

УДК 004.032.3

DOI: [10.26102/2310-6018/2025.48.1.037](https://doi.org/10.26102/2310-6018/2025.48.1.037)

Значение фактора портовости для конфигурирования цикла акторной системы управления реальным временем

А.А. Зеленский, А.А. Грибков✉

Научно-производственный комплекс «Технологический центр», Москва, Зеленоград, Российская Федерация

Резюме. В статье исследуется проблема оптимизации систем управления реальным временем, описываемых в рамках акторной модели. Проблема оптимизации формулируется в виде задачи оптимального конфигурирования цикла управления, т. е. распределения функциональных элементов-акторов по группам, потокам и последовательности исполнения. Предлагается алгоритм конфигурирования, который, хотя и не уменьшает количества анализируемых вариантов конфигурации, однако сокращает объем расчетов по каждому из вариантов. В дополнение к рассмотренным в предыдущих работах авторов вариантам оптимизации с ограничением по суммарной длительности цикла и с ограничением по ресурсам системы управления, в статье рассматривается задача уменьшения числа портов входа и выхода, через которые элементы-акторы обмениваются данными. Исследования показывают, что число портов может быть сокращено без ущерба функциональности системы управления. Это обусловлено последовательным характером выполнения элементов-акторов в рамках одной группы одного потока. В результате одни и те же порты ввода или вывода могут использоваться для связи элемента-актора с несколькими другими. Наряду с сопоставлением различных конфигураций цикла управления, задача снижения количества портов также может решаться за счет использования для связи элементов-акторов общей памяти. При построении системы управления согласно памяти-ориентированной архитектуре через высокоскоростную общую память передаются малые объемы данных, что снижает остроту проблемы образования очередей.

Ключевые слова: система управления, акторная модель, цикл, оптимизация, конфигурирование, портовость, память-ориентированная архитектура.

Благодарности: Исследование выполнено при поддержке Российского научного фонда по гранту № 24-19-00692, <http://rscf.ru/project/24-19-00692/>

Для цитирования: Зеленский А.А., Грибков А.А. Значение фактора портовости для конфигурирования цикла акторной системы управления реальным временем. *Моделирование, оптимизация и информационные технологии*. 2025;13(1). URL: <https://moitvivr.ru/ru/journal/pdf?id=1839> DOI: 10.26102/2310-6018/2025.48.1.037

An importance of the portability factor for configuring the cycle of real-time actuator control system

A.A. Zelenskii, A.A. Gribkov✉

Scientific and Production Complex "Technological Center", Moscow, Zelenograd, the Russian Federation

Abstract. The paper studies the problem of optimization of real-time control systems described within the actor model. The optimization problem is formulated as a problem of optimal configuration of the control cycle, i.e., distribution of functional elements-actors by groups, flows and execution sequence. We propose a configuration algorithm, which, although it does not reduce the number of analyzed configuration variants, reduces the amount of calculations for each of the variants. In addition to the optimization variants with a limit on the total cycle time and with a limit on the control system resources

considered in the authors' previous works, the paper considers the problem of reducing the number of input and output ports through which the element-actors exchange data. The research shows that the number of ports can be reduced without compromising the functionality of the control system. This is due to the sequential nature of element-actors execution within one group of one flow. As a result, the same input or output ports can be used to communicate an actor element with several others. In addition to matching different control loop configurations, the problem of reducing the number of ports can also be solved by using shared memory for element-actor communication. When the control system is built according to memory-oriented architecture, small amounts of data are transferred through high-speed shared memory, which reduces the acuteness of the problem of queue formation.

Keywords: control system, actor model, loop, optimization, configuration, portability, memory-oriented architecture.

Acknowledgements: The research was supported by the Russian Science Foundation under grant No. 24-19-00692, <http://rscf.ru/project/24-19-00692/>

For citation: Zelenskii A.A., Gribkov A.A. An importance of the portability factor for configuring the cycle of real-time actuator control system. *Modeling, Optimization and Information Technology*. 2025;13(1). (In Russ.). URL: <https://moitvvt.ru/ru/journal/pdf?id=1839> DOI: 10.26102/2310-6018/2025.48.1.037

Введение

Специфика управления движением рабочих органов высокотехнологичного промышленного оборудования (станков, промышленных роботов и различных мехатронных систем) определяется требованием высокой точности воспроизводства заданной траектории движения и реализацией управления в реальном времени [1, 2]. Адаптивность современных систем управления движением на практике реализуется в виде дискретных изменений параметров управляемых систем, осуществляемых с некоторой малой периодичностью, характеризуемой длительностью цикла управления. За интервал времени, соответствующий циклу управления, система управления реального времени выполняет комплекс операций, необходимый для обеспечения (с заданной погрешностью) движения рабочего органа управляемого оборудования по заданной траектории с заданными скоростью и ускорением.

Проведенные авторами исследования показали, что практически реализуемым решением задачи управления сложным технологическим оборудованием в реальном времени является представление системы управления в виде акторной модели [3, 4], а цикла управления – в виде конфигурируемой совокупности элементов, соотносимых с акторами [5]. Задача конфигурирования цикла управления заключается в распределении элементов-акторов по последовательности, потокам и группам исполнения в зависимости от их взаимосвязи (причинно-следственной или информационной), длительности, потребных ресурсов (вычислительных и др.), а также необходимых каналов связи между элементами.

Рассмотренные авторами ранее решения задачи конфигурирования предполагали использование неограниченного числа каналов связи между элементами, что эквивалентно числу портов, позволяющему связывать все элементы. Практическая техническая реализация такой неограниченной портовости нецелесообразна, поскольку требует либо подключения всех элементов-акторов к общей памяти (а это приводит к образованию очередей и значительному замедлению работы), либо выстраиванию избыточного числа каналов связи, используемых редко и технически трудно реализуемых. Предварительный анализ числа задействованных портов элементов-акторов в рамках различных конфигураций цикла управления показывает, что без заметного ущерба для быстродействия системы управления возможно существенное

ограничение портовости элементов-акторов (так мы будем называть число используемых портов ввода-вывода элемента-актора для взаимодействия с другими элементами-акторами; взаимодействие элементов-акторов через общую память при этом не учитывается). Насколько жесткими могут быть эти ограничения и каково их итоговое влияние на конфигурацию цикла управления – вопросы, которые мы планируем рассмотреть в данной статье.

Оптимизационная задача конфигурирования цикла управления

На Рисунке 1 приведена диаграмма цикла системы управления, представляющая собой разбиение элементов-акторов по группам, потокам и расстановку их в заданной последовательности в зависимости от решаемой задачи управления и связи элементов-акторов. Реализованное ранее авторами решение оптимизационной задачи конфигурирования цикла системы управления имеет два варианта: оптимизацию можно проводить по критерию времени либо по критерию ресурсов системы.

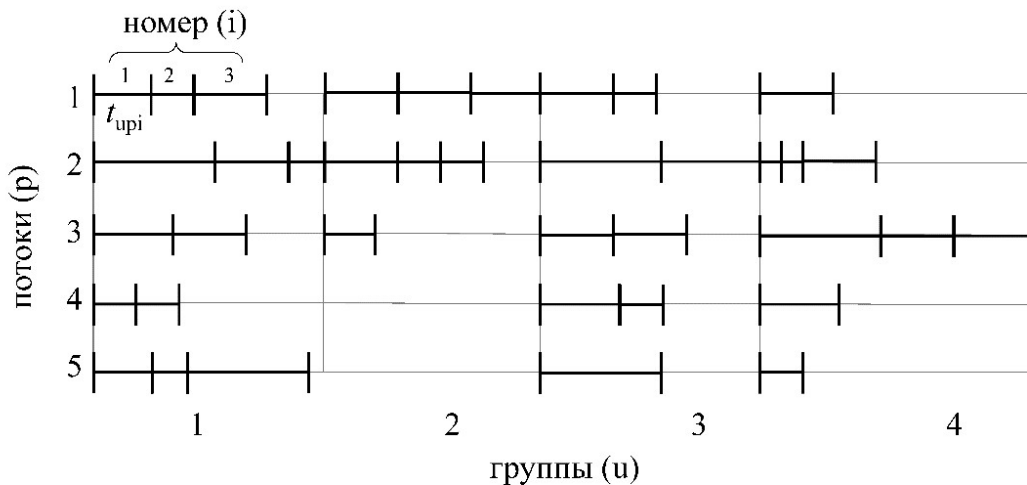


Рисунок 1 – Диаграмма цикла системы управления
Figure 1 – Control system cycle diagram

Каждый элемент p -го потока u -ой группы определяется потребными временем и ресурсами: $a_{upi} = \{t_{upi}, w_{upi}\}$. Длительность цикла системы управления определяется следующим образом [2]:

$$\tau = \sum_{u=1}^{u_{\max}} \max \left(\bigcup_{p=1}^{p_{u\max}} \sum_{i=1}^{i_{up\max}} t_{upi} \right) \leq \tau_{\max},$$

$$\forall u = 1 \dots u_{\max}: w_u(t \leq \tau_u) = \sum_{p=1}^{p_{u\max}} w_{upi} \leq w_{u\max},$$

$$\sum_{q=1}^{i-1} t_{upq} \leq t < \sum_{q=1}^i t_{upq},$$

где u_{\max} – число групп элементов-акторов в цикле управления; $p_{u\max}$ – количество параллельных потоков обработки данных в u -ой группе; $i_{up\max}$ – количество элементов в p -ом параллельном потоке обработки данных в u -ой группе; τ_{\max} – допустимая длительность цикла управления (зависит от допустимой погрешности воспроизводства траектории движения рабочего органа и скорости его движения).

В первом варианте оптимизации заданы потребные ресурсы всех элементов-акторов и определяется такая конфигурация цикла управления, при которой используемые элементами ресурсы не превышают имеющихся в наличии, а суммарная длительность цикла управления – минимальна.

Во втором варианте оптимизации задана допустимая длительность всего цикла управления или длительность выполнения группы элементов и требуется определить конфигурацию, требующую наименьших ресурсов системы, т. е. оптимизация ведется в направлении минимизации w_u . Второй случай существенно менее востребован для непосредственного использования, однако он может быть полезен на стадии определения потребных ресурсов для проектируемой системы управления.

Конфигурирование цикла управления осуществляется отдельно для каждой группы элементов-акторов. Элементы из последующих групп выполняются только тогда, когда все элементы предыдущих групп уже выполнены. Это дополнительная опция конфигурирования, которая может и не использоваться. В последнем случае число групп уменьшается или даже может быть сведено к единственной группе.

Группа образуется множеством элементов-акторов, каждый из которых характеризуется потребным временем t_{ui} , потребными (вычислительными) ресурсами w_{ui} и матрицей отношений с другими элементами-акторами K_{ui} : $a_{ui} = \{t_{ui}, w_{ui}, K_{ui}\}$, где $K_{ui} = \{k_{u1} \ k_{u2} \ \dots \ k_{ui} \ \dots \ k_{ui_{\max}}\}$. Элемент $k_{ui} \in \{-1, 0, 1, 2\}$, где $k_{ui} = -1$, если другой элемент-актор выполняется до рассматриваемого, $k_{ui} = 1$, если другой элемент-актор выполняется после рассматриваемого, $k_{ui} = 0$, если порядок выполнения элементов-акторов не имеет значения, $k_{ui} = 2$ для пары элемента-актора с самим собой.

Выстраивание конфигурации происходит посредством задействования элементов-акторов, в матрице отношений которых отсутствуют элементы $k_{ui} = -1$ и которые не включены в обновляемый список уже использованных элементов-акторов. Как только элемент-актор задействован, он включается в список использованных в виде порядкового номера. Если имеется несколько элементов-акторов, в матрице отношений которых отсутствуют элементы $k_{ui} = -1$, то под каждый такой элемент формируется отдельный поток исполнения. При этом число потоков исполнения не обязательно должно быть максимальным: в рамках формирования различных конфигураций выбирается различное число потоков, не превышающее максимальное.

После завершения выполнения элемента-актора в каком-либо из потоков осуществляется выбор элемента-актора, который будет задействован в данном потоке после него. Этот элемент-актор должен выполняться не раньше всех оставшихся элементов-акторов.

Используемые суммарные ресурсы на всех потоках должны быть не больше имеющихся в наличии. Это условие было уже нами сформулировано в формуле (1):

$$w_u(t \leq \tau_u) \leq w_{u_{\max}}, w_u(t) = \sum_{p=1}^{p_{u_{\max}}} w_{upi}, \quad (2)$$

где номер соответствующего элемента j определяется из выражений $\sum_{q=1}^i t_{upq} > t$ и $\sum_{q=1}^{i-1} t_{upq} \leq t$.

В случае оптимизации по критерию ресурсов системы последнее условие не ставится. Вместо этого производится фиксация величины потребных ресурсов для данной конфигурации и в дальнейшем эти величины сравниваются для выбора варианта с наименьшими потребными ресурсами.

По мере задействования элементов-акторов происходит запись реализуемой конфигурации для того, чтобы не повторять одну и ту же конфигурацию, а также для последующего сравнения различных конфигураций и определения оптимальной (с наименьшей длительностью цикла управления или наименьшими потребностями в ресурсах).

Записанная конфигурация имеет формат массива с числом столбцов, равным максимальному числу элементов-акторов в потоке, и числом строк, равным числу

потоков исполнения. Каждый элемент представлен своим порядковым номером в исходном множестве группы.

Исключить повторение конфигураций по ходу их исполнения невозможно, поскольку они принимаются различными, даже если отличаются в одном заключительном элементе. Поэтому для охвата всех возможных конфигураций необходимо предусмотреть альтернативность выбора элементов-акторов: вначале первого из соответствующего требованиям, затем второго, третьего и так далее, пока не будут исчерпаны все варианты. Каждый выбор приводит к альтернативной цепи последовательности выполнения элементов-акторов и другой конфигурации.

Изложенная выше последовательность конфигурирования принципиально не отличается от предложенной авторами ранее [2]. Количество вариантов, которые должны быть рассмотрены для выявления оптимальной конфигурации, соответствующей наименьшей длительности цикла управления, равно произведению двух величин. Первая величина $N = n_u!$ – число перестановок элементов в множестве элементов группы. При этом в дальнейшем задействуется первый подходящий, что эквивалентно использованию множества акторов с неизменной последовательностью, но с варьированием порядка задействования: первый подходящий, второй подходящий и т. д. Вторая величина $\mu = 2^{n_u-1}$ – количество упорядоченных разбиений числа n_u элементов в исходном множестве группы на цепочки элементов-акторов по потокам.

Максимальная сложность оптимизации очень высокая и характеризуется нотацией $O(n_u! \cdot 2^{n_u-1} \cdot n_u^2)$, минимальная (при использовании лучших алгоритмов сортировки, например, merge sort [6]) – $O(n_u! \cdot 2^{n_u-1} \cdot n_u \cdot \log n_u)$. Минимальное число вариантов, которые необходимо сравнить в процессе оптимизации, составляет, в зависимости от числа элементов-акторов, от 1.8×10^8 при $n_u = 10$ до $3,2 \times 10^{17}$ при $n_u = 15$ и $2,4 \times 10^{25}$ при $n_u = 20$. Реализация этих алгоритмов в реальном времени (т. е. за цикл управления, длительность которого для сложных систем менее 100 мкс) невозможна.

Отличие предложенной в данной статье последовательности конфигурирования заключается в том, что расчет каждого варианта конфигурации в среднем имеет меньший объем за счет того, что большинство вариантов отклоняется на начальной стадии расчета ввиду несоответствия какому-либо из условий, задаваемых матрицей отношений или ограниченностью ресурсов системы.

Портовость акторов: определение требований и минимизация

Техническая реализация системы управления движением как совокупности элементов-акторов с определенным числом портов ввода и вывода данных, накладывает ограничения на решение оптимизационной задачи.

Рассмотрим конкретную реализацию конфигурации цикла управления с заданным разбиением элементов-акторов группы по потокам. Для каждого отдельного элементов-актора $a_{u_0 p_0 i_0}$ в группе u_0 может быть составлено множество матриц отношений

$$K_u(u_0 p_0 i_0) = \left\{ \begin{array}{cccc} k(u11) & k(u12) & \dots & k(u1i_{u_{\max}}) \\ k(u21) & k(u22) & \dots & k(u2i_{u_{\max}}) \\ \dots & \dots & \dots & \dots \\ k(up_{u_{\max}}1) & k(up_{u_{\max}}2) & \dots & k(up_{u_{\max}}i_{u_{\max}}) \end{array} \right\}, u = 1 \dots u_{\max}. \quad (3)$$

Для связи с другими элементами-акторами, задействованными в одном потоке, рассматриваемый элемент-актор может нуждаться не более, чем в двух портах: один на входе (для элементов-акторов, выполняемых раньше) и один на выходе (для элементов-

акторов, выполняемых позже). Порты, необходимые для взаимодействия с элементами-акторами из других потоков или из других групп, используются независимо и должны суммироваться.

Необходимая портовость элемента-актора:

$$\begin{aligned}
 S(a_{u_0 p_0 i_0}) &= \sum_{u=1}^{u_0-1} \sum_{p=1}^{p_{u \max}} \min(-\cup_{i=1}^{i_{u \max}} k(upi), 0) + \\
 &+ \sum_{p=1}^{p_{u_0 \max}} \left(\max\left(\cup_{i=1}^{i_{u_0 \max}} k(u_0 pi), 0\right) - \min\left(\cup_{i=1}^{i_{u_0 \max}} k(u_0 pi), 0\right) \right) + \\
 &+ \sum_{u=1}^{u_0+1} \sum_{p=1}^{p_{u \max}} \max(\cup_{i=1}^{i_{u \max}} k(upi), 0) \leq \\
 &\leq \sum_{u=1}^{u_0-1} p_{u \max} + 2p_{u_0 \max} + \sum_{u=1}^{u_0+1} p_{u \max},
 \end{aligned} \tag{4}$$

где u_0 – группа, к которой принадлежит элемент-актор, портовость которого рассчитывается.

Приведенная формула составлена в предположении возможности подключения к одному порту ввода или вывода элемента-актора нескольких элементов-акторов в том случае, когда заведомо известно, что этот порт не будет задействован более, чем в одной передаче данных за раз. В каждом потоке каждой группы в любой момент времени задействуется только один процесс ввода или вывода данных. Упрощенно последовательность формирования портов можно описать следующим образом. Вначале для элемента-актора задаются все возможные порты: по два (ввод и вывод) для каждого потока своей группы и по одному для каждого потока других групп (вывод – для предшествующих групп, ввод – для последующих групп). Затем, если связи между рассматриваемым элементом-актором и элементами-акторами в выбранном потоке и группе отсутствуют, то порт исключается.

В процессе отработки заданной конфигурации цикла производится запись использованных элементами-акторами портов. В результате формируется множество матриц портов:

$$Z_u = \begin{pmatrix} z(u11) & z(u12) & \dots & z(u1i) & \dots & z(u1i_{u \max}) \\ z(u21) & z(u22) & \dots & z(u2i) & \dots & z(u2i_{u \max}) \\ \dots & \dots & \dots & \dots & \dots & \dots \\ z(up1) & z(up2) & \dots & z(upi) & \dots & z(upi_{u \max}) \\ \dots & \dots & \dots & \dots & \dots & \dots \\ z(up_{u \max}1) & z(up_{u \max}2) & \dots & z(up_{u \max}i) & \dots & z(up_{u \max}i_{u \max}) \end{pmatrix}, u = 1 \dots u_{\max}, \tag{5}$$

где $z(upi)$ – матрица портов конкретного элемента-актора a_{upi} , складывающегося из двух матриц $z(upi).i$ и $z(upi).o$, образованных из номеров элементов-акторов, формирующих с ним связи с образованием портов ввода или вывода:

$$z(upi).i = \begin{pmatrix} \dots & \dots & \dots \\ \dots & u_d p_g i_h & \dots \\ \dots & \dots & \dots \end{pmatrix}, z(upi).o = \begin{pmatrix} \dots & \dots & \dots \\ \dots & u_l p_m i_n & \dots \\ \dots & \dots & \dots \end{pmatrix}, \tag{6}$$

где число строк в матрицах соответствует количеству портов ввода и вывода рассматриваемого элемента-актора, а число столбцов – количеству других элементов-акторов, подключаемых к его портам ввода и вывода.

Поскольку увеличение числа портов у элементов-акторов ведет к росту сложности технической реализации системы управления, минимизация числа портов является значимой задачей конфигурирования системы управления. Суммарное число портов, обеспечивающее выполнение цикла управления, различно для разных

конфигураций и зависит от числа групп, потоков и последовательности выполнения элементов-акторов.

В отличие от двух других факторов оптимизации конфигурации системы управления (длительности цикла и суммарных ресурсов) изначально установить ограничение по суммарному числу портов не представляется возможным: для такого ограничения нет объективных оснований (как, например, для длительности цикла управления, следующих из требований по точности управления) и порты различных акторов независимы друг от друга (порты нельзя складывать как память или другие ресурсы). При этом, однако, можно осуществлять сравнительный анализ различных конфигураций по суммарному числу задействованных портов для выбора конфигурации с меньшим общим количеством портов.

Решение оптимизационной задачи в этом случае выполняется при заданном ограничении по длительности цикла управления и ресурсам. Условие (1) при минимизации по суммарному числу портов запишется в следующем виде:

$$\begin{aligned} \sum_{u=1}^{u_{\max}} \max \left(\bigcup_{p=1}^{p_{u\max}} \sum_{i=1}^{i_{up\max}} t_{upi} \right) &\leq \tau_{\max}, \\ \forall u = 1 \dots u_{\max}: w_u(t \leq \tau_u) &= \sum_{p=1}^{p_{u\max}} w_{upi} \leq w_{u\max}, \\ \sum_{q=1}^{i-1} t_{upq} &\leq t < \sum_{q=1}^i t_{upq}, \\ S_{\Sigma}^{\min} &= \min(\bigcup_{r=1}^{r_{\max}} S_{\Sigma r}), \end{aligned} \quad (7)$$

где S_{Σ}^{\min} – минимальное суммарное число портов; $S_{\Sigma r}$ – суммарное число портов, реализуемое в r -ой конфигурации из r_{\max} рассматриваемых:

$$\begin{aligned} S_{\Sigma r} &= \sum_{u_0=1}^{u_{\max}} \sum_{p=1}^{p_{u\max}} \sum_{i=1}^{i_{u\max}} \left[\sum_{u=1}^{u_0-1} \sum_{p=1}^{p_{u\max}} \min(-\bigcup_{i=1}^{i_{u\max}} k(upi), 0) + \right. \\ &+ \sum_{p=1}^{p_{u_0\max}} \left(\max \left(\bigcup_{i=1}^{i_{u_0\max}} k(u_0pi), 0 \right) - \min \left(\bigcup_{i=1}^{i_{u_0\max}} k(u_0pi), 0 \right) \right) + \\ &\left. + \sum_{u=1}^{u_0+1} \sum_{p=1}^{p_{u\max}} \max(\bigcup_{i=1}^{i_{u\max}} k(upi), 0) \right]. \end{aligned} \quad (8)$$

Наряду с сопоставлением различных конфигураций цикла управления по изложенной выше методике, задача снижения количества портов также может решаться за счет использования для связи элементов как прямых каналов связи, так и общей памяти. В этом случае прямые каналы связи формируются для передачи данных от различных датчиков или регуляторам исполнительных устройств (приводов). Такие элементы-акторы обычно связаны с единственным элементом-актором, выполняющим обработку данных (сортировку, вычисления, логические операции и т. д.). Для прочих взаимодействий элементов-акторов (например, различных модулей исполнительного блока) задействуется общая память. В случае использования память-ориентированной архитектуры системы управления (вычисления в памяти [7, 8] и вблизи памяти [9]) через общую память передаются данные небольшого объема (задания и результаты), что в значительной мере нивелирует проблему очередей и задержек [10] в передаче данных. Методология решения оптимизационной задачи в этом случае изменяется незначительно. Изменение заключается в исключении из суммарного количества портов прямой передачи данных тех из них, которые заменяются подключением к общей памяти.

Заключение

Резюмируем проведенное в статье исследование:

1. Современное высокотехнологичное оборудование требует высокой точности управления, для обеспечения которой необходимы системы управления с очень малой длительностью цикла, за который выполняется комплекс необходимых операций управления.

2. Указанный комплекс необходимых операций управления целесообразно описывать в рамках акторной модели в виде цикла с определенной конфигурацией, т. е. распределением элементов-акторов по группам, потокам и последовательностью в каждом потоке. В результате задача оптимизации управления сводится к конфигурированию цикла управления, т. е. определению конфигурации, обеспечивающей заданные требования: суммарную длительность цикла управления или ограничения по (вычислительным) ресурсам.

3. Дополнительным фактором оптимизации является портовость элементов-акторов, т. е. число портов ввода и вывода, используемых для передачи данных между элементами-акторами. Проведенные исследования показывают, что для обеспечения полной функциональности системы управления не требуется попарно связывать прямыми каналами все элементы-акторы. Одни и те же порты ввода или вывода элемента-актора могут использоваться для связи с несколькими другими. Кроме того, при использовании память-ориентированной архитектуры системы управления возможно применение наряду с прямыми каналами связи также общей высокоскоростной памяти для передачи коротких сообщений.

СПИСОК ИСТОЧНИКОВ / REFERENCES

1. Красильникъянц Е.В., Бурков А.П., Иванков В.А., Булдукян Г.А., Ельниковский В.В., Варков А.А. Системы управления движением технологических объектов. *Вестник Ивановского государственного энергетического университета*. 2007;(4):42–46.
Krasilnikyants E.V., Burkov A.P., Ivankov V.A., Buldukyan G.A., Elnikovskij V.V., Varkov A.A. Technological objects traffic control systems. *Vestnik of Ivanovo State Power Engineering University*. 2007;(4):42–46. (In Russ.).
2. Меркурьев И.В., Комерзан Е.В., Свириденко О.В., Лабахуа Л.Р. Методы повышения быстродействия и точности систем навигации и управления движением автоматических робототехнических средств. *Системные технологии*. 2018;(3):99–104.
Merkuriev I.V., Komerzan E.V., Sviridenko O.V., Labahua L.R. Methods to improve the speed and accuracy of navigation systems and motion control of automatic robotic tools. *Sistemnye tekhnologii*. 2018;(3):99–104. (In Russ.).
3. Зеленский А.А., Грибков А.А. Акторное моделирование когнитивных систем реального времени: онтологическое обоснование и программно-математическая реализация. *Философская мысль*. 2024;(1):1–12. <https://doi.org/10.25136/2409-8728.2024.1.69254>
Zelenskii A.A., Gribkov A.A. Actor modeling of real-time cognitive systems: ontological basis and software-mathematical implementation. *Philosophical Thought*. 2024;(1):1–12. (In Russ.). <https://doi.org/10.25136/2409-8728.2024.1.69254>
4. Burgin M. Systems, Actors and Agents: Operation in a multicomponent environment. arXiv. URL: <https://arxiv.org/abs/1711.08319> [Accessed 3rd March 2025].

5. Зеленский А.А., Грибков А.А. Конфигурирование память-ориентированной системы управления движением. *Программные системы и вычислительные методы*. 2024;(3):12–25. <https://doi.org/10.7256/2454-0714.2024.3.71073>
Zelenskii A.A., Gribkov A.A. Configuration of memory-oriented motion control system. *Software systems and computational methods*. 2024;(3):12–25. (In Russ.). <https://doi.org/10.7256/2454-0714.2024.3.71073>
6. Knuth D.E. *The Art of Computer Programming. Volume 3. Sorting and Searching. Second Edition*. Addison Wesley Longman; 1998. 780 p.
7. Каляев И., Заборовский В. Искусственный интеллект: от метафоры к техническим решениям. *Control Engineering Россия*. 2019;(5):26–31.
8. Mutlu O. Memory-Centric Computing. arXiv. URL: <https://doi.org/10.48550/arXiv.2305.20000> [Accessed 3rd March 2025].
9. Ke L., Zhang X., So J., et al. Near-Memory Processing in Action: Accelerating Personalized Recommendation with AxDIMM. *IEEE Micro*. 2022;42(1):116–127. <https://doi.org/10.1109/MM.2021.3097700>
10. Suh S.-H., Kang S.K., Chung D.-H., Stroud I. *Theory and Design of CNC Systems*. London: Springer; 2008. 456 p. <https://doi.org/10.1007/978-1-84800-336-1>

ИНФОРМАЦИЯ ОБ АВТОРАХ / INFORMATION ABOUT THE AUTHORS

Зеленский Александр Александрович, кандидат технических наук, доцент, ведущий научный сотрудник Научно-производственного комплекса «Технологический центр», Москва, Зеленоград, Российская Федерация.
e-mail: zelenskyaa@gmail.com
ORCID: [0000-0002-3464-538X](https://orcid.org/0000-0002-3464-538X)

Aleksandr A. Zelenskii, Candidate of Engineering Sciences, Docent, Leading researcher, Scientific and Production Complex "Technological Center", Moscow, Zelenograd, the Russian Federation.

Грибков Андрей Армович, доктор технических наук, ведущий научный сотрудник Научно-производственного комплекса «Технологический центр», Москва, Зеленоград, Российская Федерация.
e-mail: andarmo@yandex.ru
ORCID: [0000-0002-9734-105X](https://orcid.org/0000-0002-9734-105X)

Andrei A. Gribkov, Doctor of Engineering Science, Leading researcher, Scientific and Production Complex "Technological Center", Moscow, Zelenograd, the Russian Federation.

Статья поступила в редакцию 28.02.2025; одобрена после рецензирования 17.03.2025; принята к публикации 20.03.2025.

The article was submitted 28.02.2025; approved after reviewing 17.03.2025; accepted for publication 20.03.2025.