УДК 65.012.122 DOI 10.18522/2311-3103-2023-1-288-299

Н.В. Колесов, Е.В. Лукоянов, В.С. Тюльников, Р.Л. Крючков

МАСШТАБИРУЕМАЯ ОБРАБОТКА ИНФОРМАЦИИ В БОРТОВОЙ РАСПРЕДЕЛЕННОЙ ВЫЧИСЛИТЕЛЬНОЙ СИСТЕМЕ АНПА*

Рассмотрению вопросов разработки распределенных вычислительных систем (РВС) уделяется большое внимание в современной научно-технической литературе. При этом, как правило, в работах обсуждается только один из наиболее значимых аспектов, например, производительность, надежность, отказоустойчивость, энергоэффективность и масштабируемость. Основной вклад настоящей статьи состоит в попытке комплексного рассмотрения проблемы проектирования РВС, опирающегося на пример многоканальной бортовой РВС обработки информации автономного необитаемого подводного аппарата (АНПА). Целью статьи является формулировка единой концепции многоканальной бортовой РВС обработки информации в реальном времени. В результате предложены архитектура и принципы функционирования многоканальной бортовой РВС, основывающиеся на известных подходах к обеспечению отказоустойчивости и энергоэффективности, но учитывающие особенности масштабируемых систем. Предлагаемые решения можно рассматривать как усиление традиционных подходов к проектированию масштабируемых систем. Отказоустойчивость достигается путем применения средств тестового диагностирования. При этом для сокращения сложности этих средств в каждый программный модуль (ПМ) системы предварительно вводится избыточность, для которой далее строится тест. В работе показывается, что этот тест обнаруживает ошибки в адресации обменов между функциональными блоками ПМ, реализующими процесс обработки информации. По результатам анализа реакции на тест происходит обнаружение отказавшего ПМ, после чего он прекращает свою работу, а вместо него запускается новый ПМ, реализующий тот же процесс. Предложения в части энергоэффективности рассчитаны на случай присутствия в системе избыточных процессоров (в том числе многоядерных), которые привлекаются к исполнению ПМ системы с одновременным снижением тактовой частоты и напряжения питания. Поскольку потребляемая в РВС мощность существенно зависит от этих параметров, происходит ее заметное снижение. Для решения задачи используется оптимальный жадный алгоритм, предполагающий последовательное введение в систему дополнительных процессоров. Важно, что данное предложение по увеличению энергоэффективности РВС сообщает последней дополнительные возможности по отказоустойчивости. Практическая значимость предложенной концепции заключается в возможности её использования не только в АНПА, но и в других случаях комплексного проектирования современных масштабируемых многоканальных бортовых РВС обработки информации в реальном времени с повышенными требованиями по отказоустойчивости и энергоэффективности.

Отказоустойчивость; энергоэффективность; распределенные вычислительные системы; масштабируемые системы; автономные необитаемые подводные аппараты.

N.V. Kolesov, E.V. Lukoyanov, V.S. Tyulnikov, R.L. Kryuchkov

SCALABLE DATA PROCESSING IN AUV ONBOARD DISTRIBUTED COMPUTING SYSTEM

Development of distributed computing systems (DCS) takes an important place in modern scientific and technical literature. Generally, only one of the most significant features of DCS is discussed in the papers, for example, performance, reliability, fault tolerance, energy efficiency and scalability. In this paper authors attempt to overall consider the problem of DCS design, based on the example of a multi-channel onboard DCS for data processing places in autonomous underwater vehicle (AUV). The aim of this paper is to formulate a unified concept of a multi-channel onboard DCS for real-time data processing. As a result, the architecture and principles of operation of a multi-channel onboard DCS are proposed, based on well-known approaches to fault tolerance and ener-

* Работа выполнена при поддержке гранта РНФ № 22-29-00339.

gy efficiency, taking into account the features of scalable systems. The proposed solutions can be viewed as an advancement of traditional approaches to scalable systems development. Fault tolerance is achieved by using test-based diagnostic tools. In order to reduce the complexity of these tools, redundancy is preliminarily added into each software module (SM) of the system. Then tests for the redundancies are built. It is shown that this test detects failures in the addressing of exchanges between SM blocks that implement the data processing. Based on the results of the analysis of the diagnostic tool reaction to the test, a failed software module is detected. Then failed module stops its work, and a new SM that implements the same algorithm is started execute instead. Energy efficiency proposals are suitable for the case of the presence of redundant processors in the system which could support multi-core technology. These processors could be involved in the execution process of the system SMs with a simultaneous decrease in the clock frequency and supply voltage. Since the power consumption in the DCS significantly depends on frequency and supply voltage, it decreases along with this parameter values. An optimal greedy algorithm is used to solve described problem, which assumes sequential involving of additional processors into the system. It is important that energy efficiency proposal of the DCS provides the latter additional fault tolerance capabilities. The practical importance of the proposed concept consists in the possibility of using not only in AUVs application. It also could be used in other cases of scalable multi-channel onboard DCS development with real-time data processing which have fault tolerance and energy efficiency requirements.

Fault tolerance; energy efficiency; distributed computing systems; scalable systems; autonomous uninhabited underwater vehicles.

Введение. Проблемы создания и применения АНПА являются актуальными [1–4], при этом большое внимание уделяется и проектированию их РВС [5–8]. В каждой из указанных работ, как правило, обсуждается только один из наиболее значимых аспектов проектирования, например, производительность, надежность, отказоустойчивость, энергоэффективность и масштабируемость. Основной вклад настоящей работы состоит в попытке комплексного рассмотрения проблемы проектирования РВС. Будем предполагать, что объектом дальнейшего рассмотрения является масштабируемая многоканальная бортовая РВС реального времени, степень «жесткости» которого не будет уточняться. В результате прототипом системы, на наш взгляд, может служить, например, многоканальная система обработки информации в гидроакустическом комплексе АНПА. Далее для подобных масштабируемых систем последовательно обсуждаются вопросы отказоустойчивости и энергоэффективности. Предлагаемые решения можно рассматривать как усиление традиционных подходов к проектированию масштабируемых систем.

Основными понятиями при описании математической модели масштабируемой РВС являются понятия "процесса" и "программного ресурса" [9, 10]. При этом под процессом понимается последовательность блоков (команд, процедур) $Q_1, Q_2, ..., Q_S$. В рассматриваемой системе процессы представляются в виде ПМ, которые исполняются процессорами системы. Будем считать допустимой ситуацию, когда в системе выполняются параллельно во времени многочисленные копии одного и того же ПМ. Исполняемые ПМ будем называть программным ресурсом. Текущий размер программного ресурса в масштабируемой системе в каждый момент времени ограничен и может меняться в зависимости от режима работы РВС, определяемого условиями применения. В результате намеченные к исполнению ПМ оказываются в роли конкурирующих.

Отказоустойчивость масштабируемых РВС. Традиционный подход к обеспечению отказоустойчивости масштабируемых систем предполагает наличие в системе средств диагностирования, обнаруживающих нарушение в работе любого ПМ. После этого ПМ с нарушением прекращает свою работу, а вместо него запускается новый ПМ, реализующий тот же процесс, что и в остановленном ПМ. При этом остается вопрос о средствах обнаружения нарушения. В конкретных приложениях он решается по-разному и с разным успехом. Ниже предлагается использовать путь, который, как нам кажется, хорошо сочетается с особенностями масштабируемой РВС. Решение основано на упрощенной версии авторского подхода к тестовому диагностированию РВС реального времени [11, 12].

Коротко изложим основную идею. Прежде заметим, что подход можно считать составной часть иерархического процесса проектирования средств диагностирования для сложных систем [5]. В этом случае компоненты системы с соблюдением отношения включения размещаются по уровням сложности, после чего для каждого уровня синтезируются свои средства диагностирования, опирающиеся на свои модели и методы. В данном случае речь идет о диагностировании на верхнем уровне, на котором распределенная система представляется композицией локальных систем или ПМ, составляющих программное обеспечение системы. При этом рассматриваемый класс отказов включает всевозможные нарушения в адресации обменов между ПМ, размещенными на процессорах РВС и обменивающимися необходимой информацией асинхронно, т.е. по ее готовности.

Итак, в системе используются средства тестового диагностирования (СТД), представленные отдельным ПМ. Они генерируют для каждого ПМ (каждого запущенного в работу процесса) персональный тест, проверяющий правильность его «сборки» из набора блоков. Однако известно, что в общем случае, как процедура построения теста, так и длина самого теста будут характеризоваться экспоненциальной по отношению к размерности модели [13]. В связи с этим постановка задачи тестового диагностирования изменяется, а именно, она предполагает предварительное введение в каждый ПМ системы избыточности с целью упрощения процедуры построения тестов и сокращения их длины. Остается решить вопросы о способе введения избыточности и о процедуре построения теста.

Вводимая избыточность представляет собой встраиваемую в каждый ПМ цепочку из простых динамических звеньев M_i (по одному звену на каждый функциональный блок Q_i), исполняемую параллельно основному алгоритму ПМ (рис. 1). Как уже отмечалось, в системе присутствуют СТД, но они формируют тесты не для сложного ПМ, а для введенной в ПМ простой цепочки из динамических звеньев. В результате и процедура построения теста и сам тест оказываются простыми. При этом, поскольку реакция звеньев M_i на тест передается в общих массивах с основной информацией от блоков Q_i, то в случае неправильной адресации основной информации от Q_i неправильной окажется и адресация тестовой информации от M_i. Таким образом, контролируя выходную тестовую информацию с выхода цепочек, можно контролировать правильность адресации основной информации внутри цепочек. Отсюда следует, что встраиваемая цепочка может рассматриваться как диагностическая модель ПМ.

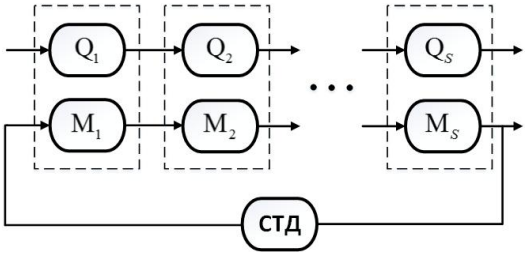

Рис. 1. Иллюстрация идеи тестового диагностирования процесса, Q_i – функциональный блок, M_i – диагностическое звено, СТД – средства тестового диагностирования

Уточним модели звена в цепочке. Из теории технического диагностирования известно [5], что наибольшей эффективности диагностирования при наименьших затратах можно достичь в случае, когда объект диагностирования – линейный.

Кроме того, ясно, что, поскольку речь идет об анализе последовательности событий (последовательности выполнения блоков), то этот алгоритм должен быть динамическим. В результате получаем, что моделью звена должна быть линейная динамическая система, матрицы которой определены в двоичном поле $\mathbb{F} = \{0,1\}$:

$$\begin{aligned} &\mathbf{x}_{i,j}(k+1) = \mathbf{f}_{i,j}\mathbf{x}_{i,j}(k) + \mathbf{g}_{i,j}\mathbf{u}_{i,j}(k), \\ &\mathbf{y}_{i,j}(k) = \mathbf{h}_{i,j}\mathbf{x}_{i,j}(k), \\ &i = \overline{1, l(j)},\ j = \overline{1, m}, \end{aligned} \tag{1}$$

где $\mathbf{x}_{i,j}(k) \in \mathbb{F}^n, \mathbf{u}_{i,j}(k) \in \mathbb{F}^q, \mathbf{y}_{i,j}(k) \in \mathbb{F}^p$ – векторы состояния, входа и выхода соответственно для i-го звена модели j-й цепи, n – размерность вектора состояния, q – размерность входного вектора, p – размерность выходного вектора, $\mathbf{f}_{i,j} \in \mathbb{F}^{n\times n}, \mathbf{g}_{i,j} \in \mathbb{F}^{n\times q}, \mathbf{h}_{i,j} \in \mathbb{F}^{p\times n}$ – матрицы динамики, входа и выхода, $l(j)$ – число динамических звеньев (функциональный блоков) в ПМ$_j$, j – количество ПМ (процессов) в системе.

Обсуждая настоящую тему, следует упомянуть важное направление в моделировании сложных систем, получившее широкое распространение в последние годы. Имеется в виду использование модели дискретной событийной системы [14]. В этом случае поведение системы описывается в виде последовательности событий. Данная модель достаточно часто применяется и при решении задач диагностирования [15]. Рассматриваемую далее модель также можно отнести к классу дискретных событийных моделей, поскольку в этом случае поведение системы представляется как последовательность событий обмена между блоками ПМ.

Описание модели цепи получается по следующим правилам. Формируется вектор состояния цепи $\mathrm{X}(k)$, составленный из векторов состояния входящих в нее звеньев $\mathrm{x}_i(k)$, $i = \overline{1, l}$, динамика которых описывается уравнениями типа (1). С помощью блочных матриц $\mathbf{F}(j(k)), \mathbf{G}(j(k)), \mathbf{H}(j(k))$ описывается перенос информации между блоками ПМ или средствами диагностирования и ПМ в каждом $j(k)$-м информационном обмене. В силу периодичности процесса съема информации с датчиков в системе реального времени обработка информации носит также периодический характер. Для удобства описания свяжем с каждой последовательностью матриц на интервале, равном этому периоду, свою последовательность индексов, множество которых обозначим через $\Gamma = \{\gamma_s\}_{s=1}^{N}$, где $N = l + 1$, так как добавляются дополнительные сеансы обмена информацией со средствами диагностирования. Тогда описание модели для системы принимает вид:

$$\begin{aligned} &\mathbf{X}(k+1) = \mathbf{F}(\gamma_s(j(k)))\mathbf{X}(k) + \mathbf{G}(\gamma_s(j(k)))\mathbf{u}(k), \\ &\mathbf{y}(k) = \mathbf{H}(\gamma_s(j(k)))\mathbf{X}(k), \end{aligned} \tag{2}$$

где $\mathbf{x}(k) \in \mathbb{F}^{N\cdot n}, \mathbf{u}(k) \in \mathbb{F}^{N\cdot q}, \mathbf{y}(k) \in \mathbb{F}^{N\cdot p}$ – векторы состояния, входа и выхода, $\mathbf{F}(\gamma_s(j(k))) \in \mathbb{F}^{N\cdot n\times N\cdot n}$, $\mathbf{G}(\gamma_s(j(k))) \in \mathbb{F}^{N\cdot n\times N\cdot q}$, $\mathbf{H}(\gamma_s(j(k))) \in \mathbb{F}^{N\cdot p\times N\cdot n}$ – матрицы динамики входа и выхода соответственно, $j(k) = \overline{1, N}$ – счётчик информационных обменов.

Если цепь состоит из l звеньев, то уравнения (2) описывают $l-1$ межзвенных обменов и два обмена со средствами диагностирования: прием теста на вход цепи и выдача реакции модели на тест. Если предположить, что при обмене информацией срабатывает лишь звено принимающее информацию, то в соответствии с выражением (2) получим следующие значения для матриц модели цепи:

- при приеме информации от средств диагностирования $(j(k)=1)$ срабатывает только модель принимающего звена цепи $\mathrm{M}_1 : \mathbf{F}(1)\neq 0,\ \mathbf{G}(1)\neq 0,\ \mathbf{H}(1)=0$, $\mathbf{u}(1)=test$;
- при передаче информации между звеньями цепи M_i и $\mathrm{M}_{i+1}(1<j(k)<l+1)$ срабатывает модель звена $\mathrm{M}_{i+1}: F(j(k))\neq 0, G(j(k))\neq 0,\ H\big(j(k)\big)\neq 0, u_{i+1}(k)=y_i(k)$, при этом на его вход подается информация с выхода предыдущего звена;
- при выдаче информации в средства диагностирования $(j(k)=l+1)$ не срабатывает модель ни одного звена, $\mathrm{M}_l : \mathbf{F}(l+1)\neq 0,\ \mathbf{G}(l+1)=0,\ \mathbf{H}(l+1)\neq 0$, $\mathbf{y}(l+1)=reaction$.

Для создания отказоустойчивой системы требуется дополнительно решить еще несколько вопросов. Среди них построение теста с использованием рассмотренной выше модели, а также определение процедур диагностирования отказавшего ПМ и восстановления системы после отказа. Эти вопросы могут быть решены с применением PMC-модели [5, 16] с распределенным принятием решения об отказе, что позволит реализовать децентрализованную отказоустойчивую систему, лишенную «узких» мест [17].

Обсудим реализацию PMC-модели, предварительно определив модель отказа ПМ как утрату функции обработки информации при сохранении функции ее трансляции со входа на выход. Таким образом, процесс диагностирования можно разбить на три этапа: проверка, сбор диагностической информации и принятие решения об отказе. На первом этапе предполагается, что ПМ осуществляют взаимные проверки. Каждая из проверок характеризуется малой длительностью, поскольку нацелена на диагностирование лишь небольшой части объекта диагностирования, опираясь на которую объект может далее построить процесс самодиагностирования. Второй этап, очевидно, должен отличаться наибольшей длительностью, поскольку предполагает передачу результатов взаимных проверок и самодиагностирования от каждого ПМ к каждому. Наконец, на третьем этапе в каждом из ПМ происходит анализ результатов диагностирования и принятие решения об отказе.

Проанализируем требования, предъявляемые к коммуникационной системе. Понятно, что с одной стороны число информационных связей между ПМ необходимо сокращать, однако с другой стороны число и геометрия этих связей влияет на эффективность диагностирования. Действительно, применяемая коммуникационная система должна не только допускать организацию в системе принятой архитектуры диагностики, но и не увеличивать чрезмерно её длительность.

Моделирование. С целью получения таких характеристик системы коммуникации было проведено моделирование в среде YACSIM [18]. При этом исследовались коммуникационные системы трех типов – решетка, тор, гиперкуб. Целевой характеристикой моделирования была длительность диагностического эксперимента, включающего взаимные межпроцессные проверки и сбор всех результатов в каждом из ПМ. Моделирование проводилось в предположении, что реализуется эксперимент с проверкой соседних ПМ, причем время проверки одного ПМ другим было принято равным 2,1 усл. ед., а время трансляции диагностической информации со входа ПМ на его выход равным 1,4 усл. ед. На рис. 2 для каждого из трех типов приведены зависимости времени принятия решения об отказе от числа

реализуемых информационных связей. Последовательности расчетных точек на графиках соответствуют последовательности размерностей коммуникационной системы – 2x2, 3x3, 4x4, 5x5.

В ходе диагностического эксперимента получено, что наиболее эффективной коммуникационной системой с точки зрения времени принятия решений является схема типа «тор».

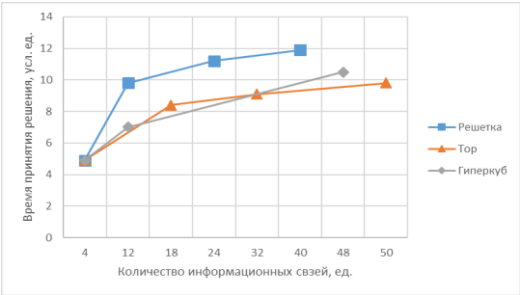

Рис. 2. Результаты моделирования различных систем коммуникации

Энергоэффективность масштабируемых РВС. Коротко опишем предлагаемое решение для обеспечения энергоэффективности бортовой РВС. Предположим, что осуществлено предварительное назначение задач на процессоры, в результате которого каждая задача задания выполняется на отдельном процессоре. В дальнейшем процессор в исходной архитектуре системы будем называть стадией. Предположим, что в системе существуют дополнительные неиспользуемые процессоры. Постановка задачи состоит в определении такой архитектуры A и такого перераспределения вычислительной нагрузки между всеми процессорами, чтобы мощность P, потребляемая системой, была бы минимальной:

$$A^* = \arg\min_{A} P(A). \tag{3}$$

Обсудим соотношения, описывающие потребляемую системой мощность P, поскольку именно она будет составлять основу критерия оптимизации (3) на этапе назначения задач. Сразу отметим, что случай с минимизацией энергии вместо мощности аналогичен рассматриваемому ниже, поскольку энергия, потребляемая системой, равна произведению мощности на время работы системы. Известно [19, 20], что мощность имеет две составляющие – динамическую P_d и статическую P_s. Выражения, описывающие эти составляющие без излишней для данного изложения детализации, имеют вид:

$$P_d = aNV^2 f\ ,\ P_s = bN\ , \tag{4}$$

где a, b – коэффициенты пропорциональности, зависящие от свойств кристалла, N – число процессоров в системе, V – напряжение питания, f – тактовая частота. Поскольку вклад статической мощности в суммарную потребляемую мощность невелик, далее будем учитывать лишь динамическую составляющую. Для ее анализа полезна приближенная формула, определяющая задержку, вносимую схемой при напряжении питания V [21]:

$$D = \frac{c}{V}\ , \tag{5}$$

где c – коэффициент пропорциональности, зависящий от свойства кристалла.

При снижении частоты тактовых импульсов в обратной пропорции возрастает их период, ограничивающий допустимое время для переходных процессов, возникающих в системе при каждом срабатывании. При исходном значении напряжения питания фактическое время переходных процессов будет мало по отношению к новому увеличенному значению периода, а значит, возникает возможность пропорционально снизить напряжение питания с увеличением задержки в рамках периода тактовых импульсов в соответствии с (5). В результате выполнения этих двух шагов может быть достигнуто существенное снижение потребляемой мощности (4). Описанный факт положен в основу предлагаемого подхода к определению архитектуры системы. Суть подхода состоит в последовательном определении для каждой стадии числа реализующих ее процессоров. Причем предполагается, что все процессоры стадии работают на одной частоте и при одном напряжении питания, а также что для системы известен допустимый исходный вариант значений параметров f_0, V_0. Безусловно, проектируя систему, разработчик всегда ограничен не только по потребляемой мощности, но по числу запасных процессоров, а также по напряжению питания и частоте.

В общем случае, когда стадий несколько, возникает вопрос о том, как наилучшим образом с точки зрения минимизации потребляемой мощности распорядиться дополнительными процессорами в рамках ограничений. На этот вопрос отвечает приводимый ниже алгоритм определения энергоэффективной архитектуры. Будем называть процессоры исходной архитектуры системы "исходными", процесс добавления в i-ю стадию $n_{d,i}$ дополнительных процессоров – "расщеплением i-го исходного процессора", а образованное в результате этого расщепления множество процессоров – "расщепленным множеством i-го процессора". Кроме того, будем считать расщепленное множество i -го процессора предельным, если исключение из него одного процессора делает его среднюю по множеству потребляемую мощность максимальной среди стадий системы. Интуитивно, по-видимому, ясно, что экономия мощности будет максимальной, если разгружаться будут наиболее загруженные процессоры, а распределение нагрузки между процессорами в расширенном множестве будет осуществляться сбалансированным образом, т.е. равномерно. По этому принципу работает предлагаемый алгоритм. Заметим, что идеальная сбалансированность при распределении нагрузки, когда нагрузка делится на равные части между процессорами стадии, на практике в общем случае невозможна. В связи с этим речь идет лишь о приближенной сбалансированности. Архитектуру системы представим вектором состава $A = (a_1 \quad a_2 \quad ... \quad a_n)$, где a_i – число процессоров в расщепленном множестве i-го процессора, и вектором средней потребляемой мощности $\bar{P} = (\bar{P}_1 \quad \bar{P}_2 \quad ... \quad \bar{P}_n)$, где $\bar{P}_i$ – средняя по расщепленному множеству i-го процессора потребляемая мощность. Итак, предлагается следующий простой и оптимальный алгоритм.

Алгоритм 1 (определение энергоэффективной архитектуры).

Шаг 1. Сделать начальные присвоения: $M = n_d$ – допустимое число дополнительных процессоров, $A = (1 \quad 1 \quad ... \quad 1)$, $\bar{P} = (\bar{P}_1 \quad \bar{P}_2 \quad ... \quad \bar{P}_n)$.

Шаг 2. Выбрать в $\bar{P} = (\bar{P}_1 \quad \bar{P}_2 \quad ... \quad \bar{P}_n)$ компоненту с максимальным значением $\bar{P}_{\max}$. Пусть ее номер равен единице. Ввести дополнительный процессор в l-ю стадию. Произвести между процессорами l-ой стадии приближенно сбалансированное перераспределение нагрузки. Пересчитать параметры алгоритма: $\bar{P}, \ a_i := a_i + 1, \ M := M - 1$. Если $M \neq 0$, то повторить шаг 2, иначе конец.

Поскольку приведенный алгоритм оперирует не с абсолютными значениями мощностей, а с их соотношением, то с инженерной точки зрения возможен переход к оперированию не потребляемыми в стадиях мощностями, а их вычислительными сложностями.

Поскольку обычно сбалансированное распределение задач невозможно, возможен переход к оперированию вычислительными сложностями решаемых в стадиях задач. Итак, пусть планируется m заданий $\{\tau_j\}_{j=1}^{m}$, каждое j-е из которых включает n задач $\{\tau_{j,i}\}_{i=1}^{n}$. Введем понятие вычислительной сложности Ψ_j задания τ_j, под которой будем понимать число составляющих это задание операций. В относительных расчетах, проводимых далее в примере, в качестве оценок можно использовать его исходную длительность (до снижения частоты). Аналогично введем вычислительную сложность $\Psi_{j,i}$, для задачи $\tau_{j,i}$, а также вычислительную сложность Ψ^i для стадии i. При этом

$$\Psi_j = \sum_{i=1}^{n} \Psi_{j,i}, \quad \Psi^i = \sum_{j=1}^{m} \Psi_{j,i}.$$

Выбирая в расщепленном множестве каждого i-го исходного процессора процессор с максимальной нагрузкой $\Psi^i_{\max}$, рассчитываем для стадии минимальную тактовую частоту, как пропорциональную вычислительной сложности:

$$f^i_{\min} = f_0 \frac{\Psi^i_{\max}}{\Psi^i}. \qquad (6)$$

Далее в той же пропорции снижается напряжение питания: $V^i_{\min} = V_0 \frac{\Psi^i_{\max}}{\Psi^i}$.

Подчеркнем, что выбор варианта размещения задач стадии целесообразно делать в пользу сбалансированного назначения, поскольку в этом случае вычислительная сложность максимального блока будет минимальной, а, значит, и минимальной будет выбираемая тактовая частота стадии (6). Рассмотрим иллюстративный пример.

Пример. Пусть система содержит три процессора, на которых выполняются четыре задания $\tau_1, \tau_2, \tau_3, \tau_4$, имеющие длительности стадий, приведенные в табл. 1, которые выражены в условных единицах.

Таблица 1

Длительности стадий для заданий $\tau_1, \tau_2, \tau_3, \tau_4$

№	Параметр	$\tau_{j,1}$	$\tau_{j,2}$	$\tau_{j,3}$
1	τ_1	3	2	1
2	τ_2	3	1	1
3	τ_3	3	1	1
4	τ_4	2	1	1
5	τ_Σ	11	5	4
6	$\overline{P}$	22	10	8

В 5-ой строке таблицы приведены суммарные длительности для всех задач каждой из стадий. Далее они будут использованы как оценки вычислительной сложности и оценки потребляемой на каждой стадии мощности в условных единицах. В 6-ой строке приведены значения потребляемой в стадиях мощности. Необходимо определить наилучшее расщепление для каждого процессора при условии, что запас свободных процессоров позволяет добавить в систему четыре процессора ($n_d = 4$). Для решения задачи воспользуемся вышеприведенным алгоритмом с анализом длительностей стадий.

Делаем начальные присвоения $\mathrm{M} = 4$, $A = (1 \quad 1 \quad 1)$, $\bar{P} = (22 \quad 10 \quad 8)$.

Расщепляем процессор первой стадии, вводя два дополнительных процессора. Распределяем нагрузку примерно равномерно между тремя процессорами: {5, 3, 3}. Вычисляем коэффициент снижения тактовой частоты (напряжения питания) на основании (6) $k_1 = \dfrac{\Psi^1}{\Psi^1_{\max}} = \dfrac{\tau_{\Sigma,1}}{\tau_{3,1} + \tau_{4,1}} = 2{,}2.$ Вычисляем новое значение удельной мощности, потребляемой процессорами расщепленного множества 1-й стадии: $\bar{P}_1' = \dfrac{\bar{P}_1}{k_1^3} = 2{,}1 < 10$. Вычисляем количество запасных процессоров $n_d = 2 \neq 0.$ Переходим к следующему шагу расщепления.

Расщепляем процессор второй стадии, вводя один дополнительный процессор. Распределяем нагрузку примерно равномерно между двумя процессорами: {3, 2}. Вычисляем коэффициент снижения тактовой частоты (напряжения питания) на основании (6) $k_2 = \dfrac{\Psi^2}{\Psi^2_{\max}} = \dfrac{\tau_{\Sigma,2}}{\tau_{1,2} + \tau_{2,2}} = 1{,}7.$ Вычисляем новое значение удельной мощности, потребляемой ядрами расщепленного множества 2-ой стадии: $\bar{P}_2' = \dfrac{\bar{P}_2}{k_2^3} = 2 < 8$. Вычисляем количество запасных процессоров $n_d = 1 \neq 0$. Переходим к следующему шагу расщепления.

Расщепляем процессор третьей стадии, вводя один дополнительный процессор. Распределяем нагрузку равномерно между двумя процессорами: {2, 2}. Вычисляем коэффициент снижения тактовой частоты (напряжения питания) на основании (6) $k_3 = \dfrac{\Psi^3}{\Psi^3_{\max}} = \dfrac{\tau_{\Sigma,3}}{\tau_{1,3} + \tau_{2,3}} = 2.$ Вычисляем новое значение удельной мощности, потребляемой процессорами расщепленного множества 3-й стадии: $\bar{P}_3' = \dfrac{\bar{P}_3}{k_3^3} = 1$. Вычисляем количество запасных процессоров $n_d = 0$. Конец.

Таким образом, результирующая система содержит семь процессоров и характеризуется следующим вектором удельных нагрузок для стадий $\bar{P} = (2{,}1 \quad 2 \quad 1)$. Оценим приближенно результирующее снижение потребляемой мощности. В качестве оценки мощности в условных единицах, потребляемой исходной системой, будем использовать, как уже отмечалось $P_0 = \sum_j \bar{P}_{\Sigma,j} = 40$. При этом для преобразованной системы выражение будет иметь вид $P = \sum_j \dfrac{\bar{P}_{\Sigma,j}}{k_j^3} k = 12{,}3,$ где $k_j = \dfrac{\tau_{\Sigma,j}}{\tau_{j,\max}}$ – коэффициент снижения частоты (напряжения

питания), $\overline{P}_{j,\max}$ – наибольшая средняя мощность, потребляемая процессорами расщепленного множества j-ой стадии. В результате снижение мощности выражается величиной $l = \frac{P_0}{P} = 3,3$.

Важно отметить, что рассмотренное предложение по увеличению энергоэффективности РВС сообщает последней дополнительные возможности по отказоустойчивости. Действительно, при отказах процессоров возможно производить их замещение работоспособными с одновременным повышением таковой частоты и напряжения питания.

Заключение. В работе предложен подход к построению масштабируемой отказоустойчивой и энергоэффективной бортовой РВС. Подход включает два этапа, во-первых, решение проблемы отказоустойчивости средствами тестового диагностирования, и, во-вторых, определение для системы энергоэффективной архитектуры за счет введения в состав системы дополнительных процессоров, сопровождающегося снижением тактовой частоты и напряжения питания.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Ramírez I. S., Bernalte Sánchez P. J., Papaelias M., Márquez F. P. G.* Autonomous underwater vehicles and field of view in underwater operations // Journal of Marine Science and Engineering. – 2021. – Vol. 9, No. 3. – P. 277.
2. *Yang Y., Xiao Y., Li T.* A survey of autonomous underwater vehicle formation: Performance, formation control, and communication capability // IEEE Communications Surveys & Tutorials. – 2021. – Vol. 23, No. 2. – P. 815-841.
3. *Sahoo A., Dwivedy S.K., Robi P.S.* Advancements in the field of autonomous underwater vehicle //Ocean Engineering. – 2019. – Vol. 181. – P. 145-160.
4. *Инзарцев А.В., Киселев Л.В., Костенко В.В., Матвиенко Ю.В., Павин А.М., Щербатюк А.Ф.* Подводные робототехнические комплексы: системы, технологии, применение. – Владивосток: Институт проблем морских технологий Дальневосточного отделения Российской академии наук, 2018. – 368 с.
5. *Колесов Н.В., Толмачева М.В., Юхта П.В.* Системы реального времени. Планирование, анализ, диагностирование. – СПб.: ОАО "Концерн "ЦНИИ "Электроприбор", 2014. – 185 с.
6. *Kshemkalyani A. D., Singhal M.* Distributed computing: principles, algorithms, and systems. – Cambridge University Press, 2011. – 731 p.
7. *Топорков В.В.* Модели распределенных вычислений. – М.: Физматлит, 2011. – 320 с.
8. *Воеводин В.В., Воеводин Вл.В.* Параллельные вычисления. – СПб.: БХВ-Петербург, 2002. – 608 с.
9. *Kapitonova Y.V., Kovalenko N.S., Pavlov P.A.* Optimality of systems of identically distributed competing processes // Cybernetics and Systems Analysis. – 2005. – Vol. 41, No. 6. – P. 793-799.
10. *Павлов П.А.* Эффективность распределённых вычислений в масштабируемых системах // Информатика, телекоммуникации и управление. – 2010. – Т. 93, № 1. – С. 83-89.
11. *Грузликов А.М., Колесов Н.В.* Дискретно-событийная диагностическая модель распределенной вычислительной системы. Независимые цепи // Автоматика и телемеханика. – 2016. – № 10. – С. 140-155.
12. *Грузликов А.М., Колесов Н.В., Лукоянов Е.В., Толмачева М.В.* Диагностическая модель для распределенной вычислительной системы реального времени // Известия Российской академии наук. Теория и системы управления. – 2020. – № 5. – С. 44-55.
13. *Бурдонов И.Б., Косачев А.С., Кулямин В.В.* Использование конечных автоматов для тестирования программ // Программирование. – 2000. – Т. 26, № 2. – С. 61-73.
14. *Cassandras C.G., Lafortune S.* Introduction to discrete event systems. – Boston, MA: Springer US, 2008. – 848 p.
15. *Zaytoon J., Lafortune S.* Overview of fault diagnosis methods for discrete event systems // Annual Reviews in Control. – 2013. – Vol. 37, No. 2. – P. 308-320.

16. *Preparata F. P., Metze G., Chien R. T.* On the connection assignment problem of diagnosable systems // IEEE Transactions on Electronic Computers. – 1967. – No. 6. – P. 848-854.
17. *Gruzlikov A.M., Kolesov N.V., Kostygov D.V., Tolmacheva M.V.* A real-time fault-tolerant and power-efficient multicore system on chip // 2019 IEEE 13th International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSoC). – IEEE, 2019. – P. 354-361.
18. *Молдованова О.В.* Адаптивный алгоритм децентрализованной самодиагностики распределённых вычислительных систем различных топологий // Вестник СибГУТИ. – 2013. – Т. 22, № 2. – С. 22-30.
19. *Panda P. R., Shrivastava A., Silpa B.V.N., Gummidipudi K.* Power-efficient system design. – Springer Science & Business Media, 2010. – 253 p.
20. *Rubavani R., Saranraj S., Saranya S., Ranjani Devi R.* Power efficient scheduling for network on chip applications on multicore processor // International Journal of Applied Engineering Research. – 2016. – Vol. 11, No. 7. – P. 4751-4757.
21. *Грузликов А.М., Колесов Н.В., Костыгов Д.В., Ощуев В.В.* Энергоэффективное планирование в распределенных вычислительных системах реального времени // Известия Российской академии наук. Теория и системы управления. – 2019. – № 3. – С. 66-76.

REFERENCES

1. *Ramírez I. S., Bernalte Sánchez P. J., Papaelias M., Márquez F. P. G.* Autonomous underwater vehicles and field of view in underwater operations, *Journal of Marine Science and Engineering*, 2021, Vol. 9, No. 3, pp. 277.
2. *Yang Y., Xiao Y., Li T.* A survey of autonomous underwater vehicle formation: Performance, formation control, and communication capability, *IEEE Communications Surveys & Tutorials*, 2021, Vol. 23, No. 2, pp. 815-841.
3. *Sahoo A., Dwivedy S.K., Robi P.S.* Advancements in the field of autonomous underwater vehicle, *Ocean Engineering*, 2019, Vol. 181, pp. 145-160.
4. *Inzartsev A.V., Kiselev L.V., Kostenko V.V., Matvienko Yu.V., Pavin A.M., Shcherbatyuk A.F.* Podvodnye robototekhnicheskie kompleksy: sistemy, tekhnologii, primenenie [Underwater robotic complexes: systems, technologies, application]. Vladivostok: Institut problem morskikh tekhnologiy Dal'nevostochnogo otdeleniya Rossiyskoy akademii nauk, 2018, 368 p.
5. *Kolesov N.V., Tolmacheva M.V., Yukhta P.V.* Sistemy real'nogo vremeni. Planirovanie, analiz, diagnostirovanie [Real-Time Systems: Planning, Analysis, Diagnostics]. St. Petersburg: OAO "Kontsern "TSNII "Elektropribor", 2014, 185 p.
6. *Kshemkalyani A. D., Singhal M.* Distributed computing: principles, algorithms, and systems. Cambridge University Press, 2011, 731 p.
7. *Toporkov V.V.* Modeli raspredelennykh vychisleniy [Models of Distributed Computing]. Moscow: Fizmatlit, 2011. – 320 s.
8. *Voevodin V.V., Voevodin Vl.V.* Parallel'nye vychisleniya [Parallel computing]. St. Petersburg: BKhV-Peterburg, 2002, 608 p.
9. *Kapitonova Y.V., Kovalenko N.S., Pavlov P.A.* Optimality of systems of identically distributed competing processes, *Cybernetics and Systems Analysis*, 2005, Vol. 41, No. 6, pp. 793-799.
10. *Pavlov P.A.* Effektivnost' raspredelennykh vychisleniy v masshtabiruemykh sistemakh [Efficiency of distributed computing in scalable systems], *Informatika, telekommunikatsii i upravlenie* [Informatics, telecommunications and management], 2010, Vol. 93, No. 1, pp. 83-89.
11. *Gruzlikov A.M., Kolesov N.V.* Diskretno-sobytiynaya diagnosticheskaya model' raspredelennoy vychislitel'noy sistemy. Nezavisimye tsepi [Discrete-event diagnostic model for a distributed computational system. Independent chains], *Avtomatika i telemekhanika* [Automation and Remote Control], 2016, No. 10, pp. 140-155.
12. *Gruzlikov A.M., Kolesov N.V., Lukoyanov E.V., Tolmacheva M.V.* Diagnosticheskaya model' dlya raspredelennoy vychislitel'noy sistemy real'nogo vremeni Diagnostic model for a distributed real-time computing system], *Izvestiya Rossiyskoy akademii nauk. Teoriya i sistemy upravleniya* [Proceedings of the Russian Academy of Sciences. Theory and control systems], 2020, No. 5, pp. 44-55.
13. *Burdonov I.B., Kosachev A.S., Kulyamin V.V.* Ispol'zovanie konechnykh avtomatov dlya testirovaniya programm [The use of finite automata for program testing], *Programmirovanie* [Programming], 2000, Vol. 26, No. 2, pp. 61-73.

14. *Cassandras C.G., Lafortune S.* Introduction to discrete event systems. Boston, MA: Springer US, 2008, 848 p.
15. *Zaytoon J., Lafortune S.* Overview of fault diagnosis methods for discrete event systems, *Annual Reviews in Control*, 2013, Vol. 37, No. 2, pp. 308-320.
16. *Preparata F. P., Metze G., Chien R. T.* On the connection assignment problem of diagnosable systems, *IEEE Transactions on Electronic Computers*, 1967, No. 6, pp. 848-854.
17. *Gruzlikov A.M., Kolesov N.V., Kostygov D.V., Tolmacheva M.V.* A real-time fault-tolerant and power-efficient multicore system on chip, *2019 IEEE 13th International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSoC)*. IEEE, 2019, pp. 354-361.
18. *Moldovanova O.V.* Adaptivnyy algoritm detsentralizovannoy samodiagnostiki raspredelennykh vychislitel'nykh sistem razlichnykh topologiy [Adaptive algorithm for decentralized self-diagnostics of distributed computing systems of various topologies], *Vestnik SibGUTI* [Vestnik SibGUTI], 2013, Vol. 22, No. 2, pp. 22-30.
19. *Panda P.R., Shrivastava A., Silpa B.V.N., Gummidipudi K.* Power-efficient system design. Springer Science & Business Media, 2010, 253 p.
20. *Rubavani R., Saranraj S., Saranya S., Ranjani Devi R.* Power efficient scheduling for network on chip applications on multicore processor, *International Journal of Applied Engineering Research*, 2016, Vol. 11, No. 7, pp. 4751-4757.
21. *Gruzlikov A.M., Kolesov N.V., Kostygov D.V., Oshuev V.V.* Energoeffektivnoe planirovanie v raspredelennykh vychislitel'nykh sistemakh real'nogo vremeni [Energy-efficient planning in real-time distributed computing systems], *Izvestiya Rossiyskoy akademii nauk. Teoriya i sistemy upravleniya* [News of the Russian Academy of Sciences. Theory and control systems], 2019, No. 3, pp. 66-76.

Статью рекомендовала к опубликованию д.т.н. Л.А. Мартынова.

Колесов Николай Викторович – АО «Концерн «ЦНИИ «Электроприбор»; e-mail: kolesovnv@mail.ru; г. Санкт-Петербург, Россия; тел.: 89213886806; д.т.н.; профессор; г.н.с.

Лукоянов Егор Васильевич – e-mail: lukoyanov.egor@mail.ru; тел.: 89522665251; к.т.н.; научный сотрудник.

Тюльников Виктор Сергеевич – e-mail: viktor.tyulnikov@gmail.com; тел.: 89119757141; инженер-программист.

Крючков Роман Леонидович – e-mail: romankr@gmail.com; тел.: 89119580841; инженер-программист.

Kolesov Nikolai Viktorovich – Concern CSRI Elektropribor, JSC; e-mail: kolesovnv@mail.ru; Saint Petersburg, Russia; phone: +79213886806; dr. of eng. sc.; professor; chief researcher.

Lukoyanov Egor Vasilievich – e-mail: lukoyanov.egor@mail.ru; phone: +79522665251; cand. of eng. sc.s; researcher.

Tyulnikov Viktor Sergeevich – e-mail: viktor.tyulnikov@gmail.com; phone: +79119757141; software engineer.

Kryuchkov Roman Leonidovich – e-mail: romankr@gmail.com; phone: +79119580841; software engineer.