УДК 004.3.02 DOI: 10.17587/mau.19.770-776

А. Д. Иванников, д-р техн. наук, гл. науч. сотр., e-mail: adi@ippm.ru,
Институт проблем проектирования в микроэлектронике РАН

Составление множества функций при формировании отладочных тестов для цифровых систем управления объектами[1]

Рассматриваются управляющие цифровые системы, функционирование которых может быть представлено в виде выполнения последовательности функций из конечного алфавита. Для отладки проектов таких систем методом моделирования необходимо формирование некоторого набора тестовых воздействий на моделируемую систему для проверки правильности ее функционирования. Данная работа посвящена формированию тестовых наборов для проверки правильности выполнения последовательности функций. Показано, что на множестве допустимых последовательностей функций определена частичная полугруппа. Множество слов в конечном алфавите функций может быть задано некоторой праволинейной грамматикой. Допустимые последовательности формализуются путем введения графа функций, задающего возможные для выполнения функции для различных состояний цифровой системы. Если допустимость последовательного выполнения двух функций зависит от выполненных ранее функций и состояния цифровой системы, то некоторые функции должны быть разделены на подфункции. Рассматривается методика этого процесса. Предлагается метод составления множества функций цифровой системы, при котором возможно представление допустимых последовательностей функций в виде графа. Сформированный граф функций совместно с множествами входных взаимодействий для каждой функции цифровой системы задают спецификацию внешнего поведения цифровой системы. Предлагаемый метод проиллюстрирован на примере формирования множества функций цифровой системы управления чертежным автоматом. Предлагается метод формирования набора тестов на основе графа функций.

Ключевые слова: *отладка методом моделирования, цифровые системы, алфавит функций, графовое представление последовательности функций*

Введение

При проектировании цифровых систем управления объектами для отладки проектов широко используется метод моделирования [1—4]. На компьютерную модель цифровой системы подаются некоторые входные воздействия, а реакция модели проектируемой системы проверяется на соответствие техническому заданию. В связи с тем, что выполнение какой-либо функции может быть инициировано не только входным сигналом цифровой системы управления, но и самой цифровой системой, в качестве аргументов функционирования цифровых систем могут рассматриваться входные взаимодействия — последовательности сигналов, включающие как входные сигналы цифровой системы управления, так и ее выходные сигналы управления обменом [5]. При отладке методом моделирования важной задачей является выбор конечного числа конечных по времени тестовых входных взаимодействий (тестовых примеров).

В работе [5] показано, что функционирование цифровых систем управления объектами может быть представлено как последовательность выполняемых функций, каждая из которых задается подачей на цифровую систему некоторого входного взаимодействия. Иными словами, каждая цифровая система выполняет некоторую последовательность функций из конечного алфавита **K**, причем выполнение каждой функции вызывается одним из входных взаимодействий определенного класса. Входные взаимодействия, задающие выполнение цифровой системой функции $k \in \mathbf{K}$, могут различаться значениями данных, временными соотношениями между отдельными сигналами в допустимых пределах, могут иметь и другие различия.

Отладочные тестовые примеры должны осуществлять проверку как правильности выполнения цифровой системой всех ее функций из множества **K**, так и правильности выполнения допустимых последовательностей функций $f \in \mathbf{F}$, где **F** — множество допустимых последовательностей функций из множества **K**. Алгоритм формирования минимального полного набора отладочных тестов для проверки правильности выполнения функций множества **K** предложен и проанализирован в работе [4]. Задачей данной работы является разработка метода формирования отладочного набора тестов для проверки правильности выполнения допустимых последовательностей функций.

[1] Работа выполнена при поддержке гранта РФФИ № 17-07-00683.

Формальное представление множества последовательностей выполняемых функций

Рассматривая последовательное выполнение функций k_i и k_j как операцию умножения $\cdot$, а последовательность k_ik_j — как произведение, можно сказать, что на множестве последовательностей выполняемых функций **F** определена полугруппа, в общем случае частичная.

Рассмотрим полугруппу <**F**, $\cdot$>, в общем случае частичную, с конечным множеством **K** порождающих элементов k.

При этом возможны три случая.

1. Произведение $f_i \cdot f_j$ определено для всех $f_i \in \mathbf{F}$, $f_j \in \mathbf{F}$. В этом случае полугруппа <**F**, $\cdot$> является полной.

2. Произведение $f_i \cdot f_j$ определено, если f_i и f_j могут быть представлены как $f_i = k_{i_1} \cdot k_{i_2} \cdot \ldots \cdot k'$, $f_j = k'' \cdot k_{j_2} \cdot \ldots \cdot k_{j_l}$, причем произведение $k' \cdot k''$ существует. В этом случае вопрос о существовании $f_i \cdot f_j$ сводится к вопросу о существовании произведений $k' \cdot k''$, где $k' \in \mathbf{K}$, $k'' \in \mathbf{K}$.

3. Произведение $f_i \cdot f_j$ определено, если существуют произведения $k_{i_1} \cdot k_{i_2} \cdot \ldots \cdot k' \cdot k''$, $k_{i_1} \cdot k_{i_2} \cdot \ldots \cdot k' \cdot k'' \cdot k_{j_2}$, ..., $k_{i_1} \cdot k_{i_2} \cdot \ldots \cdot k' \cdot k'' \cdot k_{j_2} \cdot \ldots \cdot k_{j_l}$. В этом случае вопрос о существовании $f_i \cdot f_j$ сводится к вопросу о существовании произведения $f \cdot k$, где $f \in \mathbf{F}$, $k \in \mathbf{K}$.

Третий случай является наиболее общим, в связи с чем проведем его анализ.

Проектируемые цифровые системы всегда имеют конечное множество внутренних состояний. В связи с этим все существующие произведения могут быть заданы конечным числом правил.

Множество **F** есть множество слов в конечном алфавите **K**, которое может быть задано некоторой праволинейной грамматикой [6] с конечным числом правил вывода. Множество допустимых слов **F** может быть задано конечным инициальным автоматом без выходов

$$\mathrm{A} = (\mathbf{K}, \mathbf{X}, x_{\text{н}}, \varphi), \qquad (1)$$

где **K** — множество входных символов; **X** — множество состояний автомата А; $x_{\text{н}}$ — начальное состояние, $x_{\text{н}} \in \mathbf{X}$; φ: $\mathbf{K} \times \mathbf{X} \to \mathbf{X}$ — частичное переходное отображение.

Слова множества **F** могут заканчиваться любым символом из **K**. В связи с этим любое состояние из **X** может являться заключительным. Каждому состоянию x, $x \in \mathbf{X}$, соответствует подмножество $\mathbf{F}^x$, $\mathbf{F}^x \subset \mathbf{F}$, слов, заканчивающихся в состоянии x.

Автомат А задает все возможные последовательности функций, выполняемые цифровой системой, т. е. множество допустимых слов **F**. Будем называть автомат А автоматом функций. Графическое задание автомата А в виде диаграммы переходов является наиболее наглядным и простым для использования разработчиками цифровой аппаратуры. Спецификации на входные взаимодействия цифровой системы могут определяться заданием множества входных взаимодействий, соответствующих каждой функции k, $k \in \mathbf{K}$ [7], и автомата функций А.

Анализ последовательностей выполняемых функций на примере цифровой системы управления чертежным автоматом

Проиллюстрируем вышесказанное на примере цифровой системы управления несложным чертежным автоматом, рассмотренным в работе [4]. Чертежный автомат включает плоский планшет, на котором осуществляется вычерчивание, два шаговых двигателя, которые управляют движением головки, в держатели которой устанавливается два пера. Головка может перемещаться параллельно плоскости планшета в поднятом положении или с одним опущенным пером (собственно вычерчивание). Имеются концевые выключатели, которые выдают некоторый сигнал при достижении каким-либо пером крайнего положения. Предусмотрены также кнопки "Ввод" и "Стоп", нажимаемые оператором при начале сеанса вычерчивания и необходимости его останова, и кнопки "П1", "П2" ("Перо 1" и "Перо 2"), "↑", "↓", "→", "←" для ручного опускания и перемещения вычерчивающих головок, кнопка "– –", при нажатии которой вычерчивается штриховая линия. Предусмотрена также клавиатура, на которой набирается расстояние между установленными в головке перьями по двум координатам. Имеется также цифровая система управления, которая после нажатия кнопки "Ввод" получает команды от компьютера, выдает последовательности сигналов на шаговые двигатели и осуществляет подъем и опускание перьев. Компьютер по каналу связи передает на устройство управления чертежным автоматом последовательность кадров, которые описывают выполняемый чертеж. Каждый кадр является командой для перемещения одной из вычерчивающих головок в опущенном или

поднятом положениях по отрезку прямой или дуге окружности. При перемещении в опущенном положении головка осуществляет вычерчивание сплошной, штриховой или штрихпунктирной линии. Система управления должна также реагировать на сигналы концевых выключателей и нажатия управляющих кнопок.

Алфавит функций **K**, выполняемых такой системой, приведен в табл. 1.

Рассмотрим всевозможные пары k_i, k_j с точки зрения существования их произведений (табл. 2). При составлении табл. 2 сделана попытка задать существование произведений $k_i \cdot k_j$ независимо от предыдущих взаимодействий. Такая попытка опирается на опыт проектирования, который показывает, что для многих цифровых систем или блоков это справедливо. Кроме того, при надежном проектировании системы должна быть предусмотрена ее реакция на любую последовательность входных взаимодействий. В случае последовательной

Таблица 1

Перечень функций, выполняемых цифровой системой управления чертежным автоматом

Table 1. List of functions for drawing automata digital control system

Обозначение	Функция
k_1	Ввод и отработка кадра без нажатия кнопки "Стоп" и срабатывания концевых выключателей
k_2	Ввод и отработка кадра без нажатия кнопки "Стоп" со срабатыванием одного или одновременно двух концевых выключателей
k_3	Нажатие кнопки "Ввод" и ожидание отличных от нуля входных данных
k_4	Нажатие кнопок "↑", "↓", "→", "←" и движение головки по горизонтали, вертикали или с наклоном 45 ° при поднятых или одном опущенном перьях (кнопки "П1" или "П2"); вычерчивание штриховой линии (кнопка "– –"; просто опускание и подъем пера при нажатии кнопок "П1", "П2"
k_5	Нажатие кнопки "Стоп" во время ввода и отработки кадра, в том числе оставшейся части кадра, окончание ввода кадра, если происходит его ввод, останов отработки кадра и ввода новых кадров
k_6	Нажатие кнопки "Пуск" и возобновление отработки прерванного кадра
k_7	Ввод данных, выявление символа "Начало чертежа"
k_8	Ввод команды "Конец чертежа" и возврат цифровой системы в исходное состояние
k_9	Изменение сигналов "Поправка *X*", "Поправка *Y*"
k_{10}	Отработка оставшейся (в том числе пустой) части кадра после нажатия кнопки "Пуск" без срабатывания концевых выключателей
k_{11}	Отработка оставшейся (в том числе пустой) части кадра после нажатия кнопки "Пуск" со срабатыванием одного или одновременно двух концевых выключателей
k_{12}	Нажатие кнопки "Сброс" во время выполнения функций k_1, k_2, k_3, k_5 (во время ввода кадра), k_7, k_{10}, k_{11} или в промежутке между выполнениями любых функций

Таблица 2

Возможные произведения $k_i \cdot k_j$ для цифровой системы управления чертежным автоматом

Table 2. Permissible products $k_i \cdot k_j$ for drawing automata digital control system

Первый элемент пары	Возможные вторые элементы пары	Условия существования произведения
k_1	k_1	Нет
	k_2	Нет
	k_5	Нет
	k_8	Нет
	k_{12}	Нет
k_2	k_4	Движение головки обеспечивает отвод последней от края чертежа
	k_9	Нет
	k_{12}	Нет
k_3	k_7	Нет
	k_{12}	Нет
k_4	k_3	Ранее или после последней k_8 или k_{12} не было k_2, k_5 или k_{11}
	k_4	Нет
	k_6	Ранее было k_2 и k_4, отводящее головку от края чертежа, k_8 или k_{11} и k_4, отводящее головку от края чертежа, после чего не было k_8 или k_{12}
	k_9	Нет
	k_{12}	Нет
k_5	k_4	Нет
	k_6	Нет
	k_9	Нет
	k_{12}	Нет
k_6	k_5	Нет
	k_{10}	Нет
	k_{11}	Нет
	k_{12}	Нет
k_7	k_1	Нет
	k_2	Нет
	k_5	Нет
	k_{12}	Нет
k_8	k_3	Нет
	k_4	Нет
	k_9	Нет
	k_{12}	Нет

Окончание табл. 2

Первый элемент пары	Возможные вторые элементы пары	Условия существования произведения
k_9	k_3	Ранее или после последней k_8 или k_{12} не было k_2, k_5 или k_{11}
	k_4	Если ранее было k_{12} или k_{11}, после которых еще не было k_4, то k_4 должно отводить головку от края чертежа
	k_6	Ранее было k_2 и k_4, отводящее головку от края чертежа, k_5 или k_{11} и k_4, отводящее головку от края чертежа, после чего не было k_6 или k_{12}
	k_9	Нет
	k_{12}	Нет
k_{10}	k_1	Нет
	k_2	Нет
	k_5	Нет
	k_8	Нет
	k_{12}	Нет
k_{11}	k_4	Движение головки обеспечивает ее отвод от края чертежа
	k_9	Нет
	k_{12}	Нет
k_{12}	k_3	Нет
	k_4	Нет
	k_9	Нет
	k_{12}	Нет

подачи двух входных взаимодействий k_i, k_j, физически не допустимых или оставляющих внутреннее состояние неопределенным, должна быть предусмотрена реакция цифровой системы, указывающая на ошибочность (недопустимость) ситуации. В этом случае произведение $k_i \cdot k_j$ существует и учитывается в спецификации. При разработке цифровой системы нет необходимости предусматривать реакцию на какую-то последовательность входных взаимодействий только в том случае, если инициация того или другого входного взаимодействия осуществляется самой цифровой системой или блоком. В нашем случае, например, ввод кадра возможен только после нажатия кнопки "Пуск".

В табл. 2 предусмотрена графа 3, в которую заносится условие существования произведения $k_i \cdot k_j$. Если произведение $k_i \cdot k_j$ существует независимо от предыстории, то условие отсутствует. Наличие условия (произведения $k_2 \cdot k_4$, $k_4 \cdot k_3$, $k_4 \cdot k_6$, $k_9 \cdot k_3$, $k_9 \cdot k_4$, $k_9 \cdot k_6$, $k_{11} \cdot k_4$) указывает на зависимость существования произведений от предыдущих входных взаимодействий.

В случае отсутствия условий для всех произведений в таблице типа табл. 2 имеем второй случай, когда существование всех $k_i \cdot k_j$ не зависит от предыдущих взаимодействий. Тогда для каждого k_i, $k_i \in \mathbf{K}$, существует подмножество $\mathbf{K}^i$, $\mathbf{K}^i \subseteq \mathbf{K}$ такое, что $k_i \cdot k_j$, где $k_j \in \mathbf{K}^i$, всегда существует. В этом случае на множестве слов $\mathbf{F}$ задано отношение эквивалентности o, такое что $(f_1, f_2) \in$ o тогда и только тогда, когда f_1 и f_2 кончаются на одну букву.

Отношение эквивалентности o определяет разбиение $\mathbf{F}$ на классы эквивалентности $\mathbf{F}^i$, $i = 1, 2, ..., n$, где n — мощность алфавита $\mathbf{K}$. При отождествлении классов $\mathbf{F}^i$, $i = 1, 2, ..., n$, и $\mathbf{F}^x$, $x \in \mathbf{X}$, и соответственно множества внутренних состояний автомата (1) и множества $\mathbf{K}$ автомат A может быть заменен автоматом

$$A' = (\mathbf{K}, \mathbf{K}, x_н, \varphi), \qquad (2)$$

где $\mathbf{K}$ — множество входных символов и множество внутренних состояний; $x_н$ — начальное состояние, $x_н \in \mathbf{K}$; $\varphi: \mathbf{K} \times \mathbf{K} \to \mathbf{K}$ — частично определенное переходное отображение.

Таблица 3

Разделение функций цифровой системы управления чертежным автоматом на подфункции

Table 3. Subfunctions for drawing automata digital control system functions

Начальные функции	Новые функции	Описание новых функций
k_4	k_4^1	Передвижение головки в исходном состоянии без прерванного кадра
	k_4^2	Передвижение головки после срабатывания концевого выключателя, головка находится у края чертежа
	k_4^3	Передвижение головки после срабатывания концевого выключателя, головка отведена от края чертежа
	k_4^4	Передвижение головки после останова работы чертежного автомата нажатием кнопки "Стоп"
k_9	k_9^1	Изменение поправок в исходном состоянии без прерванного кадра
	k_9^2	Изменение поправок после срабатывания концевого выключателя, головка находится у края чертежа
	k_9^3	Изменение поправок после срабатывания концевого выключателя, головка отведена от края чертежа
	k_9^4	Изменение поправки после останова работы чертежного автомата нажатием кнопки "Стоп"

Каждое внутреннее состояние k, $k \in \mathbf{K}$ задает множество слов, заканчивающихся на букву k.

Представление автомата функций в виде (2) весьма удобно как при составлении спецификаций на входные взаимодействия, так и при использовании для выбора отладочных тестов.

В связи с этим попробуем привести табл. 2 к таблице без условий путем разделения элементов множества **K**, для которых имеются условия существования $k_i \cdot k_j$ в табл. 2, на несколько подэлементов (табл. 3).

Разделение элементов k_i на подэлементы и формирование нового алфавита **K** осуществлено в соответствии с различными внутренними состояниями автомата функций A. Полученная табл. 4 задает автомат вида A′.

Любую таблицу типа табл. 2 можно привести к виду табл. 4 путем разделения некоторых элементов алфавита **K** на подэлементы. При этом всегда существует физический смысл в таком разделении, связанный с различными внутренними состояниями цифровой системы или ее части.

Таблица 4

Возможные произведения $k_i \cdot k_j$ для цифровой системы управления чертежным автоматом после модификации алфавита функций

Table 4. Permissible products $k_i \cdot k_j$ for drawing automata digital control system after function alphabet modification

Первый элемент пары	Возможные вторые элементы пары
k_1	$k_1, k_2, k_5, k_8, k_{12}$
k_2	k_4^2, k_9^2, k_{12}
k_3	k_7, k_{12}
k_4^1	$k_3, k_4^1, k_9^1, k_{12}$
k_4^2	$k_4^3, k_6, k_9^3, k_{12}$
k_4^3	$k_4^2, k_6, k_9^3, k_{12}$
k_4^4	$k_4^4, k_6, k_9^4, k_{12}$
k_5	$k_4^4, k_6, k_9^4, k_{12}$
k_6	$k_5, k_{10}, k_{11}, k_{12}$
k_7	k_1, k_2, k_5, k_{12}
k_8	$k_3, k_4^1, k_9^1, k_{12}$
k_9^1	$k_3, k_4^1, k_9^1, k_{12}$
k_9^2	k_4^2, k_9^2, k_{12}
k_9^3	$k_4^3, k_6, k_9^3, k_{12}$
k_9^4	$k_4^4, k_6, k_9^4, k_{12}$
k_{10}	$k_1, k_2, k_5, k_8, k_{12}$
k_{11}	k_4^2, k_9^2, k_{12}
k_{12}	$k_3, k_4^1, k_9^1, k_{12}$

Методика формирования автомата функций

Автомат A′ является неприведенным. В целях упрощения выделим в **A′** классы эквивалентных состояний [8] и заменим каждый класс на одно состояние. Применительно к автомату вида (2) приведение означает выделение таких подмножеств $\widehat{\mathbf{K}}$ состояний, $\widehat{\mathbf{K}} \subset \mathbf{K}$, что для всех $\hat{k}$, $\hat{k} \in \widehat{\mathbf{K}}$, отображение $\varphi(k, \hat{k})$ определено для одинакового подмножества $\widetilde{\mathbf{K}}$ входных символов, и замену подмножеств $\widehat{\mathbf{K}}$ одним состоянием. Выделение подмножеств $\widehat{\mathbf{K}}$ может быть осуществлено по табл. 4 путем поиска одинаковых наборов функций в графе (2). После приведения автомата A′ получим автомат функций

$$A'' = (\mathbf{K}, \mathfrak{R}, x_н, \varphi{:}\mathbf{K}\times\mathfrak{R} \to \mathfrak{R}),$$

где **K** — алфавит функций; $\mathfrak{R}$ — множество подмножеств **K**, причем $\mathbf{K} = \bigcup \widehat{K}_i$, $\overline{\mathbf{K}}_{i_1} \cap \overline{\mathbf{K}}_{i_2} = \varnothing$ при $\overline{\mathbf{K}}_{i_1} \neq \overline{\mathbf{K}}_{i_2}$; $\mathfrak{R}$ — множество состояний приведенного автомата; $x_н$ — начальное состояние, $x_н \in \mathfrak{R}$.

Для дальнейшего упрощения автомата **A″** можно вновь полностью или частично объединить функции k — символы входного алфавита, соответствующие подаче одних и тех же входных взаимодействий в различных состояниях цифровой системы. Кроме того, можно объединить k_i и k_j — элементы входного алфавита, если все помеченные ими ребра в графе переходов автомата A″ всегда начинаются и заканчиваются в одних и тех же вершинах. Однако окончательное решение об объединении таких ребер принимает разработчик на основе физического смысла входных взаимодействий.

После проведения последней операции — объединения ряда ребер с соответствующим объединением элементов алфавита **K** — получим окончательный вид автомата функций:

$$A_ф = (\mathbf{K}, \mathbf{X}, x_н, \varphi{:}\mathbf{K}\times\mathbf{X} \to \mathbf{X}), \qquad (3)$$

где **K** — модифицированный алфавит входных символов — функций цифровой системы; **X** — множество внутренних состояний; $x_н$ — начальное состояние, $x_н \in \mathbf{X}$; φ — частично определенное переходное отображение.

Автомат функций вида (3) вместе с множествами $\mathbf{M}^k$ входных взаимодействий, соответствующих каждой функции k [7], является формальной спецификацией входных взаимодействий **M**. При этом задание множества входных взаимодействий в виде $\mathbf{M} = (\mathbf{F}, \{\mathbf{M}^k\})$ заменяется следующим образом:

$$\mathbf{M} = (\mathbf{K}, \mathbf{A}_ф, \{\mathbf{M}^k | k \in \mathbf{K}\}), \qquad (4)$$

где **K** — алфавит функций; $A_ф$ — автомат функций вида (3), определяющий допустимые последовательности функций **F**; $\mathbf{M}^k$ — множество входных взаимодействий, соответствующих k-й функции цифровой системы или блока.

Методика получения автомата функций по неформальному описанию работы цифровой системы или блока, приводимому в ТЗ, состоит в следующем.

1. Выделить перечень функций и соответствующие входные взаимодействия (аналогично табл. 1).

2. Составить таблицу возможных последовательных пар функций с указанием условий существования таких пар в виде, аналогичном табл. 2.

3. Разбить элементы перечня функций на подфункции либо по физическому смыслу, либо в зависимости от состояния системы, в котором инициируется второе входное взаимодействие пары, таким образом, чтобы можно было задать возможные пары функций независимо от предыстории работы (аналогично табл. 3, 4).

4. Объединить эквивалентные состояния полученного автомата функций, т. е. строки табл. 4 с одинаковым перечнем функций во втором столбце.

5. Объединить функции, разделенные на этапе 3, и, возможно, другие функции, всегда параллельные на графе переходов автомата функций.

Следует отметить, что выделение набора функций цифровой системы — в определенной степени процесс субъективный. Так, в случае чертежного автомата ввод кадра может быть представлен как последовательный ввод ряда чисел; ввод кадра и его отработка могут быть объединены в одну функцию. Однако, несмотря на различный вид автомата функций $\mathbf{A}_ф$ при выборе различных множеств **K** в связи с однозначностью требований к поведению цифровой системы получаемая спецификация входных взаимодействий приведет к проверке функционирования отлаживаемой системы одинаковым образом.

Предлагаемый подход к тестированию последовательностей выполнения функций цифровых систем

Для задания отладочных тестов в спроектированной цифровой системе необходимо выделить классы внутренних состояний $[x_i]$, гомоморфные каждому внутреннему состоянию x_i автомата функций $A_ф$. Множество отладочных тестов должно быть составлено таким образом, чтобы для каждого x_i, $x_i \in \mathbf{X}$, автомата $A_ф$ предусматривалась поочередная подача входных взаимодействий, соответствующих функциям ребер, выходящих из x_i на графе переходов автомата $A_ф$. При этом в процессе моделирования необходимо убедиться, что при установке автомата $A_ф$ в любое состояние из $[x_i]$ и подаче отладочного теста k_j автомат $A_ф$ переходит в одно из состояний $[\varphi(k_j, x_i)]$ и выдает правильное выходное воздействие. Такой метод выбора отладочных тестов соответствует тестированию всех переходов автомата функций спецификации.

Список литературы

1. **Lin Yi-Li, Su Alvin W. Y.** Functional Verification for SoC Software/Hardware Co-Design: From Virtual Platform to Physical Platform // 2011 IEEE International SOC Conference (SOCC). P. 201—206.

2. **Shi Jin, Liu Weichao, Jiang Ming** et al. Software Hardware Co-Simulation and Co-Verification in Safety Critical System Design // 2013 IEEE International Conference on Intelligent Rail Transportation (ICIRT). P. 71—74.

3. **Кащеев Н. И., Пономарев Д. М., Подъяблонский Ф. М.** Построение тестов цифровых схем с использованием обобщенной модели неисправностей и непрерывного подхода к моделированию // Вестник Нижегородского университета им. Н. И. Лобачевского. 2011. № 3 (2). С. 72—77.

4. **Иванников А. Д.** Формирование отладочного набора тестов для проверки функций цифровых систем управления объектами // Мехатроника, автоматизация, управление. 2017. Т. 18, № 12. С. 795—801.

5. **Иванников А. Д., Стемпковский А. Л.** Формализация задачи отладки проектов цифровых систем // Информационные технологии. 2014. № 9. С. 3—10.

6. **Оллонгрен А.** Определение языков программирования интерпретирующими автоматами. М.: Мир, 1977. 288 с.

7. **Иванников А. Д., Северцев В. Н.** Математическая модель множества входных воздействий цифровых систем при их моделировании на уровне логических сигналов // Информационные технологии. 2018. Т. 24, № 10. С. 627—632.

8. **Иванов Н. Н., Михайлов Г. И., Руднев В. В., Таль А. А.** Конечные автоматы: эквивалентность и поведение. М.: Наука, 1984. 192 с.

Control Digital System Function Set Defining While Debugging Tests Development

A. D. Ivannikov, adi@ippm.ru,
Institute for Design Problems in Microelectronics of the Russian Academy of Sciences,
Moscow, 124365, Russian Federation

Corresponding author: **Ivannikov Aleksandr D.,** D. Sc., Head Scientific Researcher,
Institute for Design Problems in Microelectronics of the Russian Academy of Sciences, Moscow,
124365, Russian Federation, e-mail: adi@ippm.ru

Accepted on August 30, 2018

Abstract

Digital control systems are considered, the functioning of which can be represented as a sequence of functions from a finite alphabet. For such systems projects debugging by simulation it is necessary to generate some set of tests for the applying on the simulated system to verify that it is functioning correctly. This paper is devoted to the development of test sets for function successions correctness. It is shown that on admissible function successions partly defined semigroup exists. There exists also word set on limited alphabet of functions, and they could be defined by some right liner grammar. Admissible successions are formally described by the graph of functions. Such a graph defines admissible functions for all digital system states. Digital system function set development is proposed in a way that admissible function successions could be defined as a graph. If the admissibility of two functions fulfillment one after another depends on previously fulfilled functions and the digital system internal state, then some functions should be divided into several subfunctions. The method of such a process is described. Developed graph of functions together with input interaction set for each digital system function define specification for digital system behavior. Proposed method is illustrated on the drawing machine control digital system functions development. The method of test set development on graph function is proposed.

Keywords: *debugging by simulation, digital systems, function alphabet, graph representation of function successions*

For citation:

Ivannikov A. D. Control Digital System Function Set Defining for Debugging Tests Development, *Mekhatronika, Avtomatizatsiya, Upravlenie*, 2018, vol. 19, no. 12, pp. 770—776.

DOI: 10.17587/mau.19.770-776

References

1. **Lin Yi-Li, Su Alvin W. Y.** Functional Verification for SoC Software / Hardware Co-Design: From Virtual Platform to Physical Platform, *2011 IEEE International SOC Conference (SOCC)*, pp. 201—206.

2. **Shi Jin, Liu Weichao, Jiang Ming** el al. Software Hardware Co-Simulation and Co-Verification in Safety Critical System Design, *2013 IEEE International Conference on Intelligent Rail Transportation (ICIRT)*, pp. 71—74.

3. **Kasheev N. I., Ponomarev D. M., Podyablonsky F. M.** *Postroenie testov cifrovih chem s ispolsovaniem obobshennoy modeli neispravnostei i neprerivnogo podhoda k modelirovaniu* (Digital circuits tests generation based on generalized malfunction model and continuous simulation approach), *Vestnik Nijegorodskogo Universiteta,* 2011, no. 3 (2), pp. 72—77 (in Russian).

4. **Ivannikov A. D.** *Formirovanie otladochnogo nabora testov dlya proverki funkcty cifrovih system upravleniya obektami* (Debugging Input Set Generation for Testing of Control Digital Systems Functions), *Mekhatronika, Avtomatizatsiya, Upravlenie*, 2017, vol. 18, no. 12, pp. 795—801 (in Russian).

5. **Ivannikov A. D., Stempkovsky A. L.** *Formalizaciya zadachi otladki proektov cifrovih sistem* (Formal model of digital system design debugging task), *Informacionnie Technologii*, 2014, no. 9, pp. 3—10 (in Russian).

6. **Ollongren A.** Opredelenie yazikiv programmirovaniya interpretiruushimi avtomatami (Programming Languages Definition by Interpretation Automata), Moscow, Mir, 1977, 288 p.

7. **Ivannikov A. D., Severcev V. N.** *Matematicheskaya model mnojestva vhodnih vozdeistviy cifrovih system pri ih modelirovanii na urovne logicheskih signalov* (Mathematical Model of Digital System Input Impact Set while Modelling on Logical Signal Level), *Informacionnie Technologii*, 2018, no. 10, pp. 627—632 (in Russian).

8. **Ivanov N. N., Mikhailov G. I., Rudnev V. V., Tal A. A.** *Konechnie avtomati: ekvivalentnost i povedenie* (Finite automata: equivalence and behavior), Moscow, Nauka, 1984, 192 p. (in Russian).