

А. Д. Иванников, д-р техн. наук, зам. директора по научной работе, adi@ippm.ru,
Институт проблем проектирования в микроэлектронике РАН

Формирование отладочного набора тестов для проверки функций цифровых систем управления объектами¹

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 минимального полного в смысле определенного критерия набора тестовых воздействий на моделируемую систему для проверки правильности ее функционирования. Управляющие цифровые системы моделируются на уровне логических сигналов, которыми они обмениваются с внешней средой, в том числе с управляемыми объектами. В качестве входных данных для моделирования используются входные взаимодействия, включающие как собственно входные сигналы, так и выходные сигналы управления обменом. Предлагается алгоритм формирования минимально полного набора тестов для отладки проектов, основанный на задаваемых разработчиком классах эквивалентности входных взаимодействий.

Ключевые слова: отладка методом моделирования, логическое моделирование цифровых систем, отладочные тесты

Введение

Будем рассматривать проектирование цифровых систем, реализуемых в виде аппаратного обеспечения (одна или несколько интегральных схем) и программного обеспечения, которые в реальном времени осуществляют управление работой некоторого устройства: промышленного робота, технологической установки, бытовой техники и др. При проектировании таких систем для отладки проектов широко используется метод моделирования. На компьютерную модель цифровой системы подаются некоторые входные воздействия, а реакция модели проектируемой системы проверяется на соответствие техническому заданию.

При этом важной задачей является выбор конечного числа конечных по времени тестовых входных взаимодействий (тестовых примеров). С ростом сложности проектируемых цифровых систем и, соответственно, с ростом сложности и длительности тестирования их проектов все более актуальной становится задача выбора минимального полного в определенном смысле набора тестов, правильное выполнение которого позволяет убедиться в отсутствии ошибок проектирования [1, 2]. При решении этой задачи вначале необходимо выбрать уровень модели цифровой управляющей системы. Обычно осуществляется декомпозиция задачи отладки проекта [3], прежде всего по типу выявляемых ошибок. Так, для верификации временных диаграмм обмена информацией между блоками используются модели цифровых элементов с многозначным представлением электрических сигналов [4]. Для

проверки правильности логического функционирования используются модели с булевым представлением сигналов на входах и выходах [1, 5]. Используются также различные высокоуровневые модели [6, 7].

Наиболее часто используемым подходом является введение некоторой модели неисправностей, которые должны быть исключены в результате отладки проекта [1, 8]. Существуют также подходы к проверке правильности функционирования цифровой системы на основе имеющейся формальной спецификации [1, 5, 9].

Целью настоящей работы является формализация выбора тестовых примеров для отладки проектов цифровых систем управления исходя из перечня выполняемых ими функций, что не требует наличия детализированной формальной спецификации на функционирование проектируемой цифровой системы.

Описание используемой модели цифровых систем управления

Взаимодействие цифровой системы с объектом управления и внешним миром вообще осуществляется через внешние линии и шины — наборы линий, по которым передается однородная информация, например адреса или данные. В цифровых системах управления широко используются двунаправленные шины и линии, имеющие также состояние с высоким выходным сопротивлением (отключенное состояние). Будем рассматривать логическую модель сигналов на шинах и линиях цифровых систем, т. е. считать, что значения сигналов представляются на линиях как 0 или 1 и как число из диапазона $0...2^n$ — на шинах системы. Цифровые сигналы P внешних шин и линий назовем терми-

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

нальными переменными, они составляют множество $\mathbf{P}$. Переменная $p \in \mathbf{P}$ всегда имеет одно из значений конечного множества $\mathbf{Z}_p$, элементы которого определяют как целочисленное значение сигнала, так и направленность работы шины или линии.

Событием по переменной p называется изменение переменной p со значения $z_1 \in \mathbf{Z}_p$ на значение $z_2 \in \mathbf{Z}_p$ в момент времени t . Обозначим такое событие χ_{p, z_1, z_2}^t . Взаимодействие цифровой системы

с внешней средой, включая управляемый объект, есть последовательность переключений сигналов на терминальных шинах и линиях, т. е. последовательность событий. Для каждой проектируемой системы имеется множество Ψ допустимых взаимодействий с внешней средой, каждое из которых есть отображение $\psi: [0, t) \rightarrow \mathbf{Q}$, $t \in \mathbf{T}$, $\mathbf{Q} = \prod_{p \in \mathbf{P}} \mathbf{Z}_p$.

В цифровых системах для каждого конечного временного интервала число событий по терминальным переменным, т. е. число изменений их значений, конечно. В связи с этим любое взаимодействие ψ может быть представлено в виде вектора $(z_{p_1}^n, \dots, z_{p_k}^n)$ начальных значений переменных $p_1, \dots, p_k$ (k — мощность множества $\mathbf{P}$) в момент времени $t = 0$ и последовательности событий по переменным множества $\mathbf{P}$ с конечным числом событий за любой конечный интервал времени.

Если в этой последовательности выделить только события, являющиеся изменениями входных сигналов, то такую последовательность можно назвать входным воздействием. Однако часто моменты подачи входных сигналов на цифровую систему определяются готовностью системы принять эти сигналы, на что указывают определенные выходные сигналы системы. Выполнение какой-либо операции, например считывания данных цифровой системой, может инициироваться не сигналами внешней среды, а самой системой. В связи с этим использование в качестве аргументов функционирования цифровой системы входных воздействий не всегда удобно.

Выделим из последовательности событий взаимодействия ψ последовательность входных событий и выходных событий управления обменом, которые по заданному протоколу обмена обуславливают моменты времени входных событий. Назовем эту последовательность входным взаимодействием

$$\mu = (z_{p_1}^n, \dots, z_{p_{n+q}}^n), \chi_{p_{i_1}, z_{j_1}, z_{j_2}}^{t_1}, \chi_{p_{i_2}, z_{j_3}, z_{j_4}}^{t_2}, \chi_{p_{i_3}, z_{j_5}, z_{j_6}}^{t_3}, \dots, \quad (1)$$

где $t_1 \leq t_2 \leq t_3 \leq \dots$ — упорядоченная последовательность времен событий; $p_{i_1}, p_{i_2}, p_{i_3}, \dots$ — переменные, принадлежащие множеству $\mathbf{P}$; $z_{j_1}, z_{j_3}, z_{j_5}, \dots$ — значения переменных непосредственно перед событием; $z_{j_2}, z_{j_4}, z_{j_6}, \dots$ — значения переменных непосред-

ственно после события; $\chi_{p_{i_1}, z_{j_1}, z_{j_2}}^{t_1}, \chi_{p_{i_2}, z_{j_3}, z_{j_4}}^{t_2}, \chi_{p_{i_3}, z_{j_5}, z_{j_6}}^{t_3},$

... — входные события и выходные события управления обменом; n — число входных переменных; q — число выходных переменных управления обменом.

В рассматриваемой модели в качестве аргументов функционирования цифровых систем используются входные взаимодействия, что дает возможность рассматривать режимы работы, инициируемые как внешними входными сигналами, так и самими цифровыми системами. Более подробно формальное представление допустимых взаимодействий цифровых систем рассмотрено в работах [3, 10].

Структура множества допустимых входных взаимодействий

Пусть для некоторой проектируемой цифровой системы заданы входные взаимодействия

$$\mu_1: [0, t_1) \rightarrow \mathbf{U} \times \mathbf{Y}^0, \mu_2: [0, t_2) \rightarrow \mathbf{U} \times \mathbf{Y}^0, \\ t_1 \in T_1, t_2 \in T_2,$$

где $\mathbf{U}$ — множество состояния выходных переменных; $\mathbf{Y}^0$ — множество состояния выходных переменных управления обменом. Определим произведение $\mu_1, \mu_2 = \mu_1 \cdot \mu_2, \mu_1, \mu_2: [0, t_1 + t_2) \rightarrow \mathbf{U} \times \mathbf{Y}^0$ как

$$\mu_1, \mu_2 = \mu_1 \cdot \mu_2 = \begin{cases} \mu_1(t) & \text{при } t \in [0, t_1); \\ \mu_2(t - t_1) & \text{при } t \in [t_1, t_1 + t_2), \end{cases}$$

также являющееся входным взаимодействием.

Тогда на множестве входных взаимодействий $\mathbf{M}$ определена в общем случае частичная мультипликативная полугруппа $\langle \mathbf{M}, \cdot \rangle$.

Практика показывает, что каждая цифровая система выполняет некоторую последовательность функций из конечного алфавита $\mathbf{K}$, причем выполнение каждой функции вызывается одним из входных взаимодействий определенного класса. Из этого утверждения следует, что полугруппа $\langle \mathbf{M}, \cdot \rangle$ имеет бесконечное множество $\bar{\mathbf{M}}$ порождающих элементов, причем $\bar{\mathbf{M}} = \bigcup_{k \in \mathbf{K}} \mathbf{M}^k$, где $\mathbf{M}^k$ — множество входных взаимодействий, обуславливающих выполнение цифровой системой функции k .

С точки зрения влияния на поведение цифровой системы различия двух входных взаимодействий μ_1 и μ_2 , $\mu_1 \in \bar{\mathbf{M}}$, $\mu_2 \in \bar{\mathbf{M}}$ могут быть в той или иной степени "невелики". Так, одно из входных взаимодействий может содержать, а другое не содержать одно или несколько безразличных событий, никак не влияющих на функционирование цифровой системы. Примером несущественного события может служить следующая ситуация. После того, как выполнено считывание информации с какого-либо входа, и до момента, когда цифровая система может опять обратиться к этому же источнику информации, сигнал на этом входе может переключиться в любое состояние или содержать любое число произвольных переключений. При другом критерии "близости" "почти одинаковыми" являются входные

взаимодействия, содержащие равное число событий, отличающихся только значениями ряда t_j в соотношении (1) в пределах допустимых ограничений. Могут быть рассмотрены и более крупные группы входных взаимодействий.

Представим множество значений данных $\mathbf{D}$ входных взаимодействий, обуславливающих выполнение цифровой системой определенной функции k ,

как $\mathbf{D} = \bigcup_{i=1}^n \mathbf{D}_i$, $\mathbf{D}_i \cap \mathbf{D}_j = \emptyset$ при $i \neq j$. В одну группу входных взаимодействий могут быть отнесены взаимодействия, данные которых принадлежат одному и тому же подмножеству $\mathbf{D}_i$. Каждое подмножество $\mathbf{D}_i$, в свою очередь, может быть разбито на непересекающиеся подмножества $\mathbf{D}_{ij}$ и т.д. Разбиение множества значений данных должно осуществляться исходя из физического смысла задачи таким образом, чтобы входные взаимодействия с данными различных подмножеств $\mathbf{D}_i$ обуславливали несколько различных алгоритмы их обработки. Самой большой группой являются множества $\mathbf{M}^k$, $k \in K$ входных взаимодействий, обуславливающих выполнение цифровой системой функции k .

Математическим аналогом понятия "близости" входных взаимодействий является отношение эквивалентности [11]. Пусть имеется конечное множество Λ отношений эквивалентности λ . В частности, множество Λ обязательно включает отношения эквивалентности λ' , λ'' :

- $(\mu_1, \mu_2) \in \lambda'$ тогда и только тогда, когда $\mu_1 \in \mathbf{M}^k$ и $\mu_2 \in \mathbf{M}^k$. Именно это отношение эквивалентности позволяет выделить входные взаимодействия, обуславливающие выполнение функции k , в множестве $\mathbf{M}^k$;
- $(\mu_1, \mu_2) \in \lambda''$, где $\mu_1 \in \mathbf{M}^k$, $\mu_2 \in \mathbf{M}^k$, тогда и только тогда, когда набор данных d_1 и d_2 , присутствующих в μ_1 , μ_2 , принадлежит одной и той же подобласти $\mathbf{D}_i$ данных, где $\bigcup_{i=1}^n \mathbf{D}_i = \mathbf{D}$, $\mathbf{D}_i \cap \mathbf{D}_j = \emptyset$ при $i \neq j$, $\mathbf{D}$ — область данных для $\mathbf{M}^k$. Так, возможен случай, когда $(\mu_1, \mu_2) \in \lambda''$ тогда и только тогда, когда μ_1 и μ_2 , заданные в виде (2), отличаются в ряде событий только значениями z'_j или z''_j , соответствующими различным данным на информационных входах, обрабатываемых по одинаковому алгоритму, и изменяющими выходные последовательности цифровой системы или ее блока только в части значений на информационных выходах. Может быть задано несколько отношений эквивалентности такого типа для различных разбиений области данных $\mathbf{D}$.

В случае, когда функция k предусматривает периодический фрагмент последовательности событий, который может повторяться в различных входных взаимодействиях множества $\mathbf{M}^k$ разное число раз (например, ввод различного числа

слов информации), может быть использовано отношение эквивалентности λ''' : $(\mu_1, \mu_2) \in \lambda'''$, если μ_1 и μ_2 содержат одинаковое число периодически повторяющихся фрагментов. Могут быть заданы и другие отношения эквивалентности.

Выбор множества Λ отношений эквивалентности должен осуществляться разработчиком исходя из требуемого поведения разрабатываемой цифровой системы и физического смысла задачи. При этом, как будет ясно далее, множество Λ задается косвенно.

Каждое отношение эквивалентности λ множества Λ задается на своем множестве. Так, отношение λ' задано на всем множестве $\bar{\mathbf{M}} = \bigcup_{k \in K} \mathbf{M}^k$. На

каждом множестве $\mathbf{M}^k$, являющемся классом эквивалентности $\bar{\mathbf{M}}$ по λ' , задаются свои отношения эквивалентности типа отношения λ'' . Так, возможен случай, когда области данных для $\mathbf{M}^k$ есть $\mathbf{D} = \mathbf{D}^1 \times \mathbf{D}^2 \times \dots \times \mathbf{D}^n$, а на множестве значений $\mathbf{D}^i$, $i = 1, \dots, n$, каждого параметра входных взаимодействий из $\mathbf{M}^k$ определены свои отношения эквивалентности $\lambda_1, \lambda_2, \dots, \lambda_n$, где $\lambda_1 \in \Lambda$, $\lambda_2 \in \Lambda$, ..., $\lambda_n \in \Lambda$. Возможен и другой случай, когда отношение λ_1 оп-

ределяет разбиение $\mathbf{D} = \bigcup_{i=1}^n \mathbf{D}_i$, $\mathbf{D}_i \cap \mathbf{D}_{i'} = \emptyset$ при $i \neq i'$, а отношение λ_2 определено только на одном подмножестве или на части подмножеств $\mathbf{D}_{i_1}, \dots, \mathbf{D}_{i_l}$, где $\{i_1, \dots, i_l\} \subset \{1, \dots, n\}$.

Если на некотором множестве $\mathbf{M}^*$ (в качестве $\mathbf{M}^*$ может выступать $\bar{\mathbf{M}}$, $\mathbf{M}^k$ или класс эквивалентности $\mathbf{M}^k$ по отношению эквивалентности λ) задано отношение λ_1 , $\lambda_1 \in \Lambda$, то существует разбиение $\mathbf{M}^*$ на классы эквивалентности $\mathbf{M}^{*i}$ по отношению λ_1 . Если на некотором $\mathbf{M}^{*i}$, определено отношение λ_2 , $\lambda_2 \in \Lambda$, то для μ , где $\mu \in \mathbf{M}^{*i}$, существует произведение эквивалентностей $\lambda_1 \cdot \lambda_2$, которое всегда является эквивалентностью [131]. Произведения эквивалентностей $\prod_{\lambda_i \in \Lambda'} \lambda_i$, где $\Lambda' \subseteq \Lambda$, которые всегда являются эк-

вивалентностями, позволяют провести классификацию множества $\bar{\mathbf{M}}$ с той степенью подробности, которая необходима разработчику, определившему множество Λ . Входные взаимодействия каждого класса эквивалентности, определяемого максимально возможными произведениями $\prod_{\lambda_i \in \Lambda'} \lambda_i$, где

$\Lambda' \subseteq \Lambda$, различаются только значениями ряда t_j в соотношении (1) в пределах, не нарушающих ограничений на допустимые времена событий.

Как указывалось выше, самыми крупными классами эквивалентности на множестве $\bar{\mathbf{M}}$ являются множества $\mathbf{M}^k$, соответствующие выделению алфавита выполняемых функций $\mathbf{K}$.

Проиллюстрируем вышесказанное на примере цифровой системы управления несложным чертежным автоматом. Приведем его краткое функциональное описание. Чертежный автомат включает плоский планшет, на котором осуществляется вычерчивание, два шаговых двигателя, управляющие 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 в начале сеанса вычерчивания и при необходимости его остановки, и кнопки "П1", "П2" ("Перо 1" и "Перо 2"), "↑", "↓", "→", "←" для ручного опускания и перемещения вычерчивающих головок, кнопка "—" —, при нажатии которой вычерчивается пунктирная 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, пунктирной или штрихпунктирной линии. Система управления должна также реагировать на сигналы концевых выключателей и нажатия управляющих кнопок.

Так, для цифровой системы управления чертежным автоматом функциями являются ввод и обработка кадра, движение пера при нажатии одной из кнопок ручного управления вверх, вниз, влево или вправо.

Для функции ввода и обработки кадра классами эквивалентности по некоторому λ_1 могут являться входные взаимодействия, задающие вычерчивание дуг и прямых, классами эквивалентности по λ_2 — входные взаимодействия, задающие сплошную, пунктирную или штрихпунктирную линию. На классе эквивалентности входных взаимодействий, задающих вычерчивание прямых, задается отношение λ_3 , определяющее подклассы эквивалентности горизонтальных, вертикальных и наклонных отрезков.

Составление набора отладочных тестов

Набор отладочных тестов должен проверять как возможность выполнения одной функции после другой, если такая последовательность является допустимой, так и правильность выполнения са-

мых функций. При этом правильность выполнения самих функций является основой.

Совершенно ясно, что большее число отладочных тестов позволяет говорить о правильной работе отлаживаемой системы с большей уверенностью. В связи с тем, что большее число отлаживаемых тестов повышает трудоемкость и временные затраты на отладку, необходимо, чтобы при увеличении числа тестовых примеров все они были наиболее информативными.

Как указывалось выше, для входных взаимодействий, вызывающих выполнение одной функции, определено множество Λ отношений эквивалентности. Каждому λ_i , где $\lambda_i \in \Lambda$, поставим в соответствие признак $\mathcal{L}_i$ — переменную с конечным множеством значений $\mathbf{Z}_i^*$. Каждое значение z_i^* , $z_i^* \in \mathbf{Z}_i^*$, признака $\mathcal{L}_i$ указывает на принадлежность μ , $\mu \in \mathbf{M}$, j -му классу эквивалентности входных взаимодействий по λ_i .

Так, для цифровой системы управления чертежным автоматом значениями одного из признаков $\mathcal{L}_i$ является вид вычерчиваемого кадра: отрезок прямой или дуга окружности, значениями другого признака $\mathcal{L}_i$ — тип линии: сплошная, пунктирная или штрихпунктирная. Разработчик задает не множество эквивалентностей Λ , а набор признаков $\mathcal{L}_1, \mathcal{L}_2, \dots, \mathcal{L}_n$ и алфавиты их значений. Множество значений признака $\mathcal{L}_1$ всегда есть $\mathbf{K}$. Признак $\mathcal{L}_1$ определяет разбиение $\mathbf{M}$ на $\mathbf{M}^k$.

Сформулируем задачу составления набора отладочных тестов для проверки отдельных функций формально. Пусть имеется конечное множество признаков $\mathcal{L}_1, \mathcal{L}_2, \dots, \mathcal{L}_n$, каждый из которых может принимать конечное множество значений $\mathbf{Z}_1^*, \mathbf{Z}_2^*, \dots, \mathbf{Z}_n^*$. Пусть признаки $\mathcal{L}_1, \mathcal{L}_2, \dots, \mathcal{L}_n$ упорядочены таким образом, что для каждого $\mathcal{L}_l$, $l = 2, \dots, n$, существует один и только один признак $\mathcal{L}_i$, $i < l$, такой, что значение $\mathcal{L}_l$ для входного взаимодействия μ определено в том и только в том случае, если признак $\mathcal{L}_i$ для μ

имеет значение $z_i^{j_i}$, $z_i^{j_i} \in \mathbf{Z}_i^*$. Тогда взаимосвязь признаков можно представить двудольным графом $G(v, \varepsilon)$ (см. рисунок). Множество вершин этого графа есть $V = \{\mathcal{L}_1, \dots, \mathcal{L}_n\} \cup \left\{ \bigcup_{i=1}^n \mathbf{Z}_i^* \right\}$. Одна группа вершин представляет множество признаков, а другая группа — возможные значения каждого признака. Множество ребер $\varepsilon = \varepsilon^1 \cup \varepsilon^2$, $\varepsilon^1 \cap \varepsilon^2 = \emptyset$. Ребра множества ε^1 соединяют вершину $\mathcal{L}_i$ с вершиной z , если $z \in \mathbf{Z}_i^*$ (на рисунке — сплошные линии). Ребра множества ε^2 соединяют вершину z_i^j , где $z_i^j \in \mathbf{Z}_i^*$, с вершиной $\mathcal{L}_l$, если признак $\mathcal{L}_l$ определен для входного взаимодействия μ в том случае, если для этого входного взаимодействия $\mathcal{L}_i = z_i^j$ (на рисунке — штриховые линии).

Пусть множество отладочных тестов считается полным, если для любого признака $\mathcal{L}_i$, $i = 1, \dots, n$, и $z \in \mathbf{Z}_i^*$ в множестве отладочных тестов найдется по крайней мере одно входное взаимодействие, для которого $\mathcal{L}_i = z$. Минимальным полным множеством отладочных тестов назовем такое полное множество тестов, число тестов в котором минимально. Задача составления минимального полного множества отладочных тестов состоит в выборе сочетаний значений признаков для каждого отладочного теста множества.

Алгоритм формирования минимального множества отладочных тестов

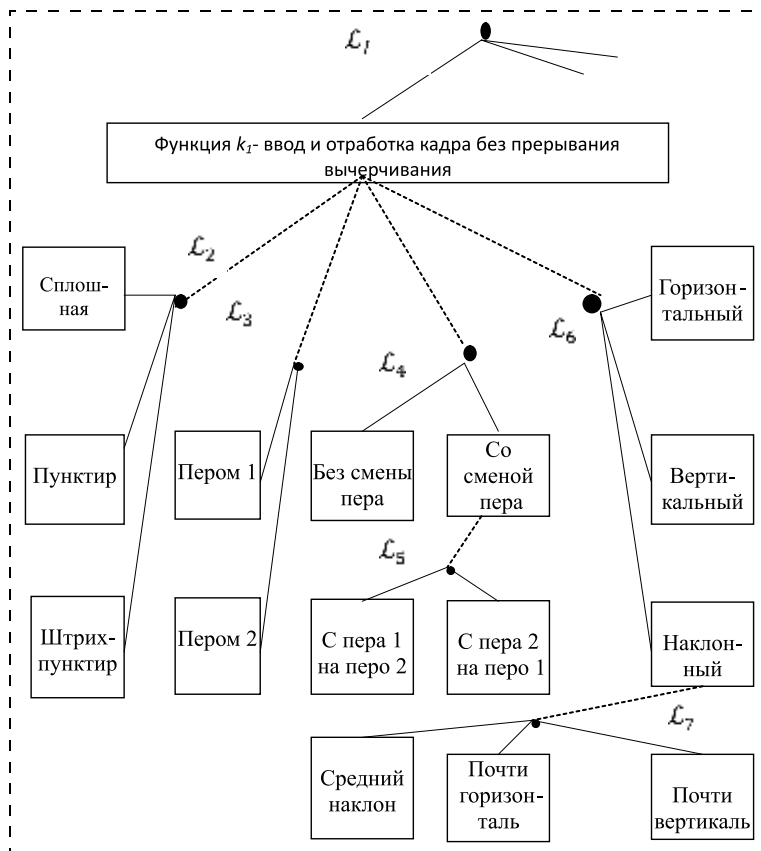
1. Присвоим ранги вершинам $\mathcal{L}_1, \mathcal{L}_2, \dots, \mathcal{L}_n$, равные числу ребер из множества $\mathfrak{E}^2$, входящих в путь от $\mathcal{L}_1$ до $\mathcal{L}_i$. Вершина $\mathcal{L}_1$ имеет ранг 0. Поскольку в $\mathcal{G}^{\mathcal{L}}(\mathbf{v}, \mathfrak{E})$ каждая вершина достижима из вершины $\mathcal{L}_1$, то все вершины $\mathcal{L}_1, \dots, \mathcal{L}_n$ имеют ранг, причем ранг 0 имеет только одна вершина $\mathcal{L}_1$.

2. Среди вершин множества $\{\mathcal{L}_1, \dots, \mathcal{L}_n\}$, имеющих максимальный ранг, выделим вершину $\mathcal{L}_{i_1}$, такую что $|\mathbf{Z}_{i_1}^*|$ максимально.

Рассмотрим подграф $\mathcal{G}_{i_1}$ графа $\mathcal{G}(\mathbf{v}, \mathfrak{E})$, который включает вершину $\mathcal{L}_{i_1}$, вершины z_{i_1} ,

$z_{i_1} \in \mathbf{Z}_{i_1}^*$, вершину z_l , $z_l \in \mathbf{Z}_l^*$, связанную с вершиной $\mathcal{L}_{i_1}$ ребром из множества $\mathfrak{E}^2$, вершины $\mathcal{L}_{i_2}, \dots, \mathcal{L}_{i_m}$, связанные с вершиной z_l ребрами из множества $\mathfrak{E}^2$ (если такие вершины имеются), вершины z , где $z \in \mathbf{Z}_{i_2}^*$ или $z \in \mathbf{Z}_{i_3}^* \dots$ или $z \in \mathbf{Z}_{i_n}^*$, а также ребра, соединяющие перечисленные вершины. Все вошедшие в подграф $\mathcal{G}_{i_1}$ вершины из множества $\{\mathcal{L}_1, \dots, \mathcal{L}_n\}$ имеют максимальный ранг.

Подграфу $\mathcal{G}_{i_1}$ поставим в соответствие $|\mathbf{Z}_{i_1}^*|$ наборов значений признаков $(z_{i_1}, z_{i_2}, \dots, z_{i_m})$, $z_{i_1} \in \mathbf{Z}_{i_1}^*$, $z_{i_2} \in \mathbf{Z}_{i_2}^*, \dots, z_{i_m} \in \mathbf{Z}_{i_m}^*$. Признак $\mathcal{L}_{i_1}$ в первом наборе имеет значение $z_{i_1}^1$, во втором наборе $z_{i_1}^2$ и так далее. В последнем наборе признак $\mathcal{L}_{i_1}$ имеет значение $z_{i_1}^k$, где $k = |\mathbf{Z}_{i_1}^*|$. Признаки $\mathcal{L}_{i_s}$, $s = 2, \dots, m$, в первом наборе имеют значения $z_{i_s}^1$, и в каждом последующем $z_{i_s}^{(p+1)}$, где p — номер значения признака $\mathcal{L}_{i_1}$ в предыдущем наборе.



Часть графа $\mathcal{G}(\mathbf{v}, \mathfrak{E})$ для цифровой системы управления чертежным автоматом (для наглядности учтено вычерчивание только отрезков прямых)

В случае, если подграф $\mathcal{G}_{i_1}$ содержит только одну вершину из множества $\{\mathcal{L}_1, \dots, \mathcal{L}_n\}$, а именно $\mathcal{L}_{i_s}$, наборы значений признаков суть $(z_{i_1}^1), (z_{i_1}^2), \dots, (z_{i_1}^k)$.

Удалим из графа $\mathcal{G}(\mathbf{v}, \mathfrak{E})$ подграф $\mathcal{G}_{i_1}$ и ребро из множества $\mathfrak{E}^1$, связывающее вершины z_l и $\mathcal{L}_l$. Добавим в полученный граф $|\mathbf{Z}_{i_1}^*|$ вершин, помеченных наборами значений признаков $(z_{i_1}, z_{i_1}, z_{i_2}, \dots, z_{i_m})$, $z_l \in \mathbf{Z}_l^*$, причем для всех $|\mathbf{Z}_{i_1}^*|$ вершин значение z_{i_1} одинаково, а поднаборы $(z_{i_1}, z_{i_2}, \dots, z_{i_m})$ равны наборам значений признаков, соответствующих подграфу $\mathcal{G}_{i_1}$. Соединим добавленные вершины с вершиной $\mathcal{L}_l$, полученные ребра отнесем к множеству $\mathfrak{E}^1$. Полученный граф примем за $\mathcal{G}(\mathbf{v}, \mathfrak{E})$.

3. Если $\mathcal{L}_l$ не есть $\mathcal{L}_1$, то повторим пункт 2. Если $\mathcal{L}_l$ есть $\mathcal{L}_1$, то полученное множество наборов значений признаков представляет отладочные тесты минимального полного множества.

Анализ предложенного алгоритма

Покажем, что приведенный выше алгоритм всегда сходится, а полученное множество наборов

значений признаков есть минимальное полное множество отладочных тестов.

Каждое выполнение шага 2 алгоритма уменьшает число вершин множества $\{L_1, \dots, L_n\}$ в графе $G(v, \epsilon)$. Следовательно, после конечного числа выполнения шага 2 граф $G(v, \epsilon)$ будет содержать только одну вершину из множества $\{L_1, \dots, L_n\}$, а именно вершину L_1 , что является признаком окончания работы.

Для каждого подграфа G_i , выделяемого на шаге 2, наборы значений признаков будут содержать все возможные значения признаков $L_{i_1}, \dots, L_{i_m}$, вошедших в подграф G_i . Все наборы значений признаков подграфа G_i сохраняются в новом графе $G(v, \epsilon)$. Следовательно, для всех признаков полученные наборы будут содержать все значения, т. е. полученное множество наборов значений признаков будет полным множеством отладочных тестов.

Для каждого подграфа G_i , выделяемого на шаге 2, полученное множество наборов признаков минимально и содержит $\max |Z_{i_k}^*|$, $k = 1, \dots, m$, наборов. Эти наборы признаков могут войти в множество окончательных наборов только в сочетании со значением z_l признака L_l , т. е. значение z_l признака должно встречаться в полном множестве отладочных тестов не менее $\max |Z_{i_k}^*|$, $k = 1, \dots, m$, раз. Следовательно, полученное в результате работы алгоритма множество отладочных тестов будет минимальным.

Пример минимального полного множества отладочных тестов, соответствующего рисунку, приведен в таблице.

Минимальная мощность минимального полного множества отладочных тестов есть $\max |Z_i^*|$, $i = 1, \dots, n$, где n — число признаков $L_1, \dots, L_n$; максимальная мощность минимального полного множества отладочных тестов есть $\sum_{i=1}^n (|Z_i^*| - 1) + 1$. По-

кажем это.

Минимальное полное множество отладочных тестов не может содержать менее чем $\max |Z_i^*|$, $i = 1, \dots, n$, тестов, так как в связи с нашим предположением для каждого признака множества $\{L_1, \dots, L_n\}$,

Минимальное полное множество отладочных тестов для проверки функции K_1 -ввода и отработки кадра без прерывания вычерчивания

Номер набора значений признаков	Описание
1	Сплошной горизонтальный отрезок, вычерчиваемый пером 1 без смены пера
2	Пунктирный вертикальный отрезок, вычерчиваемый со сменой пера с 1-го на 2-е
3	Штрихпунктирный отрезок среднего наклона, вычерчиваемый со сменой пера со 2-го на 1-е
4	Сплошной, почти горизонтальный отрезок, вычерчиваемый пером 1 без смены пера
5	Пунктирный, почти вертикальный отрезок, вычерчиваемый со сменой пера с 1-го на 2-е

в том числе для L_i , для которого $|Z_i^*|$ максимально, полное множество тестов должно содержать все возможные значения из Z_i^* .

Минимальное полное множество будет содержать $\max |Z_i^*|$, $i = 1, \dots, n$, отладочных тестов в том случае, когда в графе $G(v, \epsilon)$ все вершины $L_2, \dots, L_n$ будут непосредственно связаны с L_1 ребрами подмножества ϵ^2 .

При наличии только одного признака L_1 минимальное полное множество отладочных тестов содержит $|Z_1^*|$ тестов, при двух признаках L_1, L_2 — $(|Z_1^*| + |Z_2^*| - 1)$ тестов, при трех признаках L_1, L_2, L_3 — не более $(|Z_1^*| + |Z_2^*| - 1 + |Z_3^*| - 1)$ тестов и т.д. Таким образом, максимальная мощность минимального полного множества отладочных тестов

равна $\sum_{i=1}^n (|Z_i^*| - 1) + 1$. Минимальное полное мно-

жество отладочных тестов содержит $\sum_{i=1}^n (|Z_i^*| - 1) + 1$

тестов в том случае, если в графе $G(v, \epsilon)$ каждой вершине z , где $z \in Z_i^*$, $i = 1, \dots, n$, инцидентно не более одного ребра из множества ϵ^2 .

Заключение

Предлагаемый алгоритм обеспечивает формирование минимального множества отладочных тестов на основе заданного разработчиком перечня отношений эквивалентности на множестве входных взаимодействий. Другими словами, разработчик выбирает разделение каждой функции, выполняемой цифровой системой управления, на подфункции исходя из физического смысла. Безусловно, при этом присутствует некоторой субъективный фактор. При иерархическом разделении функций на подфункции разработчик в какой-то мере ориентируется на свое понимание о том, как данная подфункция реализуется в цифровой системе управления. Тем не менее, выбор тестовых примеров для отладки проектов цифровых систем управления объектами на основе проверки выполняемых функций является весьма эффективным.

Список литературы

1. Кашеев 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.
2. Cruz A. M., Fernandez R. B., Lozano H. M., Ramirez Salinas M. A., Vila Vargas L. A. Automated Functional Test Generation for Digital Systems Through a Compact Binary Differential Evolution Algorithm // Journal of Electronic Testing-Theory and Applications. 2015. Vol. 31, N. 4. P. 361–380.
3. Зеленко Г. В., Иванников А. Д., Рошин А. В., Стемковский А. Л. Алгебраические модели декомпозиции задачи отладки проектов цифровых систем с помощью моделирования //

7. **Jain S., Govani P., Podda K. B., La A. K., Parmar R. M.** Functional verification of DSP based on-board VLSI design //

11. **Мальцев А. И.** Алгебраические системы. М.: Наука. 1970. 392 с.

1. **Kasheev N. I., Ponomarev D. M., Podyablonsky F. M.** *Postroenie testov cifrovih chem s ispolzovaniem obobshennoy modeli neispravnosti 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).
2. **Cruz A. M., Fernandez R. B., Lozano H. M., Ramirez Salinas M. A., Vila Vargas L. A.** Automated Functional Test Generation for Digital Systems Through a Compact Binary Differential Evolution Algorithm, *Journal of Electronic Testing-Theory and Applications*, 2015, vol. 31, no. 4, pp. 361—380.
3. **Zelenko G. V., Ivannikov A. D., Roshin A. V., Stempkovsky A. L.** *Algebraicheskie modeli dekompozicii zadachi otladki proektov cifrovih sistem s pomoshu modelirovaniya* (Algebraic models for digital system design debugging by simulation), *Problemi razrabotki perspektivnih mikro-I nanoelertronnih system*, 2016, no. 3, pp. 232—239 (in Russian).
4. **Stempkovsky A. L., Gavrilov S. V., Glebov A. L.** *Metodi logicheskogo i logiko-vremennogo analiza cifrovih KMOP SBIS* (Logical and logical-timing methods of digital CMOS VLSI analysis), Moscow, Nauka, 2007, 220 p. (in Russian).

5. **Jasnjetski A., Oyeniran S. A., Tsertoy A.** High-Level Modeling and Testing of Multiple Control Faults in Digital Systems, *IEEE 19th International Symposium on Design and Diagnostics of Electronic Circuits & Systems (DDECS)*, 2016, Paper # 7482445.
6. **Bereskin A. V., Fedotova A. A., Filippov A. S.** Testipovanie cifrovih sistem, zadannih visokourovnevimi specificacijami (Testing of digital systems defined by high level specifications), *Nauchno-tehnicheskie vedomosti Sankt-Peterburgskogo gosudarstvennogo politehnicheskogo universiteta. Informatika. Telekomunikacii. Upravlenie*, 2011, vol. 6-1, no. 138, pp. 62—70 (in Russian).
7. **Jain S., Govani P., Poddar K. B., Lal A. K., Parmar R. M.** Functional verification of DSP based on-board VLSI design, *International Conference on VLSI Systems, Architectures, Technology and Applications (VLSI-SATA)*, 2016, pp. 1—4.
8. **Ubar R., Oyeniran S. A.** Multiple control fault testing in digital systems with high-level decision diagrams, *20th IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR)*, 2016, Paper # 7501287.
9. **Ivannikov V. P., Kamkin A. S., Kulyamin V. V., Petrenko A. K.** Ispolzovanie kontraktnih specificacij dlya predstavleniya trebovanij i funkcionalnogo testirovaniya modeli apparatury (Contract specifications usage for requirement presentation and functional testing of equipment models), *Programmirovaniye*, 2007, vol. 33, no. 5, pp. 47—62 (in Russian).
10. **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).
11. **Maltcev A. I.** *Algebraicheskie sistemi* (Algebraic systems), Moscow, Nauka, 1970. 392 p. (in Russian).