# Формальный дедуктивный анализ автоматного алгоритма управления генератором эндогаза с помощью платформы Rodin

## Часть 2. Алгоритм управления и платформа Rodin

**Максим Нейзов** (neyzov.max@gmail.com)

**Формальный дедуктивный анализ представляет собой строгий математический подход к верификации алгоритмов: алгоритм описывается с помощью аксиом, а требуемые свойства доказываются как теоремы. Цель представленного анализа – доказать соответствие алгоритма управления предъявляемым требованиям надёжности и безопасности. В статье представлен алгоритм управления в виде системы взаимосвязанных автоматов и платформа Rodin – инструмент для формального анализа систем и автоматизации доказательства теорем.**

### Введение: формальные аксиоматические теории

В первой части статьи [1] были определены требования надёжности и безопасности технологического процесса. Для гарантии соответствия алгоритма данным требованиям необходим его формальный анализ, который будет состоять в построении формальной аксиоматической теории.

Аксиоматический способ построения теории был известен ещё до нашей эры и применён Евклидом в геометрии. Аксиоматические теории – это теории, в основе которых лежит набор аксиом и правил вывода. *Аксиомы* – элементарные утверждения, принимаемые без доказательств. *Теоремы* – утверждения, имеющие доказательства. *Доказательство* – цепочка логических рассуждений, которая строится с помощью правил вывода, аксиом и уже доказанных теорем [2]. Таким образом, теорема – это логическое следствие из аксиом. Метод получения нового знания является *дедуктивным*. Теория интересна, прежде всего, как множество теорем. Полученные знания достоверны, если доказательства теорем не содержат ошибок. Для исключения ошибок строятся *формальные теории*. Формализация теории – это её построение с помощью строгого математического аппарата. Все объекты и действия в формальной теории строго фиксированы. Формальное доказательство представляет собой легко проверяемую (в том числе и компьютерной программой) цепочку формул.

Для гарантии соблюдения заданных требований выполняется *формальный дедуктивный анализ* алгоритма управления. Строится формальная аксиоматическая теория алгоритма. Алгоритм представляется с помощью набора аксиом. Теоремы представляют свойства данного алгоритма. Доказанное свойство является атрибутом алгоритма по построению, так как оно логически следует из него. Если теорема доказана, то алгоритм гарантированно обладает данным свойством. Если алгоритм не обладает данным свойством, то свойство не является теоремой данной теории, следовательно, не существует цепочки рассуждений для доказательства. После доказательства наличия заданного свойства у алгоритма нет смысла в его тестировании с целью поиска контрпримеров, нарушающих это свойство. Так же бессмысленно, как после доказательства теоремы Пифагора пытаться найти прямоугольный треугольник, нарушающий равенство: $a^2 + b^2 = c^2$, где a, b – катеты, c – гипотенуза.

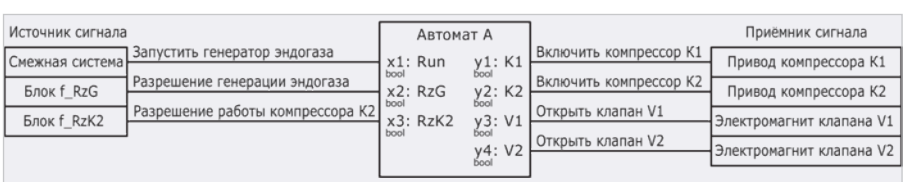


**Рис. 1. Интерфейс автомата A**

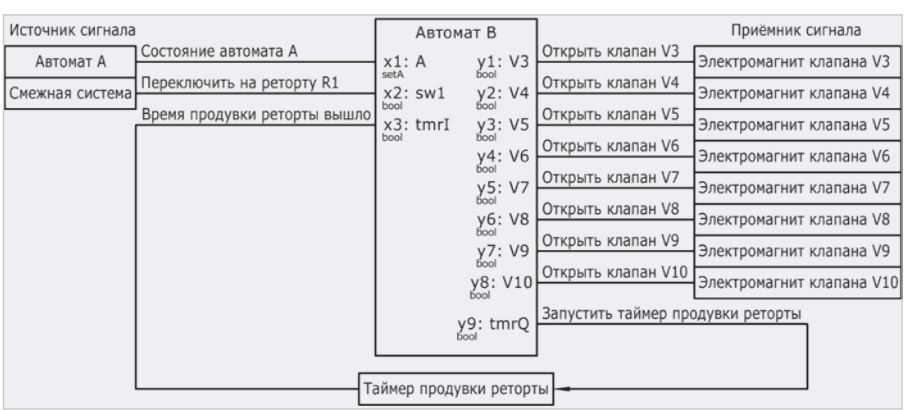


**Рис. 2. Интерфейс автомата B**

### Платформа Rodin

Платформа Rodin [3] – инструментальное программное обеспечение, предназначенное для поддержки метода Event-B [4–6]. Event-B – формальный метод моделирования и анализа систем. Платформа Rodin имеет множество плагинов и позволяет редактировать модели, запускать на исполнение для валидации и тестирования, выполнять проверку модели (model checking), доказывать простые теоремы автоматически и теоремы любой сложности в интерактивном режиме.

Для моделирования в Event-B используются теория множеств и логика предикатов [5]. Язык множеств является универсальным: любые математиче-

ские утверждения можно переформулировать на языке теории множеств [2]. Одной из главных концепций метода Event-B является *уточнение* [3–6]. Сложные системы строятся как последовательность уточнений. Каждое уточнение добавляет некоторые детали и конкретизирует систему. Таким образом, осуществляется последовательное движение от абстрактного к конкретному. В Event-B существует два вида компонентов: *контекст* и *машина* [3–6]. Контекст задаёт статическую часть модели, машина – динамическую. Аксиомы задаются в контексте. Машине доступна вся информация из указанного контекста. Машина содержит переменные, отвечающие за текущее состояние, и *события*, действия которые изменяют значения переменных. События могут иметь *параметры* и *охранные условия*. Событие может произойти, только если его параметры удовлетворяют охранным условиям. Если охранные условия позволяют произойти сразу нескольким событиям, то случайным образом выбирается одно из них. Таким образом, построенная модель определяет *недетерминированную дискретную систему переходов* [3, 5]. В машине объявляются *инварианты* – утверждения, истинные в любом её состоянии. Инварианты включаются в список *обязательств по доказательству* [3] и должны быть доказаны, как и теоремы. Теоремы могут быть объявлены как в контексте, так и в машине. При доказательстве теорем в интерактивном режиме могут применяться различные *тактики*. Возможно добавление произвольных *гипотез*, которые становятся *леммами* после их доказательства.

## Автоматный подход к разработке алгоритмов

Автоматное программирование [7] – концепция разработки алгоритмов, управляющая часть которых проектируется как *система взаимосвязанных автоматов* (СВА) [8]. Алгоритмы управления чаще всего состоят только из управляющей части и представляют собой СВА. *Автомат* является преобразователем входных последовательностей сигналов в выходные [2, 9]. Выходной сигнал зависит от входного сигнала и состояния автомата, хронящего предысторию всех входных сигналов. Автоматный подход позволяет избежать ряда ошибок и способствует построению более надёжных и безопасных алгоритмов управления [10]. Автоматный подход к разработке алгоритмов получил широкое распространение [7, 8, 10–16]. Автоматы удобно задавать в виде *графов переходов*.

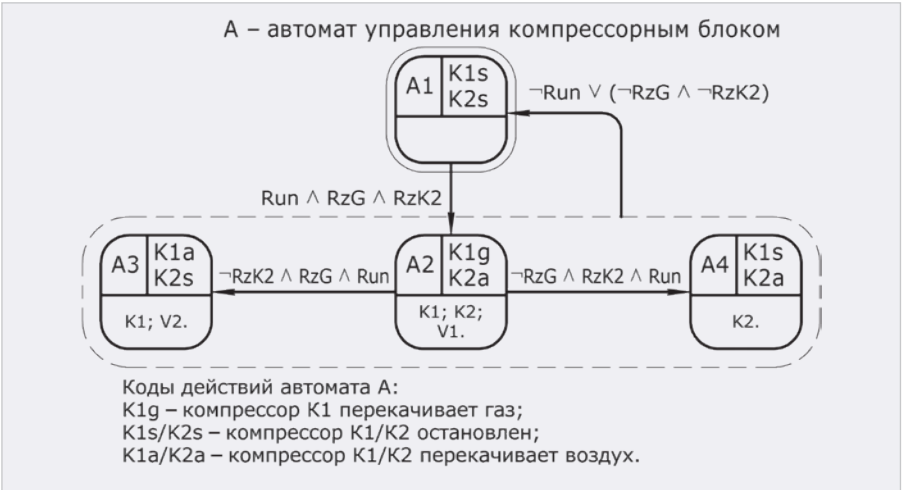


**Рис. 3. Граф переходов автомата A**

Для формального дедуктивного анализа алгоритма необходимо его представление в строгой математической форме. Автоматный подход проектирования в данном случае является очень удобным, так как автомат уже является строгим математическим объектом. Не требуется никаких дополнительных преобразований.

## Автоматный алгоритм управления

Алгоритм управления генератором эндогаза представлен как СВА, состоящая из двух автоматов – A и B. Автомат A управляет компрессорным блоком, автомат B – клапанами реторт. Интерфейсы автоматов представлены на рисунках 1 и 2. Автомат A имеет три входа и четыре выхода, а автомат B – три входа и девять выходов. Блок f_RzG устанавливает сигнал, если выполняется условие: ¬P1min ∧ ¬P1max ∧ water ∧ ¬Fail_K1. Блок f_RzK2 устанавливает сигнал, если выполняется условие: ¬Fail_K2. Если заданное условие не выполняется, то блок сбрасывает выходной сигнал. Для повторной установки сигнала требуется перезапуск блока передним фронтом сигнала Run. Автомат B использует текущее состояние автомата A для управления. Автомат B взаимодействует с таймером для выдержки заданного интервала времени.

Графы переходов автоматов являются полной и точной спецификацией их поведения (представлены на рисунках 3 и 4). A и B – автоматы Мура [2, 7–9]. Состояния A1 и B1 – начальные. В левом верхнем углу вершин графа указаны состояния автомата, в правом верхнем углу – коды действий автомата, внизу – установленные сигналы. Таким образом, в состоянии A4 компрессор K1 остановлен, а компрессор K2 перекачивает воздух, для этого установлен один выходной сигнал K2, остальные сигналы сброшены. Если условия на всех дугах, исходящих из состояния, ложны, то автомат остаётся в прежнем состоянии. Если в состояние можно перейти из других состояний по одному и тому же условию, то происходит объединение в группу (обрамляется пунктиром) и указывается один переход из данной группы. Таким образом, из состояний A2...A4 можно перейти в состояние A1 по указанному условию.

При наличии сигнала Run и разрешений RzG и RzK2 автомат A переходит из состояния A1 в состояние A2. При этом включаются компрессоры K1, K2 и открывается клапан V1. Переход автомата A в состояние A2 приводит к переходу автомата B из состояния B1 в состояние B4 (при наличии сигнала sw1). Состоянию B4 соответствует продувка реторты R1 газом, а также реторты R2 воздухом. При этом открываются клапаны V3, V6, V9, V10 и устанавливается сигнал tmrQ для запуска таймера продувки реторты. После окончания времени продувки таймер устанавливает сигнал tmrI и автомат B переходит в состояние B7. Состоянию B7 соответствует работа реторты R1 и продувка реторты R2 воздухом. При этом клапан V9 закрывается, а клапан V7 открывается, остальные клапаны остаются в прежнем положении.

При исчезновении разрешения генерации RzG автомат A перехо-

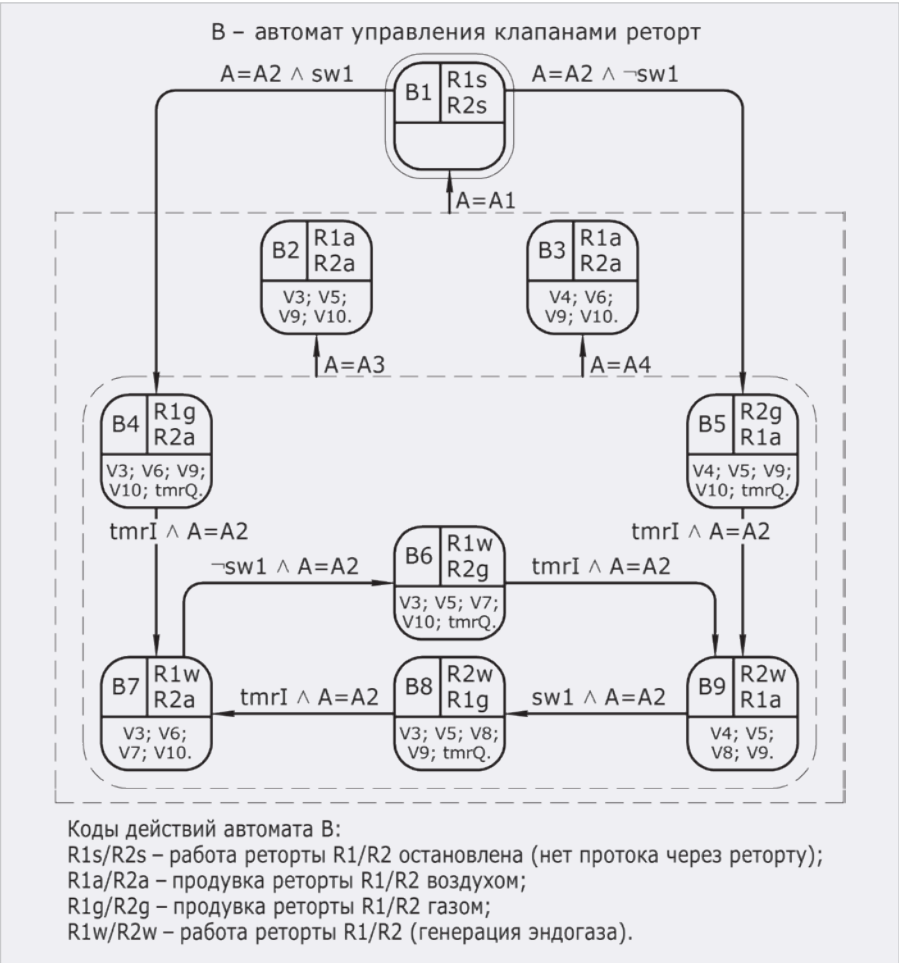


**Рис. 4. Граф переходов автомата B**

дит из состояния A2 в состояние A4. При этом включённым остаётся компрессор K2. Переход автомата A в состояние A4 приводит к переходу автомата B из состояния B7 в состояние B3, которому соответствует продувка реторт R1/R2 воздухом. При этом открыты клапаны V4, V6, V9, V10.

Снятие сигнала Run приводит к переходу автомата A в начальное состояние A1. При этом автомат B, находясь в любом состоянии, также переходит в начальное состояние B1. В начальных состояниях всё оборудование отключено.

## Математическое и программное обеспечение

Рассмотренный алгоритм в виде СВА представляет математическое обеспечение. Программное обеспечение является формальной и изоморфной [10] трансляцией математического обеспечения (графов переходов). Трансляция может быть ручной или автоматической при наличии транслятора на целевой язык программирования. Также платформа Rodin поддерживает автоматическую трансляцию моделей Event-B на языки C, C++, C#, Java [17].

Для создания сверхнадёжного программного обеспечения написание кода вручную противопоказано. Во избежание ошибок нужно использовать автоматическую кодогенерацию по исходной модели. Это даёт гарантию эквивалентности поведения программы и её модели.

## Заключение

В статье рассмотрены автоматный алгоритм управления и платформа Rodin. Алгоритм управления был создан в виде СВА и поэтому является строгим математическим объектом, удобным для формального анализа. Для верификации алгоритма требуется представить СВА с помощью аксиом и доказать ряд теорем, соответствующих предъявляемым требованиям надёжности и безопасности. Для этого будет использоваться платформа Rodin.

## Литература


1. *Нейзов М.* Формальный дедуктивный анализ автоматного алгоритма управления генератором эндогаза с помощью платформы Rodin. Часть 1. Определение требований надёжности и безопасности работы генератора эндогаза. Современная электроника. 2020. № 9.
2. *Кузнецов О. П., Адельсон-Вельский Г. М.* Дискретная математика для инженера. 2-е изд., перераб. и доп. Энергоатомиздат. М. 1988. С. 480.
3. URL: www3.hhu.de/stups/handbook/rodin.
4. URL: wiki.event-b.org.
5. *Abrial J. R.* Modelling in Event-B: System and Software Engineering. Cambridge Univ. Press. 2010.
6. *Robinson K.* System Modelling & Design. Using Event-B. Draft book. 2012. P. 142.
7. *Поликарпова Н. И., Шалыто А. А.* Автоматное программирование. 2008. С. 167.
8. *Шалыто А. А.* Switch-технология. Алгоритмизация и программирование задач логического управления. Наука. СПб. 1998. С. 628.
9. *Карпов Ю. Г.* Теория автоматов: учебник для вузов. – 1-е изд. Издат. дом ПИТЕР. СПб. 2003. С. 208.
10. *Шалыто А. А.* Автоматное проектирование программ. Алгоритмизация и программирование задач логического управления. Известия РАН. Теория и системы управления. 2000. № 6. С. 63–81.
11. *Зюбин В. Е.* Процесс-ориентированное программирование: учеб. пособие. Новосибирск. Новосиб. гос. ун-т. 2011. С. 194.
12. *Шелехов В. И.* Разработка автоматных программ на базе определения требований. ИСИ СО РАН. Новосибирск. Системная информатика. 2014. № 4. С. 1–29.
13. *Любченко В. С.* К проблеме создания модели параллельных вычислений. Параллельные вычисления и задачи управления: труды третьей международной конференции. Москва. 2006. С. 1359–1374.
14. *Harel D.* et al. STATEMATE: A working environment for the development of complex reactive systems. IEEE Trans. Eng. 1990. № 4. P. 403–414.
15. *Wagner F., Schmuki R., Wagner T., Wolstenholme P.* Modeling software with finite state machines: a practical approach. Auerbach Publications. 2006. P. 390.
16. *Samek M.* Practical UML Statecharts in C/C++. Second Edition: Event-Driven Programming for Embedded Systems. Newnes. 2008.
17. *Mery D., Singh N. K.* Automatic code generation from Event-B models. In Proceedings of the Second Symposium on Information and Communication Technology. SoICT. ACM. 2011.