Обобщенная модель функционирования модульных вычислительных систем реального времени для проверки допустимости конфигураций таких систем*

А.Б. Глонина

МГУ имени М.В. Ломоносова

В работе рассматриваются системы, состоящие из вычислительных модулей, каждый из которых содержит несколько многоядерных процессоров, выполняющих вычислительные задачи в реальном времени. Автором предложена обобщенная модель функционирования таких систем, основанная на аппарате временных автоматов с остановкой таймеров. Модель позволяет для заданной конфигурации системы получить временную диаграмму ее функционирования, необходимую для проверки соблюдения директивных сроков задач. Для модели доказана детерминированность и выполнение ряда требований корректности. Программная реализация модели интегрирована с существующим средством планирования вычислений в модульных вычислительных системах.

Ключевые слова: модульные вычислительные системы, проверка корректности расписаний, временные автоматы, системы реального времени

1. Введение

В настоящее время наиболее перспективным типом архитектуры встроенных вычислительных систем является модульная архитектура. Современные модульные вычислительные системы реального времени (МВС РВ) состоят из большого количества параллельно работающих вычислительных модулей. Так автомобили премиум-класса имеют вычислительные системы архитектуры AUTOSAR, состоящие из 10–20 вычислительных модулей, содержащие в общей сложности около 100 вычислительных ядер, на которых выполняются около 800 приложений [1]. Другим примером МВС РВ являются системы интегрированной модульной авионики (ИМА), которые могут состоять из десятков модулей. На каждом модуле выполняются сотни задач, обрабатывающих большие объемы информации. Например, на борту Boeing 787 генерируется около 0,5 ТБ данных за полет, и ожидается, что на борту Airbus A350 за полет будет генерироваться около 1 ТБ данных [2]. В данной работе МВС РВ будут рассматриваться на примере систем ИМА, однако предложенные методы применимы и к другим МВС РВ.

МВС РВ состоят из стандартизированных модулей, каждый из которых содержит набор многоядерных процессоров. Один модуль может иметь процессоры нескольких типов, различающихся производительностью. Взаимодействие между модулями осуществляется посредством коммутируемых сетей с виртуальными каналами.

Рабочая нагрузка МВС РВ представляет собой набор разделов. Разделом называется группа логически связанных задач, взаимодействующих посредством общей памяти и, как правило, соответствующих одному приложению. Все задачи являются периодическими: за период должен выполниться экземпляр задачи, называемый работой. Между задачами с одинаковым периодом могут существовать зависимости по данным: очередная работа задачи-получателя не может начать выполнение до тех пор, пока не получит данные от всех соответствующих работ задач-отправителей.

Каждый раздел привязан к вычислительному ядру в составе процессора. При этом несколько разделов могут быть привязаны к одному и тому же ядру. На интервале планирования для ядра задается статическое расписание временных интервалов, называемых

*Работа выполнена при финансовой поддержке РФФИ (грант 17-07-01566 А)

окнами, в течение каждого из которых выполняются работы определенного раздела. Каждый раздел имеет собственный планировщик, управляющий выполнением работ внутри окон и работающий согласно одному из алгоритмов планирования (как правило, динамических). Допускается использование разных алгоритмов планирования для разных разделов. Одним из наиболее часто применяемых алгоритмов планирования является планирование с вытеснением на основе статических приоритетов. Каждая работа должна завершить выполнение не позднее определенного для нее директивного срока. Если директивный срок наступает до завершения выполнения работы, то она считается опоздавшей и не может далее выполняться на вычислительном ядре.

Конфигурация МВС РВ определяет набор ее модулей, характеристики рабочей нагрузки, привязку разделов к ядрам и расписания окон для каждого ядра. Конфигурация является *допустимой*, если для нее все работы полностью выполняются в рамках своих директивных сроков.

В процессе проектирования МВС РВ анализируется большое число потенциальных конфигураций и возникает задача проверки допустимости конкретной конфигурации.

Существующие аналитические методы решения этой задачи (например, [3]) не учитывают таких особенностей организации МВС РВ, как использование в разных разделах различных алгоритмов планирования. Метод построения и последующей формальной верификации модели МВС РВ позволяет преодолеть эту проблему, однако он не применим для систем большой размерности, так как вычислительная сложность этого метода экспоненциально зависит от количества задач. Так в [4] для конфигурации с 25 задачами процесс верификации занимает уже несколько минут, в то время как реальные системы содержат сотни задач, а количество проверяемых конфигураций может достигать нескольких тысяч.

Другим подходом к проверке допустимости конфигураций является построение временной диаграммы (ВД) функционирования МВС, то есть моделирование функционирования системы, в процессе которого определяются моменты постановки каждой работы на выполнение и снятия ее с вычислительного ядра. Существующие средства построения ВД функционирования вычислительных систем обладают рядом недостатков: некоторые из них не поддерживают всех особенностей МВС РВ [5, 6], некоторые не допускают автоматизации построения моделей [7], некоторые рассчитаны на решение конкретных задач, не предполагают расширения и не имеют доказательства корректности используемых моделей [8, 9]. В данной работе предлагается подход, позволяющий преодолеть перечисленные недостатки.

Автором разработана обобщенная модель функционирования МВС РВ, на основе которой по описанию конфигурации конкретной МВС РВ автоматически строится модель этой системы. Это позволяет автоматически проверять допустимость большого числа конфигураций. В качестве математического аппарата для моделирования предлагается использовать сети временных автоматов с остановкой таймеров [10]. Модели, описанные с помощью этого аппарата, позволяют получить требуемые ВД, а также для них может быть строго доказано выполнение ряда требований корректности.

В работе доказано, что из выполнения выделенного набора требований корректности к обобщенной модели следует эквивалентность всех ВД, построенных моделью, соответствующей конкретной конфигурации. Это позволяет использовать для получения ВД любой сценарий выполнения модели, что делает возможной быструю проверку допустимости конфигураций МВС, содержащих большое количество параллельно работающих компонентов.

Статья имеет следующую структуру. В разделе 2.1 формализовано понятие конфигурации МВС РВ и критерия ее допустимости. В разделе 2.2 описаны сети временных автоматов с остановкой таймеров. В разделе 2.3 приведена обобщенная формальная модель функционирования МВС РВ. В разделе 2.4 — алгоритм построения моделей конкретных МВС на основе обобщенной модели. Раздел 2.5 посвящен обоснованию корректности и детерминированности моделей. В разделе 3 описана программная реализация и апробация предложенных методов. В заключении сформулированы итоги работы.

2. Модель функционирования МВС РВ

2.1. Формальное описание МВС РВ

Конфигурация МВС РВ имеет вид $\langle HW, WL, Bind, Sched\rangle$, где:

- $HW = \{HW_i\}_{i=1}^{N}$ — множество вычислительных ядер; $Type : HW \to \overline{1, N_t}$ — тип процессора, которому принадлежит ядро ($N_t \in \mathbb{N}$ — количество типов процессоров); считается, что в каждом процессоре все ядра одинаковы;
- $WL =< Part, G >$ — рабочая нагрузка, где:
  - $Part = \{Part_i = \langle T_i, A_i\rangle\}_{i=1}^{M}$ — множество разделов, где:
    - $M \in \mathbb{N}$ — количество разделов;
    - $K_i \in \mathbb{N}$ — количество задач в i-ом разделе;
    - $T_i = \{T_{ij}\}_{j=1}^{K_i}$ — множество задач i-го раздела, каждая из которых характеризуется уникальным в рамках раздела приоритетом (pr_{ij}), временем выполнения в худшем случае (WCET) на разных типах процессоров ($\overline{C_{ij}} = (C_{ij}^1, ..., C_{ij}^{N_t})$), периодом ($P_{ij}$) и директивным сроком ($D_{ij} \le P_{ij}$);
    - A_i — тип алгоритма планирования;
  - $G = \langle \cup_{i=1}^{M} T_i, \{Msg_j\}_{j=1}^{H}\rangle$ — ациклический направленный граф зависимостей по данным, вершины которого соответствуют задачам, а дуги — сообщениям; каждое сообщение Msg_j характеризуется максимальными длительностями передачи через память модуля и через сеть (DM_j и DN_j соответственно);
- $Bind : Part \to HW$ — привязка разделов к ядрам;
- $Sched = \{\{\langle Start_{ij}, Stop_{ij}\rangle\}_{j=1}^{N_i^w}\}_{i=1}^{M}$ — расписание окон разделов, заданное на интервале планирования L, где L равно наименьшему общему кратному всех P_{ij}; $N_i^w \in \mathbb{N}$ — количество окон для i-го раздела; $Start_{ij}, Stop_{ij} \in \overline{0, L}$ — времена начала и завершения j-го окна i-го раздела.

В данной работе предполагается, что время дискретно: все времена и длительности, входящие в состав конфигурации, задаются в квантах. Квант равен наибольшему общему делителю тактов процессоров. Пусть $CONF$ — множество всевозможных конфигураций МВС РВ.

Далее введем понятие временной диаграммы (ВД) функционирования МВС РВ.

Для каждой задачи T_{ij} на интервале планирования определен набор работ $W_{ij} = \{w_{ijk}\}_{k=1}^{L/P_{ij}}$. $D_{ijk} = P_{ij} * (k-1) + D_{ij}$ — директивный срок k-й работы.

Пусть $e = \langle Type, Src, t\rangle$ — событие, где $Type \in \{EX, PR, FIN\}$ — тип события: постановка работы на выполнение (EX); вытеснение работы с ядра (PR); завершение работы, связанное с полным выполнением работы, либо с наступлением директивного срока (FIN). $Src \in \bigcup\limits_i \bigcup\limits_j W_{ij}$ — работа-источник события. $t \in \{n\}_{n=1}^{L}$ — время.

$E = \{EX, PR, FIN\} \times \bigcup\limits_i \bigcup\limits_j W_{ij} \times \{n\}_{n=1}^{L}$ — множество всевозможных событий.

$A = A_1, ..., A_M$ — совокупность алгоритмов планирования, используемых в разделах.

На практике [9,11] при проектировании МВС РВ предполагают, что время выполнения каждой работы на соответствующем ядре фиксировано и равно наихудшему, время передачи каждого сообщения также фиксировано и равно наихудшему, а алгоритм планирования детерминирован. В таких предположениях совокупность алгоритмов планирования A *однозначно* определяет отображение $Q : CONF \to 2^E$.

ВД для конфигурации $conf \in CONF$ — подмножество множества E, однозначно соответствующее $conf$, и являющееся результатом интерпретации A на $conf$.

Пусть $conf \in CONF$ — некоторая конфигурация системы, а $Q(conf)$ — ВД, соответствующая $conf$. Пусть R_{ijk} – количество интервалов выполнения работы w_{ijk}. Тогда упорядоченный набор событий для w_{ijk} имеет вид:

- $\emptyset$, если $R_{ijk} = 0$ (работа не была поставлена на выполнение);
- $\{\langle w_{ijk}, EX, t_0\rangle, \langle w_{ijk}, FIN, t_1\rangle\} \subseteq Q(conf)$, если $R_{ijk} = 1$ (работа была поставлена на выполнение в момент t_0, ни разу не вытеснялась и завершилась в момент t_1);
- $\{\langle w_{ijk}, EX, t_0\rangle, \langle w_{ijk}, PR, t_1\rangle, ..., \langle w_{ijk}, EX, t_{2R_{ijk}-2}\rangle, \langle w_{ijk}, FIN, t_{2R_{ijk}-1}\rangle\} \subseteq Q(conf)$, если $R_{ijk} > 1$, (работа была поставлена на выполнение в момент t_0, несколько раз вытеснялась и завершилась в момент $t_{2R_{ijk}-1}$).

Таким образом, критерий допустимости конфигурации, заключающийся в том, что все работы должны быть полностью выполнены в рамках своих директивных сроков, имеет следующий формальный вид:

$$CR(conf) : \forall w_{ijk}, i \in \overline{1, M}, j \in \overline{1, K_i}, k \in \overline{1, L/P_{ij}}\text{: } R_{ijk} \geq 1 \text{ и } \sum_{r=1}^{R_{ijk}} (t_{2r-1} - t_{2r-2}) = C_{ij}^{Type(Bind(Part_i))}$$

В работе решается задача построения отображения $Q : CONF \to 2^E$, позволяющего для данной конфигурации МВС РВ получить ВД ее функционирования, необходимую для проверки критерия CR допустимости конфигурации. Для построения искомого отображения Q будем строить модель функционирования МВС РВ.

2.2. Сети временных автоматов с остановкой таймеров

К математическому аппарату для построения модели функционирования МВС РВ были сформулированы следующие требования: 1) возможность получения ВД для заданной конфигурации; 2) возможность формализации и проверки выполнения требований корректности к модели; 3) наличие программных средств для моделирования и проверки выполнения требований.

Автором был рассмотрен ряд математических аппаратов [10, 12–18] и из них были выбраны сети временных автоматов с остановкой таймеров [10], так как они в наибольшей степени удовлетворяют перечисленным требованиям.

Автомат с остановкой таймеров — это конечный автомат с целочисленными переменными и таймерами. Графически такой автомат может быть представлен размеченным ориентированным графом, вершины которого называются локациями, а дуги — переходами. Состоянием автомата называется совокупность его текущей локации, значений переменных и таймеров.

Таймер — это специальная переменная, принимающая вещественные значения. Начальное значение каждого таймера равно нулю. Таймер считается активным, если его условие активности (булево выражение над множеством переменных) в текущей локации автомата истинно. В противном случае таймер считается остановленным.

Для автомата указывается начальная локация. Каждой локации сопоставлен инвариант — булево выражение над таймерами и переменными. Автомат может оставаться в данной локации до тех пор, пока инвариант этой локации имеет истинное значение.

Каждый переход характеризуется условием перехода, действиями и, возможно, синхронизацией. Переход активен, если автомат находится в локации, предшествующей этому переходу, значения переменных и таймеров удовлетворяют условию перехода, и инвариант локации-назначения перехода имеет истинное значение. Переход может (но не обязан) быть совершен, если он активен и может быть выполнена синхронизация. При совершении перехода меняется текущая локация автомата, происходит синхронизация и выполняются действия (изменение переменных и обнуление таймеров).

Состояние автомата может измениться не только в результате выполнения перехода, но и в результате синхронного увеличения значений всех таймеров. Значения всех таймеров, кроме остановленных, могут быть увеличены на некоторую вещественную величину d, если $\forall d' : 0 < d' \leq d$ значения таймеров, увеличенные на d', удовлетворяют инварианту текущей локации. В некоторых локациях, называемых срочными, продвижение времени невозможно.

Сеть автоматов представляет собой набор автоматов, функционирующих синхронно и взаимодействующих посредством общих переменных и каналов.

Канал — средство синхронизации автоматов. Синхронизация характеризуется именем канала и действием. Существует два парных действия: отправка и прием сигнала по каналу. Также существует два типа каналов — «точка-точка» и широковещательные. Если в некоторый момент времени в двух автоматах сети активны переходы с парными действиями синхронизации по одному и тому же каналу «точка-точка», то эти переходы выполняются одновременно. $N+1$ автомат могут синхронизироваться посредством широковещательного канала, если в одном автомате активен переход с отправкой сигнала по этому каналу, а в остальных N— с приемом сигнала.

Формально автомат — это кортеж $A = \langle L, U, l_0, C, V, \overline{v_0}, AA, AS, E, I, P\rangle$, где:

- L, $U \subseteq L$, $l^0 \in L$ — множества локаций и срочных локаций, начальная локация;
- C — множество таймеров;
- V, $\overline{v_0}$ — множество переменных и их начальные значения;
- AA, AS — множество действий над таймерами и переменными и множество синхронизаций;
- E — множество переходов, $E \subseteq L \times B(C,V) \times AS \times AA \times L$. $B(C,V)$ — множество всевозможных условий над C и V;
- $I : L \to B(C,V)$ — назначение инвариантов;
- $P : (L \times C) \to B(\varnothing, V)$ — назначение условий активности таймеров.

Пусть $A = A_1, ..., A_n$ — сеть автоматов, где $A_i = \langle L_i, U_i, l_i^0, C_i, V_i, \overline{v_i^0}, AU_i, AS_i, E_i, I_i, P_i\rangle$. Состоянием сети называется кортеж $\langle \bar{l}, \bar{c}, \bar{v}\rangle \in (L_1 \times ... \times L_n) \times \mathbb{R}_{\geq 0}^{|C|} \times \mathbb{Z}^{|V|}$, где $V = \bigcup\limits_{i=1}^{n} V_i$, $C = \bigcup\limits_{i=1}^{n} C_i$. Конечная или бесконечная последовательность состояний, получаемая при некотором сценарии функционирования сети автоматов, называется *вычислением* этой сети.

2.3. Обобщенная формальная модель функционирования МВС РВ

Введем ряд определений, необходимых для описания предложенной модели.

Параметризованный автомат с остановкой таймеров, или *конкретный тип автоматов* — кортеж $\langle L, U, l_0, C, V, \overline{p}, AU, AS, E, I, P\rangle$, где $\overline{p}$ — целочисленные параметры с не заданными значениями. *Интерфейс автомата* — набор переменных и синхронизаций, посредством которых автомат взаимодействует с другими автоматами. *Базовый тип автоматов* определяется интерфейсом, то есть парой $\langle V_b, AS_b\rangle$, где V_b — множество переменных, AS_b — множество синхронизаций. Конкретный тип автоматов *реализует* базовый тип, если $V_b \subseteq V$, $AS_b \subseteq AS$.

Набор базовых типов автоматов с определенными между ними логическими связями назовем *обобщенной сетью автоматов*. Под логическими связями понимается спецификация того, какие базовые типы автоматов могут взаимодействовать и с помощью каких средств (переменных и синхронизаций). *Конкретная сеть автоматов* — набор конкретных типов автоматов с определенными между ними логическими связями. Конкретная сеть автоматов *реализует* обобщенную сеть автоматов, если каждый базовый тип автоматов обобщенной

сети реализуется одним или несколькими конкретными типами автоматов конкретной сети, и логические связи между типами автоматов в базовой сети соответствуют логическим связям в конкретной сети.

Модельное время — значение специального таймера, который никогда не обнуляется, и условие активности которого всегда истинно.

Событие синхронизации — тройка $\langle CH, A, t\rangle$, где CH — канал, A — набор автоматов, участвующих в синхронизации, t — модельное время. *Временная диаграмма (ВД) сети автоматов* — набор событий синхронизации для некоторого вычисления сети.

Автором предложено представление обобщенной модели функционирования МВС РВ в виде введенной выше обобщенной сети автоматов с остановкой таймеров.

В разработанной обобщенной сети имеются следующие средства взаимодействия автоматов (см. обозначения, введенные в разделе 2.1):

- переменные $\texttt{is\_ready}_{ij}$, $\texttt{is\_failed}_{ij}$, $i \in \overline{1,M}, j \in \overline{1,K_i}$, $\texttt{is\_data\_ready}_h$, $h \in \overline{1,H}$, соответствующие готовности опозданию работы задачи T_{ij} и доставке сообщения по h-му виртуальному каналу;
- переменные $\texttt{prio}_{ij}$, $\texttt{deadline}_{ij}$, $i \in \overline{1,M}, j \in \overline{1,K_i}$, значения которых равны приоритету и директивному сроку задачи T_{ij} и не изменяются в процессе функционирования сети;
- каналы $\texttt{wakeup}_i$, $\texttt{sleep}_i$, $i \in \overline{1,M}$, соответствующие началу и завершению очередного окна i-го раздела;
- каналы $\texttt{ready}_i$, $\texttt{finished}_i$, $\texttt{exec}_{ij}$, $\texttt{preempt}_{ij}$ $i \in \overline{1,M}$, $j \in \overline{1,K_i}$, соответствующие готовности, завершению (вследствие окончания выполнения, либо наступления директивного срока), постановке на выполнение и вытеснению некоторой работы;
- широковещательные каналы $\texttt{send}_{ij}$, $\texttt{receive}_{ij}$, $i \in \overline{1,M}$, $j \in \overline{1,K_i}$, соответствующие отправке сообщения работой-отправителем и получению сообщения работой-получателем.

Обобщенная сеть автоматов состоит из следующих базовых типов автоматов.

1. Базовый тип автоматов `T`, моделирующий функциональную задачу и определенный следующим интерфейсом:

- получение сигналов по каналам `exec` и `preempt` и отправка сигналов по каналам `ready` и `finished`;
- отправка сигналов по широковещательному каналу `send` и получение сигналов по широковещательному каналу `receive`;
- изменение переменных `is_ready`, `is_failed` и $\texttt{is\_data\_ready}_h$, где h — номер виртуального канала, по которому работы данной задачи получают сообщения.

2. Базовый тип автоматов `TS`, моделирующий планировщик работ раздела и определенный следующим интерфейсом:

- получение сигналов по каналам `wakeup`, `sleep`, `ready` и `finished`;
- отправка сигналов по каналам $\texttt{exec}_j$, $\texttt{preempt}_j$, где j-й канал соответствует j-й задаче раздела;
- чтение значений переменных $\texttt{is\_ready}_j$, $\texttt{prio}_j$, $\texttt{deadline}_j$.

3. Базовый тип автоматов `CS`, моделирующий планировщик ядра. Планировщик ядра переключает окна разделов согласно статическому расписанию, являющемуся элементом конфигурации МВС РВ. `CS` определен следующим интерфейсом:

- отправка сигналов по каналам $\texttt{wakeup}_i$ и $\texttt{sleep}_i$, где j-й канал соответствует j-му разделу;

4. Базовый тип автоматов `L`, моделирующий виртуальный канал и определенный следующим интерфейсом:

- получение сигналов по широковещательному каналу `send` и отправка сигналов по широковещательному каналу `receive`;
- изменение переменной `is_data_ready`.

Структура предложенной обобщенной сети автоматов, отражающая логические связи между ее компонентами, показана на рисунке 1.

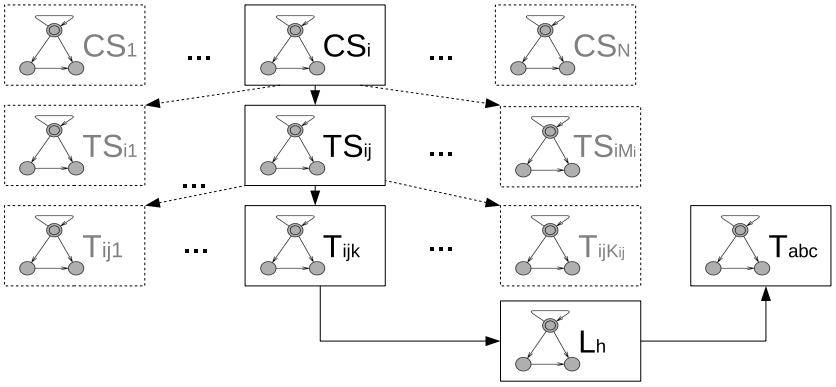

Рис. 1. Структура обобщенной сети автоматов, моделирующей функционирование МВС РВ.

2.4. Проверка допустимости конфигураций конкретных МВС РВ

Конкретная сеть автоматов, реализующая предложенную обобщенную сеть автоматов является параметризованной моделью функционирования МВС РВ, параметрами которой являются элементы конфигурации системы. Эта модель содержит следующие конкретные типы автоматов: модели задачи, планировщика ядра, виртуального канала и нескольких планировщиков работ, работающих согласно различным алгоритмам планирования (с фиксированными и динамическими приоритетами, с вытеснением и без вытеснения).

Для данной конкретной сети автоматов и конфигурации МВС РВ $conf \in CONF$ экземпляр сети автоматов, моделирующий функционирование системы, может быть построен согласно следующему алгоритму:

- цикл по ядрам ($i \in [1, N]$):
  - цикл по разделам ядра ($j \in [1, M] : Bind(Part_j) = HW_i$):
    - создание каналов $\texttt{ready}_j$, $\texttt{finished}_j$, $\texttt{wakeup}_j$, $\texttt{sleep}_j$;
    - цикл по задачам ($k \in [1, K_j]$):
      — создание каналов $\texttt{exec}_{jk}$, $\texttt{preempt}_{jk}$, $\texttt{send}_{jk}$, $\texttt{receive}_{jk}$;
      — создание переменных $\texttt{is\_ready}_{jk}$, $\texttt{prio}_{jk}$, $\texttt{deadline}_{jk}$ и их инициализация соответствующими значениями из $conf$;
      — создание переменных $\texttt{is\_data\_ready}_h$, соответствующих виртуальным каналам, для которых задача T_{jk} является получателем, и инициализация их значением 0;
      — порождение автомата конкретного типа, реализующего базовый тип T (моделирующего задачу), инициализация его интерфейса каналами $\texttt{exec}_{jk}$,

$\texttt{preempt}_{jk}$, $\texttt{send}_{jk}$, $\texttt{receive}_{jk}$, $\texttt{ready}_j$, $\texttt{finished}_j$ и переменными $\texttt{is\_ready}_{jk}$, $\texttt{prio}_{jk}$, $\texttt{deadline}_{jk}$, $\texttt{is\_data\_ready}_h$ и инициализация его параметров соответствующими значениями из $conf$;

  - порождение автомата конкретного типа, реализующего базовый тип TS (моделирующего планировщик работ) и соответствующего алгоритму планирования A_j j-го раздела, инициализация его интерфейса каналами $\texttt{exec}_{jk}$, $\texttt{preempt}_{jk}$, $\texttt{ready}_j$, $\texttt{finished}_j$, $\texttt{wakeup}_j$, $\texttt{sleep}_j$ и переменными $\texttt{is\_ready}_{jk}$, $\texttt{prio}_{jk}$, $\texttt{deadline}_{jk}$ ($k \in [1, K_j]$);

- порождение автомата конкретного типа, реализующего базовый тип CS (моделирующего планировщик ядра), инициализация его интерфейса каналами $\texttt{wakeup}_j$, $\texttt{sleep}_j$ и инициализация его параметров соответствующими значениями из $Sched$ ($j \in [1, M] : Bind(Part_j) = HW_i$);

- цикл по виртуальным каналам ($h \in [1, H]$):

  - порождение автомата конкретного типа, реализующего базовый тип L (моделирующего виртуальный канал), инициализация его интерфейса соответствующими каналами $\texttt{send}_{j_1k_1}$, $\texttt{receive}_{j_1k_1}$ и переменной $\texttt{is\_data\_ready}_h$ и инициализация его параметров соответствующими значениями из $conf$.

Таким образом, МВС РВ с фиксированной конфигурацией взаимно однозначно (в рамках предложенной модели) соответствует моделирующая ее сеть автоматов.

По ВД сети автоматов, построенной для некоторой конфигурации, может быть получена ВД МВС РВ, необходимая для проверки критерия допустимости данной конфигурации. Каждому событию синхронизации $\langle CH(AS_{sync}), A_{sync}, t_{sync}\rangle$ из ВД сети автоматов, такому что $CH(AS_{sync}) \in \{\texttt{exec}_{jk}, \texttt{preempt}_{jk}, \texttt{finished}_j\}$ однозначно ставится в соответствие событие $\langle Type, Src, t\rangle$ ВД МВС РВ по следующим правилам:

- тип события (EX, PR, FIN) соответствует каналу ($\texttt{exec}_{jk}$, $\texttt{preempt}_{jk}$, $\texttt{finished}_j$);

- работа-источник события определяется автоматом-моделью задачи, участвующим в синхронизации, и временем синхронизации: $Src = w_{ijk}$, где $i, j : A^l_{sync} = T_{ij}$, $k = \lfloor t_{sync}/P_{ij} \rfloor + 1$ (для EX и PR) и $k = \lceil t_{sync}/P_{ij} \rceil$ (для FIN);

- время события равно времени синхронизации: $t = t_{sync}$;

2.5. Корректность и детерминированность модели

В данном разделе приведен подход к доказательству корректности и детерминированности всех моделей, построенных на основе предложенной обобщенной модели.

Спецификации МВС РВ содержат ряд требований корректности к функционированию системы в целом и ее компонентов. Эти требования представляют собой ограничения на последовательности событий, происходящих в системе, и на длительности интервалов между определенными типами событий. Для того, чтобы гарантировать корректность проверки критерия допустимости конфигураций, используемая для этого модель должна удовлетворять требованиям корректности к целевой системе, применимым на выбранном для моделирования уровне абстракции.

Назовем модель МВС РВ *детерминированной*, если ВД, получаемая при ее выполнении, определяется однозначно. Детерминированность моделей крайне важна при проектировании МВС РВ большой размерности, так как из нее следует то, что для проверки допустимости некоторой конфигурации достаточно получить и анализировать одно любое вычисление сети автоматов, моделирующей МВС РВ с данной конфигурацией. В этом состоит отличие предложенного метода от верификации модели системы [4], в процессе которой анализируются все возможные вычисления сети автоматов и которая имеет неприемлемую для исследования систем большой размерности вычислительную сложность.

Выбранный математический аппарат моделирования позволяет проверять выполнение требований к компонентам модели автоматически с помощью верификатора. Для этого используется подход «автоматов-наблюдателей» [19].

«Автомат-наблюдатель» — это автомат, работающий синхронно с исходным автоматом, и соответствующий, как правило, одному требованию из спецификации МВС. У автомата-наблюдателя существует «плохая» локация, соответствующая некорректным последовательностям синхронизаций в исходном автомате. Во всех локациях «наблюдателя» активны переходы с действиями синхронизации, парными ко всем возможным действиям синхронизации исходного автомата. Достижимость «плохой» локации означает, что в исходном автомате может произойти некорректная последовательность синхронизаций, соответствующая некорректной последовательности событий в МВС РВ.

В данной работе требуется проверить корректность построенных конкретных типов автоматов, то есть параметризованных автоматов, так как это необходимо для доказательства корректности всех моделей, синтезируемых на основе обобщенной модели и содержащих экземпляры автоматов этих конкретных типов. Таким образом, нужно убедиться в том, что для каждого конкретного типа автоматов некорректные последовательности синхронизаций невозможны ни при каких значениях его параметров. Поэтому в «автомате-наблюдателе» значения параметров выбираются недетерминированно. Кроме того, если исходный автомат использует переменные, изменяемые другими автоматами, то «наблюдатель» меняет значения этих переменных недетерминированно.

Для проверки корректности компонента параметризованной модели необходимо выделить из спецификаций МВС РВ применимые к нему на выбранном уровне абстракции требования, для каждого из них построить «автомат-наблюдатель» и проверить недостижимость его «плохой» локации. Такая проверка была выполнена автором для всех разработанных конкретных типов автоматов и требований, выделенных из стандарта ARINC653 [20] на системы ИМА. Проверка проводилась с использованием верификатора UPPAAL [13].

Рассмотрим следующий пример требования:

Для любого раздела системы верно, что в каждый момент времени в нем выполняется не более одной работы.

Это требование относится к базовому типу автоматов `TS`, моделирующему планировщик работ, и должно быть выполнено для всех конкретных типов автоматов, реализующих `TS`.

В терминах синхронизаций автоматов выполнением работ задачи T_{jk} являются интервалы между синхронизациями по каналам $\texttt{exec}_{jk}$ и либо $\texttt{preempt}_{jk}$, либо $\texttt{finished}_j$. Сформулированное требование означает, что за любой синхронизацией по каналу $\texttt{exec}_{jk}$ должна следовать синхронизация по либо по каналу $\texttt{preempt}_{jk}$, либо по каналу $\texttt{finished}_j$. «Автомат-наблюдатель», соответствующий данному требованию, приведен на рисунке 2.

Проверка требований к параметризованной модели МВС РВ в целом не может быть выполнена автоматически, так как количество автоматов, входящих в модель, также является ее параметром и в общем случае неизвестно. Поэтому доказательство выполнения таких требований корректности было выполнено вручную посредством строгих логических рассуждений и основано на доказанном ранее выполнении требований корректности к отдельным компонентам модели.

Рассмотрим пример требования корректности к модели в целом и его доказательство:

Для любого ядра системы верно, что в каждый момент времени на нем выполняется не более одной работы.

Выполнение следующих требований к компонентам модели было автоматически доказано с использованием соответствующих «автоматов-наблюдателей»:

1. Для любого раздела системы верно, что в каждый момент времени в нем выполняется не более одной работы.

2. Для любого ядра системы верно, что в каждый момент времени на нем выполняется не более одной работы.

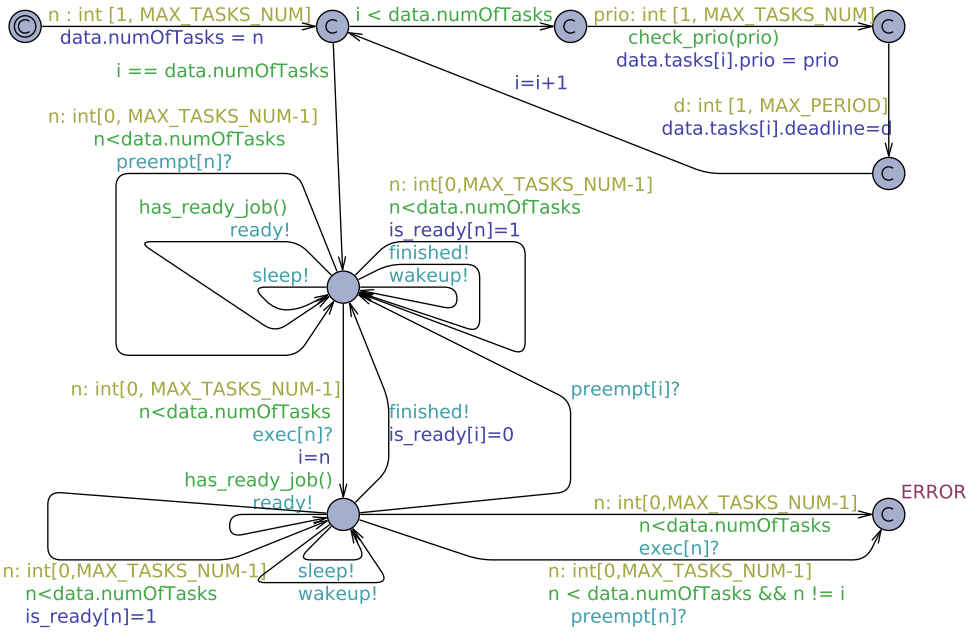

Рис. 2. Автомат-наблюдатель для требования к модели планировщика работ.

Из выполнения этих требований к компонентам модели и следует выполнение сформулированного требования к модели в целом.

Доказательство детерминированности модели основывается на ранее доказанном выполнении требований корректности к модели. Приведем подход к этому доказательству.

Предположим, что модель не является детерминированной, то есть для экземпляра сети автоматов, соответствующей некоторой конфигурации, могут быть получены две разные ВД. Упорядочим события этих двух ВД по времени. Так как события в системе могут происходить одновременно, то к каждой временной точке ВД привязано некоторое множество событий. Пусть t_i — точка с наименьшим модельным временем, для которой множества событий двух ВД различаются. Это означает, что как минимум одно событие синхронизации присутствует в множестве событий для этой точки одной трассы и отсутствует в соответствующем множестве другой. Предположим, что это синхронизация по каналу $\texttt{finished}_j$. Так как для всех предыдущих временных точек множества событий совпадают, то возможно два варианта:

1. Согласно трассе, содержащей данное событие, некоторая работа выполнялась на ядре в общей сложности WCET единиц времени. Из этого следует, что, согласно второй трассе, в которой данного события нет, эта работа выполнялась на ядре в общей сложности более, чем WCET единиц времени.

2. Согласно трассе, содержащей данное событие, некоторая работа была снята с ядра по причине наступления ее директивного срока. Из этого следует, что согласно второй трассе, в которой данного события нет, эта работа осталась на ядре после наступления ее директивного срока.

Оба варианта противоречат доказанным ранее требованиям корректности. Для синхронизаций по другим типам каналов проводятся аналогичные рассуждения и делается вывод о том, что сформулированное предположение неверно.

Таким образом, доказано, что между конфигурацией МВС и ее моделью существует взаимно однозначное соответствие, модель детерминирована и корректна, между ВД модели и ВД МВС также существует взаимно однозначное соответствие. Следовательно, проверка критерия допустимости конфигураций с помощью предложенной модели корректна.

3. Программная реализация и апробация

Для апробации предложенного подхода автором была создана программная реализация разработанных моделей и методов.

Конкретные типы автоматов, моделирующие компоненты системы, были разработаны и верифицированы с использованием средства UPPAAL [13]. Проверифицированные автоматы были помещены в библиотеку автоматных моделей компонентов МВС РВ. При необходимости библиотека может быть пополнена пользовательскими моделями при условии выполнения для них требований корректности.

В открытых источниках не было найдено средства, позволяющего эффективно моделировать функционирование сетей временных автоматов с остановкой таймеров. Поэтому на языке C++ была реализована собственная библиотека моделирования таких автоматов. Также был реализован транслятор моделей, описанных на языке UPPAAL, в представление на языке C++. Разработанная библиотека автоматных моделей была преобразована с помощью транслятора в библиотеку программных моделей, формирующих параметризованную программную модель функционирования МВС (см. рисунок 3).

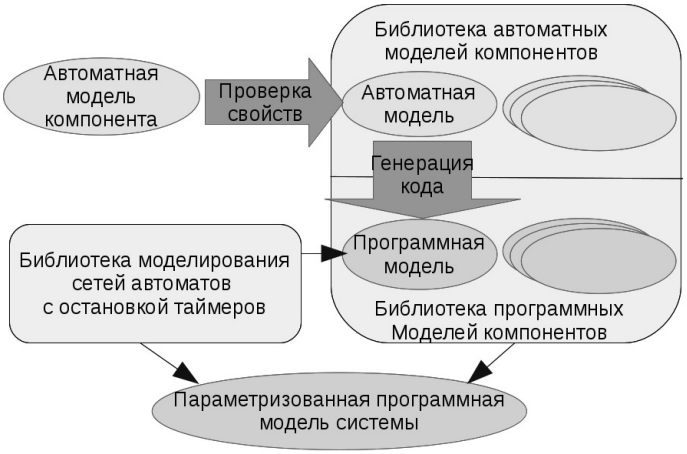

Рис. 3. Формирование параметризованной модели функционирования МВС

Для проверки предположения о большей эффективности предложенного подхода по сравнению с верификацией моделей МВС РВ была проведена серия экспериментов. В каждом эксперименте для фиксированной конфигурации МВС РВ на основе разработанных моделей компонентов строилась модель системы. В результате прогона полученной модели генерировалась ВД и с ее использованием проверялся критерий допустимости. Также проводилась верификация модели: проверялось, что ни для одного вычисления сети автоматов ни одна из переменных is_failed не принимает значение 1. Результаты сравнения времени проверки допустимости конфигурации с использованием ВД с временем верификации соответствующей модели приведены в табл. 1 и подтверждают эффективность предложенного подхода.

Таблица 1. Время проверки допустимости конфигураций с различным количеством работ (сек.)

Количество работ	10	11	12	13	14	15	16	17	18
Верификация	0.57	1.16	2.22	5.05	10.43	23.51	48.13	112.28	215.91
Использование ВД	0.027	0.027	0.028	0.030	0.031	0.032	0.033	0.035	0.036

Разработанная модель была интегрирована с САПР ИМА, решающей одну из задач

планирования вычислений [9]. На каждой итерации работы алгоритма планирования формируется конфигурация, для которой необходимо проверить критерий допустимости. Для конфигурации генерируется XML-файл с ее описанием и передается на вход параметризованной модели. По этому описанию строится и запускается экземпляр модели. В результате его выполнения генерируется ВД и в виде XML-файла передается в САПР ИМА, где производится проверка критерия допустимости. Эксперименты показали, что на данных, приближенных к реальным, генерация и прогон модели занимают несколько секунд (11 секунд для конфигурации с 12500 работами на интервале планирования), что свидетельствует о применимости предложенного подхода на практике.

4. Заключение

В работе предложена обобщенная модель функционирования МВС РВ, основанная на математическом аппарате сетей временных автоматов с остановкой таймеров и позволяющая строить на своей базе конкретные модели МВС для оценки допустимости конфигураций таких систем. Для класса моделей, синтезируемых на основе обобщенной модели, доказана детерминированность и выполнение ряда требований корректности. Из детерминированности модели следует то, что для проверки допустимости конфигурации может быть использовано любое вычисление сети автоматов. Поэтому предложенный подход значительно эффективнее подхода, заключающегося в верификации модели МВС РВ и использующего аналогичный математический аппарат, что позволяет применять ее для проверки допустимости реальных систем большой размерности, что подтверждено экспериментально. Возможными направлениями дальнейших исследований являются расширение библиотеки моделей компонентов МВС РВ, интеграция с другими средствами проектирования МВС РВ и применение подхода к другим классам распределенных систем.

Литература

1. Senthilkumar K.,Ramadoss R. Designing multicore ECU architecture in vehicle networks using AUTOSAR // Proceedings of the Third International Conference on Advanced Computing, 2011, Chennai, India. P. 270–275. DOI: 10.1109/ICoAC.2011.6165187

2. From the Ground Up: How the internet of things will give rise to connected aviation. Gogo LLC, 2016. 140 p.

3. Marinescu, S. Tamas-Selicean D., Acretoaie V., et al. Timing analysis of mixed-criticality hard real-time applications implemented on distributed partitioned architectures // Proceedings of 2012 17th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2012), 2012, Krakow, Poland. P. 1–4. DOI: 10.1109/ETFA.2012.6489720

4. Macariu, G., Cretu, V. Timed automata model for component-based real-time systems // Proceedings of 2010 17th IEEE International Conference and Workshops on Engineering of Computer Based Systems, 2010, Oxford, UK. P. 121–130. DOI: 10.1109/ECBS.2010.20

5. Craveiro, J. P., Silveira, R. O., Rufino, J. HsSim: an extensible interoperable object-oriented n-level hierarchical scheduling simulator // Proceedings of the 3rd International Workshop on Analysis Tools and Methodologies for Embedded and Real-time Systems (WATERS 2012), 2012, Pisa, Italy P. 9–14.

6. Singhoff, F., Plantec, A., Dissaux, P., et al. Investigating the usability of real-time scheduling theory with the Cheddar project // Real-Time Systems. 2009. Vol. 43, No. 3 P. 259–295. DOI: 10.1007/s11241-009-9072-y

7. Khoroshilov, A. Albitskiy, D., Koverninskiy, I., et al. AADL-Based Toolset for IMA System Design and Integration. // SAE Int. J. Aerosp. 2012. Vol. 5, No. 2 P. 294–299. DOI: 10.4271/2012-01-2146

8. Dissaux, P., Marc, O., Fotsing, C., et al. The SMART project: Multi-agent scheduling simulation of real-time architectures // Embedded Real Time Software and Systems. 2014.

9. Balashov, V.V., Balakhanov, V.A., Kostenko, V.A. Scheduling of computational tasks in switched network-based IMA systems // Proceedings of International Conference on Engineering and Applied Sciences Optimization, 2014, Athens, Greece. P. 1001–1014.

10. Cassez, F., Larsen, K. The impressive power of stopwatches // CONCUR 2000 — Concurrency Theory. LNCS. Springer, 2000. Vol. 1877. P. 138–152. DOI: 10.1007/3-540-44618-4_12

11. Tretyakov, A. Automation of scheduling for periodic real-time systems // Proceedings of the Institute for System Programming. 2012. Vol 22. P.375–400. DOI: 10.15514/ISPRAS-2012-22-20

12. Smelyansky, R.L. Model of distributed computing system operation with time // Programming and Computer Software, 2013. Vol. 39, No. 5, P. 233–241. DOI: 10.1134/S0361768813050046

13. Bengtsson, J., Yi W. Timed automata: Semantics, algorithms and tools // Lectures on Concurrency and Petri Nets. LNCS. Springer, 2004. Vol. 3098, P. 87–124. DOI: 10.1007/978-3-540-27755-2_3

14. Zuberek, W.M. Timed Petri nets definitions, properties and applications // Microelectronics Reliability, 1991. Vol. 31, No. 4. P. 627–644 DOI: 10.1016/0026-2714(91)90007-T

15. Boudjadar, A J., Kim, J.H., Larsen, K.G., et al. Model Checking Process Algebra of Communicating Resources for Real-time Systems // Proceedings of the 26th Euromicro Conference on Real-Time Systems (ECRTS 2014), 2014, Madrid, Spain. P. 51–60. DOI: 10.1109/ECRTS.2014.24

16. Lime D., Roux O.H., Seidner C., et al. Romeo: A Parametric Model-Checker for Petri Nets with Stopwatches // Tools and Algorithms for the Construction and Analysis of Systems. TACAS 2009. LNCS. Springer, 2009. Vol. 5505, P. 54–57 DOI: 10.1007/978-3-642-00768-2_6

17. Henzinger, T. The Theory of Hybrid Automata // Verification of Digital and Hybrid Systems. NATO ASI Series. Springer, 2000. Vol. 170. P. 265–292 DOI: http://dx.doi.org/10.1007/978-3-642-59615-5_1

18. Бусленко Н.П. Моделирование сложных систем. М.: Наука, 1978. 400 с.

19. Andre E. Observer patterns for real-time systems // Proceedings of 2013 18th IEEE International Conference on Engineering of Complex Computer Systems (ICECCS), 2013, Singapore. P. 125–134 DOI: 10.1109/ICECCS.2013.26

20. Avionics application software standard interface. ARINC specification 653 // Aeronautical Radio. Annapolis, 1997.

General Model of Real-Time Modular Computer Systems Operation for Checking Acceptability of Such Systems Configurations

A.B. Glonina

Lomonosov Moscow State University

In this paper we consider computer systems consisting of modules, each of which contains several multicore processors executing real time computational tasks. The author proposes a stopwatch automata-based general model of such systems operation. The model provides an ability to obtain an operation trace for a given system configuration. Such traces are necessary for configurations schedulability analysis. It is proven that the model is deterministic and satisfies a set of correctness requirements. The software implementation of the proposed model is integrated with the existing modular computer systems scheduling tool.

Keywords: modular computer systems, schedulability analysis, stopwatch automata, simulation, real-time systems

References

1. Senthilkumar K.,Ramadoss R. Designing multicore ECU architecture in vehicle networks using AUTOSAR // Proceedings of the Third International Conference on Advanced Computing, 2011, Chennai, India. P. 270–275. DOI: 10.1109/ICoAC.2011.6165187

2. From the Ground Up: How the internet of things will give rise to connected aviation. Gogo LLC, 2016. 140 p.

3. Marinescu, S. Tamas-Selicean D., Acretoaie V., et al. Timing analysis of mixed-criticality hard real-time applications implemented on distributed partitioned architectures // Proceedings of 2012 17th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2012), 2012, Krakow, Poland. P. 1–4. DOI: 10.1109/ETFA.2012.6489720

4. Macariu, G., Cretu, V. Timed automata model for component-based real-time systems // Proceedings of 2010 17th IEEE International Conference and Workshops on Engineering of Computer Based Systems, 2010, Oxford, UK. P. 121–130. DOI: 10.1109/ECBS.2010.20

5. Craveiro, J. P., Silveira, R. O., Rufino, J. HsSim: an extensible interoperable object-oriented n-level hierarchical scheduling simulator // Proceedings of the 3rd International Workshop on Analysis Tools and Methodologies for Embedded and Real-time Systems (WATERS 2012), 2012, Pisa, Italy P. 9–14.

6. Singhoff, F., Plantec, A., Dissaux, P., et al. Investigating the usability of real-time scheduling theory with the Cheddar project // Real-Time Systems. 2009. Vol. 43, No. 3 P. 259–295. DOI: 10.1007/s11241-009-9072-y

7. Khoroshilov, A. Albitskiy, D., Koverninskiy, I., et al. AADL-Based Toolset for IMA System Design and Integration // SAE Int. J. Aerosp. 2012. Vol. 5, No. 2 P. 294–299. DOI: 10.4271/2012-01-2146

8. Dissaux, P., Marc, O., Fotsing, C., et al. The SMART project: Multi-agent scheduling simulation of real-time architectures // Embedded Real Time Software and Systems. 2014.

9. Balashov, V.V., Balakhanov, V.A., Kostenko, V.A. Scheduling of computational tasks in switched network-based IMA systems // Proceedings of International Conference on Engineering and Applied Sciences Optimization, 2014, Athens, Greece. P. 1001–1014.

10. Cassez, F., Larsen, K. The impressive power of stopwatches // CONCUR 2000 — Concurrency Theory. LNCS. Springer, 2000. Vol. 1877. P. 138–152. DOI: 10.1007/3-540-44618-4_12

11. Tretyakov, A. Automation of scheduling for periodic real-time systems // Proceedings of the Institute for System Programming. 2012. Vol 22. P.375–400. DOI: 10.15514/ISPRAS-2012-22-20

12. Smelyansky, R.L. Model of distributed computing system operation with time // Programming and Computer Software, 2013. Vol. 39, No. 5, P. 233–241. DOI: 10.1134/S0361768813050046

13. Bengtsson, J., Yi W. Timed automata: Semantics, algorithms and tools // Lectures on Concurrency and Petri Nets. LNCS. Springer, 2004. Vol. 3098, P. 87–124. DOI: 10.1007/978-3-540-27755-2_3

14. Zuberek, W.M. Timed Petri nets definitions, properties and applications // Microelectronics Reliability, 1991. Vol. 31, No. 4. P. 627–644 DOI: 10.1016/0026-2714(91)90007-T

15. Boudjadar, A J., Kim, J.H., Larsen, K.G., et al. Model Checking Process Algebra of Communicating Resources for Real-time Systems // Proceedings of the 26th Euromicro Conference on Real-Time Systems (ECRTS 2014), 2014, Madrid, Spain. P. 51–60. DOI: 10.1109/ECRTS.2014.24

16. Lime D., Roux O.H., Seidner C., et al. Romeo: A Parametric Model-Checker for Petri Nets with Stopwatches // Tools and Algorithms for the Construction and Analysis of Systems. TACAS 2009. LNCS. Springer, 2009. Vol. 5505, P. 54–57 DOI: 10.1007/978-3-642-00768-2_6

17. Henzinger, T. The Theory of Hybrid Automata // Verification of Digital and Hybrid Systems. NATO ASI Series. Springer, 2000. Vol. 170. P. 265–292 DOI: http://dx.doi.org/10.1007/978-3-642-59615-5_1

18. Buslenko N.P. *Modelirovanie slozhnyh sistem* [Complex systems modeling]. Moscow, Nauka, 1978. 400 p.

19. Andre E. Observer patterns for real-time systems // Proceedings of 2013 18th IEEE International Conference on Engineering of Complex Computer Systems (ICECCS), 2013, Singapore. P. 125–134 DOI: 10.1109/ICECCS.2013.26

20. Avionics application software standard interface. ARINC specification 653 // Aeronautical Radio. Annapolis, 1997.