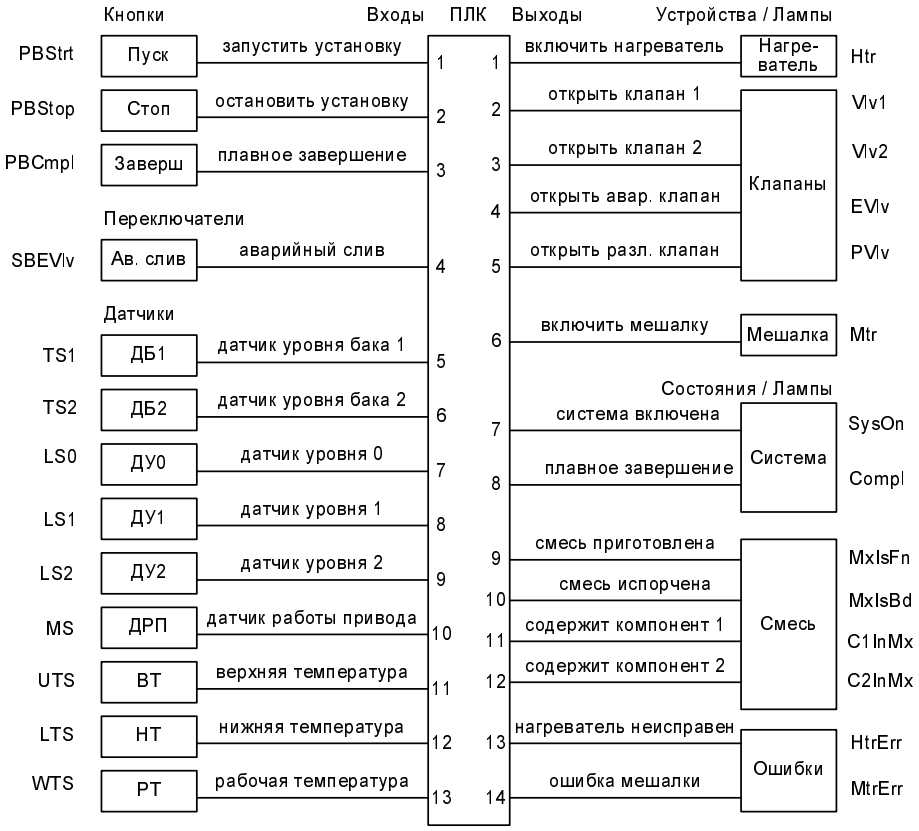

Рис. 3. Интерфейс ПЛК управления установкой для приготовления смесей
Fig. 3. Interface of mixing plant

новкой представлен на рис. 3. Предполагается, что сигналы от датчиков на соответствующие лампы панели управления идут напрямую, минуя ПЛК.

По описанию, схожему с описанием подобной установки из [2,13], была построена следующая LTL-спецификация.

```
/* Таймеры */
GX( MtrTmrIn = Mtr ∧   MS );
GX( ErrTmrIn = Mtr ∧ ¬MS );
GX( HtrTmrIn = Htr ∧ ¬WTS );
/* Состояния/Ошибки/Лампы */
GX( ¬_MtrErr ∧  MtrErr ⇒ (ErrTmrQ ∨ ErrTmrIn) ∧ _MS );
GX(  _MtrErr ∧ ¬MtrErr ⇒ PBStop );
GX( ¬_HtrErr ∧  HtrErr ⇒ (HtrTmrQ ∨ HtrTmrIn) ∧ _WTS );
GX(  _HtrErr ∧ ¬HtrErr ⇒ PBStop );
/* Состояния/Лампы */
GX( ¬_C1InMx ∧  C1InMx ⇒  LS0 ∧ _Vlv1 ∧ _TS1 );
GX(  _C1InMx ∧ ¬C1InMx ⇒ ¬LS0);
GX( ¬_C2InMx ∧  C2InMx ⇒  LS0 ∧ _Vlv2 ∧ _TS2 );
GX(  _C2InMx ∧ ¬C2InMx ⇒ ¬LS0);
GX( ¬_MxIsFn ∧  MxIsFn ⇒  LS0 ∧ MtrTmrQ ∧ ¬MxIsBd );
GX(  _MxIsFn ∧ ¬MxIsFn ⇒ ¬LS0);
```

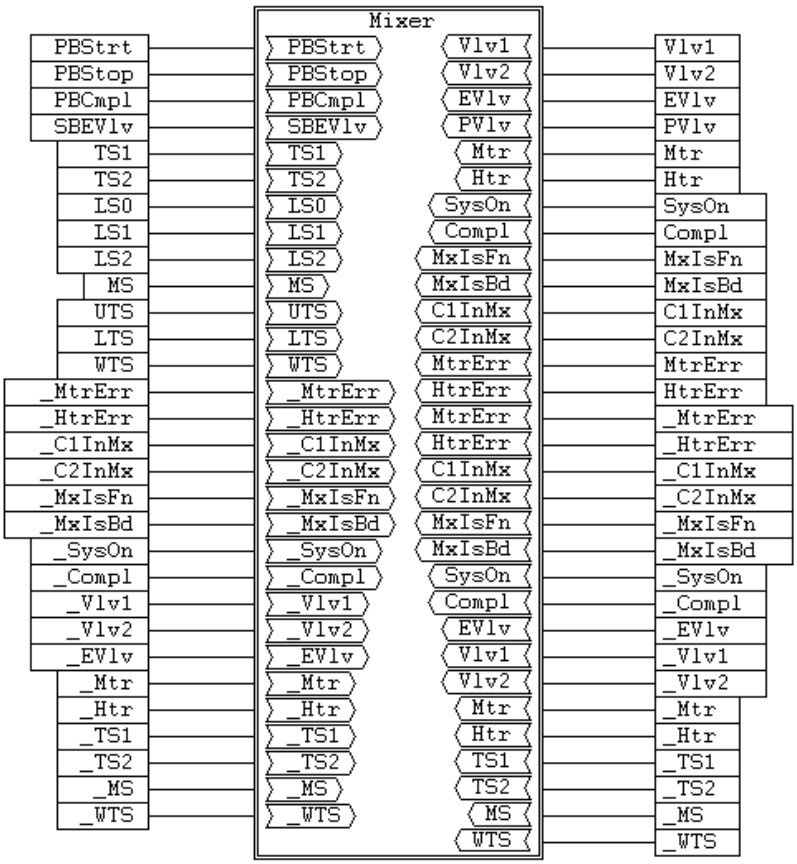

Рис. 4. Общий вид CFC-программы установки для приготовления смесей
Fig. 4. General form of CFC-program of mixing plant

$$\mathbf{GX}(\ \neg_\mathtt{MxIsBd} \wedge \ \mathtt{MxIsBd} \Rightarrow \ \mathtt{LS0} \wedge _\mathtt{EVlv} \wedge \neg_\mathtt{MxIsFn} \wedge _\mathtt{C2InMx});$$

$$\mathbf{GX}(\ \ _\mathtt{MxIsBd} \wedge \neg\mathtt{MxIsBd} \Rightarrow \neg\mathtt{LS0});$$

/* Вспомогательные переменные */

$$\mathbf{GX}(\ \mathtt{Fin} = (\mathtt{PBCmpl} \vee _\mathtt{Compl}) \wedge \neg_\mathtt{Vlv1} \wedge \neg\mathtt{LS0} \vee \mathtt{HtrErr} \vee \mathtt{MtrErr} \vee \mathtt{MxIsBd} \vee$$
$$\vee\, \mathtt{SBEVlv} \vee \mathtt{PBStop} \vee \neg\mathtt{TS1} \wedge \mathtt{LS1} \wedge \neg\mathtt{MxIsFn} \vee \neg\mathtt{TS2} \wedge \mathtt{LS2} \wedge \neg\mathtt{MxIsFn}\ );$$

/* Состояния/Лампы */

$$\mathbf{GX}(\ \neg_\mathtt{SysOn} \wedge \ \mathtt{SysOn} \Rightarrow \ \mathtt{PBStrt} \wedge \neg\mathtt{Fin})\ );$$

$$\mathbf{GX}(\ \ _\mathtt{SysOn} \wedge \neg\mathtt{SysOn} \Rightarrow \ \mathtt{Fin}\ );$$

$$\mathbf{GX}(\ \neg_\mathtt{Compl} \wedge \ \mathtt{Compl} \Rightarrow \ \mathtt{SysOn} \wedge \mathtt{PBCmpl}\ );$$

$$\mathbf{GX}(\ \ _\mathtt{Compl} \wedge \neg\mathtt{Compl} \Rightarrow \neg\mathtt{SysOn}\ );$$

/* Приводы/Лампы */

$$\mathbf{GX}(\ \mathtt{EVlv} = \mathtt{SBEVlv}\ );$$

$$\mathbf{GX}(\ \mathtt{Vlv1} = \mathtt{SysOn} \wedge \mathtt{TS1} \wedge \neg\mathtt{LS1} \wedge \neg\mathtt{C2InMx} \wedge \neg\mathtt{Compl} \wedge \neg\mathtt{LS0} \wedge \mathtt{WTS}\ );$$

$$\mathbf{GX}(\ \mathtt{Vlv2} = \mathtt{SysOn} \wedge \mathtt{TS2} \wedge \ \mathtt{LS1} \wedge \neg\mathtt{LS2} \wedge \neg\mathtt{MxIsFn} \wedge \mathtt{WTS}\ );$$

$$\mathbf{GX}(\ \mathtt{Mtr} \ = \mathtt{SysOn} \wedge \mathtt{LS2} \wedge \neg\mathtt{MxIsFn} \wedge \mathtt{WTS} \wedge \mathtt{C1InMx} \wedge \mathtt{C2InMx}\ );$$

$$\mathbf{GX}(\ \mathtt{PVlv} = \mathtt{SysOn} \wedge \mathtt{MxIsFn}\ );$$

/* Нагреватель/Лампа */

$$\mathbf{GX}(\ \neg_\mathtt{Htr} \wedge \ \mathtt{Htr} \Rightarrow \ \mathtt{SysOn} \wedge \neg\mathtt{LTS}\ );$$

$$\mathbf{GX}(\ \ _\mathtt{Htr} \wedge \neg\mathtt{Htr} \Rightarrow \neg\mathtt{SysOn} \vee \ \mathtt{UTS}\ ).$$

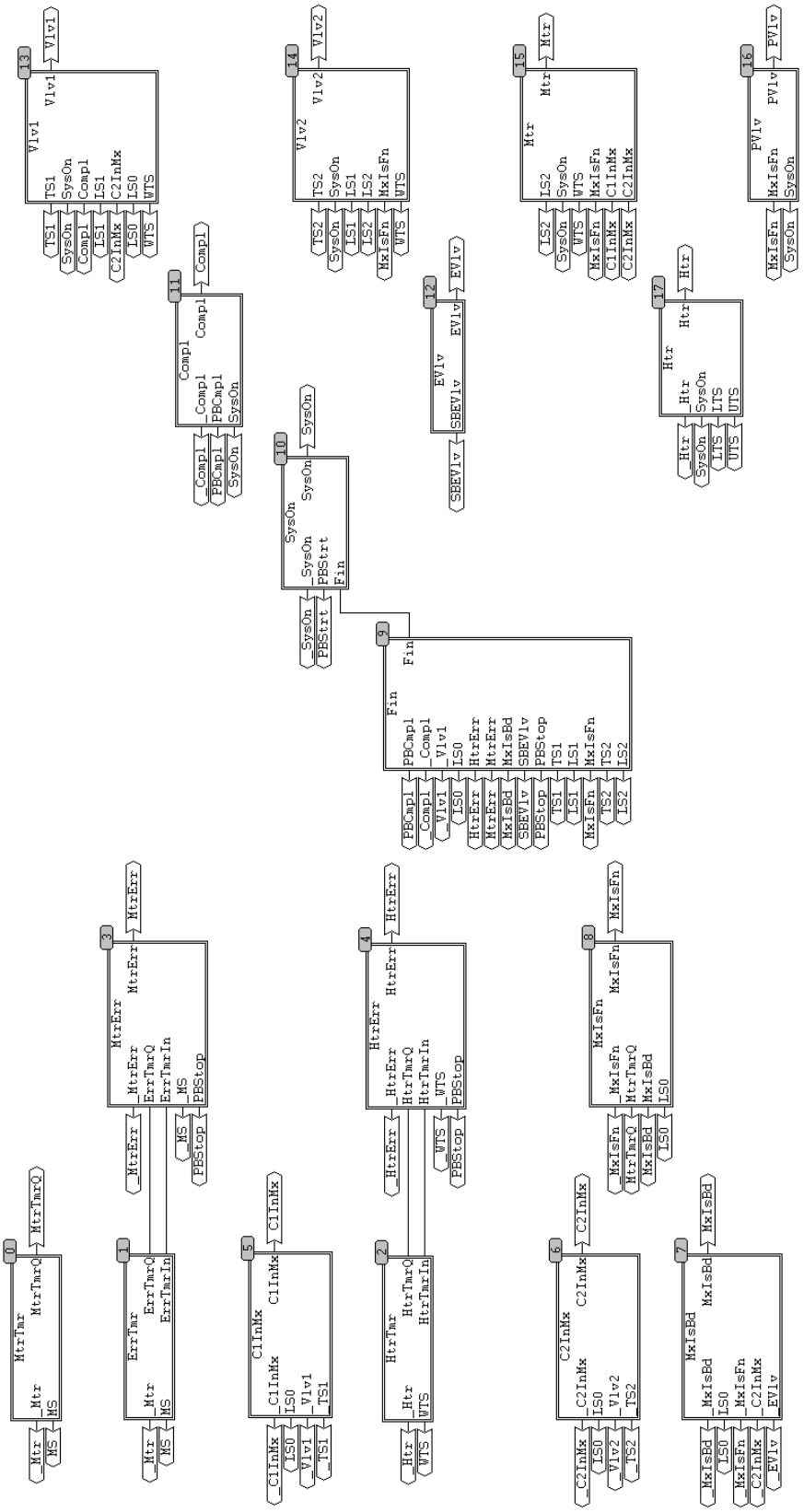

Рис. 5. CFC-программа установки для приготовления смесей
Fig. 5. CFC-program of mixing plant

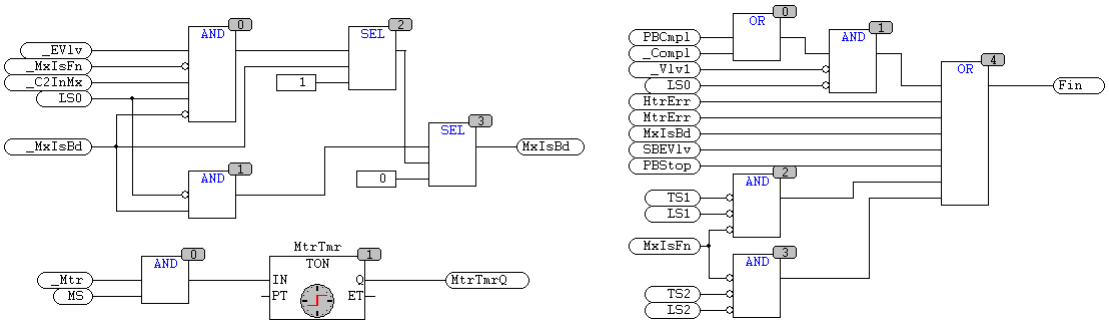

Рис. 6. Примеры CFC-блоков для переменных MxIsBd, MtrTmrQ и Fin
Fig. 6. CFC-blocks for variables MxIsBd, MtrTmrQ and Fin

Примерами программных свойств, которые предполагаются к проверке для данной LTL-спецификации (и соответственно для CFC-программы, построенной по этой спецификации), являются следующие LTL-свойства.

«Выключенная система». Если система выключена, то клапаны подачи и разливной клапан закрыты, нагреватель и привод мешалки выключены, а лампы группы «Система» погашены: $\mathbf{G}(\ \neg\texttt{SysOn} \Rightarrow \neg(\texttt{Vlv1} \vee \texttt{Vlv2} \vee \texttt{PVlv} \vee \texttt{Htr} \vee \texttt{Mtr} \vee \texttt{Compl})\ )$.

«Клапаны». В один момент времени в открытом состоянии может находиться не более одного клапана: $\mathbf{G}(\ \texttt{Vlv1} + \texttt{Vlv2} + \texttt{PVlv} + \texttt{EVlv} \leq 1\ )$.

«Пустые баки». Если хотя бы один из баков и резервуар пустые, система находится в выключенном состоянии: $\mathbf{G}(\ (\neg\texttt{TS1} \vee \neg\texttt{TS2}) \wedge \neg\texttt{LS0} \Rightarrow \neg\texttt{SysOn}\ )$.

«Мешалка». Если привод мешалки включен, все клапаны установки закрыты: $\mathbf{G}(\ \texttt{Mtr} \Rightarrow \neg(\texttt{Vlv1} \vee \texttt{Vlv2} \vee \texttt{PVlv} \vee \texttt{EVlv})\ )$.

«Перерасход компонентов». В испорченную или уже приготовленную смесь подача компонентов не осуществляется: $\mathbf{G}(\ (\texttt{MxIsBd} \vee \texttt{MxIsFn}) \Rightarrow \neg(\texttt{Vlv1} \vee \texttt{Vlv2})\ )$.

«Испорченная смесь». Испорченная смесь не может ни выдаваться через разливной клапан установки, ни перемешиваться мешалкой, ни подогреваться включенным нагревателем: $\mathbf{G}(\ \texttt{MxIsBd} \Rightarrow \neg\texttt{PVlv} \wedge \neg\texttt{Mtr} \wedge \neg\texttt{Htr}\ )$.

«Включенная система». Если система включена и в резервуаре достигнута рабочая температура, то обязательно открыт один из неаварийных клапанов или работает привод мешалки: $\mathbf{G}(\ \texttt{SysOn} \wedge \texttt{WTS} \Rightarrow (\texttt{Vlv1} \vee \texttt{Vlv2} \vee \texttt{PVlv} \vee \texttt{Mtr})\ )$.

«Включенный привод». Если привод мешалки был включен, то он обязательно рано или поздно будет выключен: $\mathbf{G}(\ \texttt{Mtr} \Rightarrow \mathbf{F}(\neg\texttt{Mtr})\ )$.

По описанной в этом разделе LTL-спецификации установки для приготовления смесей строится CFC-программа, которая в общем виде представлена на рис. 4 и 5. Примеры реализаций CFC-блоков для переменных MxIsBd, MtrTmrQ и Fin приведены на рис. 6. CFC-блоки программы на рис. 5 в основном соединены неявно с помощью соответствующих маркеров, но для примера показаны и несколько явных линий связи. Визуально блоки расположены таким образом, что значения переменных-блоков следующего ряда не оказывают влияния на значения переменных-блоков предыдущего ряда в рамках потока данных при исполнении программы за один рабочий цикл ПЛК.

Список литературы / References

[1] Кузьмин Е. В., Рябухин Д. А., Соколов В. А., "О выразительности подхода к построению ПЛК-программ по LTL-спецификации", *Моделирование и анализ информационных систем*, **22**:4 (2015), 507–520; [Kuzmin E. V., Ryabukhin D. A., Sokolov V. A., "On the Expressiveness of the Approach to Constructing PLC-programs by LTL-Specification", *Modeling and Analysis of Information Systems*, **22**:4 (2015), 507–520, (in Russian).]

[2] Кузьмин Е. В., Рябухин Д. А., Соколов В. А., "Моделирование согласованного поведения ПЛК-датчиков", *Моделирование и анализ информационных систем*, **21**:4 (2014), 75–90; [Kuzmin E. V., Ryabukhin D. A., Sokolov V. A., "Modeling a Consistent Behavior of PLC-Sensors", *Modeling and Analysis of Information Systems*, **21**:4 (2014), 75–90, (in Russian).]

[3] Рябухин Д. А., Кузьмин Е. В., Соколов В. А., "Построение IL-программ ПЛК по LTL-спецификации", *Моделиров. и анализ информационных систем*, **21**:2 (2014), 26–38; [Ryabukhin D. A., Kuzmin E. V., Sokolov V. A., "Construction of PLC IL-programs by LTL-specification", *Modeling and Analysis of Information Systems*, **21**:2 (2014), 26–38, (in Russian).]

[4] Кузьмин Е. В., Соколов В. А., Рябухин Д. А., "Построение и верификация LD-программ ПЛК по LTL-спецификации", *Моделирование и анализ информационных систем*, **20**:6 (2013), 78–94; [Kuzmin E. V., Sokolov V. A., Ryabukhin D. A., "Construction and Verification of PLC LD-programs by LTL-specification", *Modeling and Analysis of Information Systems*, **20**:6 (2013), 78–94, (in Russian).]

[5] Кузьмин Е. В., Соколов В. А., Рябухин Д. А., "Построение и верификация ПЛК-программ по LTL-спецификации", *Моделирование и анализ информационных систем*, **20**:4 (2013), 5–22; [Kuzmin E. V., Sokolov V. A., Ryabukhin D. A., "Construction and Verification of PLC-programs by LTL-specification", *Modeling and Analysis of Information Systems*, **20**:4 (2013), 5–22, (in Russian).]

[6] Кузьмин Е. В., Соколов В. А., "Моделирование, спецификация и построение программ логических контроллеров", *Моделирование и анализ информационных систем*, **20**:2 (2013), 104–120; [Kuzmin E. V., Sokolov V. A., "Modeling, Specification and Construction of PLC-programs", *Modeling and Analysis of Information Systems*, **20**:2 (2013), 104–120, (in Russian).]

[7] Baier C., Katoen J.-P., *Principles of Model Checking*, The MIT Press, 2008.

[8] Clark E. M., Grumberg O., Peled D. A., *Model Checking*, The MIT Press, 2001.

[9] CoDeSys, *Controller Development System*, http://www.3s-software.com/.

[10] Kuzmin E. V., Sokolov V. A., Ryabukhin D. A., "Construction and Verification of PLC-Programs by LTL-Specification", *Automatic Control and Computer Sciences*, **49**:7 (2015), 453–465.

[11] Kuzmin E. V., Sokolov V. A., Ryabukhin D. A., "Construction and Verification of PLC LD Programs by the LTL Specification", *Automatic Control and Computer Sciences*, **48**:7 (2014), 424–436.

[12] Kuzmin E. V., Sokolov V. A., "Modeling, Specification and Construction of PLC-programs", *Automatic Control and Computer Sciences*, **48**:7 (2014), 554–563.

[13] Kuzmin E. V., Ryabukhin D. A., Sokolov V. A., "Modeling a Consistent Behavior of PLC-Sensors", *Automatic Control and Computer Sciences*, **48**:7 (2014), 602–614.

[14] SMV, *The Cadence SMV Model Checker*, http://www.kenmcmil.com/smv.html.

[15] Wardana A., *Development of Automatic Program Verification for Continious Function Chart based on Model Checking*, Kassel university press GmbH.

[16] Ovataman T., Aral A., Polat D., Unver A. O., "An overview of model checking practices on verification of PLC software", *Software and Systems Modeling*, 2014.

[17] Pakonen A., Matasniemi T., Lahtinen J., Karhela T., "A toolset for model checking of PLC software", *Proceedings of 18th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA2013, Cagliari, Italy*, 2013.

[18] Parr E. A., *Programmable Controllers. An engineer's guide*, Newnes, 2003.

[19] Петров И. В., *Программируемые контроллеры. Стандартные языки и приемы прикладного проектирования*, М.: СОЛОН-Пресс, 2004; [Petrov I. V., *Programmiruemye kontrollery. Standartnye jazyki i priemy prikladnogo proektirovanija*, M.: SOLON-Press, 2004, (in Russian).]

Ryabukhin D. A., Kuzmin E. V., Sokolov V. A., "Construction of CFC-programs by LTL-specification", *Modeling and Analysis of Information Systems*, **23**:2 (2016), 173–184.

DOI: 10.18255/1818-1015-2016-2-173-184

Abstract. This article continues a cycle of papers, which describe an approach to construction and verification of discrete PLC-programs by an LTL-specification. The approach provides a possibility of PLC-program correctness analysis by the model checking method. For the specification of the program behavior the linear-time temporal logic LTL is used. The correctness analysis of an LTL specification is performed automatically by the symbolic model checking tool Cadence SMV.

Previously it was shown how ST-, LD- and IL-programs are constructed by a correct (with verified program properties) LTL-specification. In this article a technology of CFC-program construction by an LTL-specification is described. The language CFC (Continuous Function Chart) is a variation of FBD (Function Block Diagram). FBD is a graphical language for microcircuits. CFC provides a possibility of free allocation of program components and connections on a screen. The approach to construction of CFC-programs is shown by an example.

PLC-program representation on CFC within the approach to programming by LTL-specification differs from other representations. It gives the visualisation of a data flow from inputs to outputs. Influence and dependence between variables is explicitly shown during program execution within one PLC working cycle. In fact, CFC-program is a scheme of PLC-program data flow.

Keywords: programmable logic controllers (PLC), construction and verification of PLC-programs, LTL-specification, CFC-diagrams

On the authors:
Ryabukhin Dmitriy Aleksandrovich, orcid.org/0000-0002-0799-8868, graduate student,
P.G. Demidov Yaroslavl State University,
Sovetskaya str., 14, Yaroslavl, 150000, Russia, e-mail: dmitriy_ryabukhin@mail.ru

Kuzmin Egor Vladimirovich, orcid.org/0000-0003-0500-306X, doctor of science, associate professor,
P.G. Demidov Yaroslavl State University,
Sovetskaya str., 14, Yaroslavl, 150000, Russia, e-mail: kuzmin@uniyar.ac.ru

Sokolov Valery Anatolievich, orcid.org/0000-0003-1427-4937, doctor of science, professor,
P.G. Demidov Yaroslavl State University,
Sovetskaya str., 14, Yaroslavl, 150000, Russia, e-mail: sokolov@uniyar.ac.ru

Acknowledgments:
This work was supported by the Russian Foundation for Basic Research (RFBR), №12-01-00281-a.