ПЛАНИРОВАНИЕ РАЗМЕЩЕНИЯ ПАРАЛЛЕЛЬНЫХ ПРОГРАММ В МАСШТАБИРУЕМЫХ МУЛЬТИПРОЦЕССОРАХ

Борзов Д.Б. *(ЮЗГУ, г. Курск, Россия)*
Тел. +7(4712)58-71-05; E-mail borzovdb@kursknet.ru

Abstract: *We justify the need for specialized devices scheduling in a scalable multiprocessor systems, due to the need to improve fault tolerance in the event of an emergency. In connection with this proposed appropriate methods and algorithms for scheduling.*
Keywords*: processor, multiprocessor, allocation, program, planning, device.*

Мультипроцессоры (МП) – один из широко распространенных классов высокопроизводительных вычислительных систем [1]. Современные МП – системы с высокой степенью параллелизма, включающие десятки и сотни процессорных модулей, взаимодействующих через коммуникационную сеть. Перспективным направлением развития МП в части дальнейшего повышения их производительности, снижения энергопотребления и улучшения массо–габаритных характеристик является переход к масштабируемой реализации мультипроцессоров (ММ) [2]. ММ позволяет получить высокую эффективность межпроцессорных взаимодействий, снижая долю последовательной составляющей, упрощая программирование мультипроцессора, содержащего несколько десятков процессоров при поиске варианта размещения. Подобные ММ-системы (имеющие, как правило, матричную структуру) могут применяться как автономно, так и в качестве узлов крупномасштабных систем, например, мультикомпьютеров.

Вариантом решения вопроса повышения эффективности ММ-систем может быть оперативное переразмещение параллельных программ, но программное решение этого вопроса нецелесообразно в случае использования критических систем (системы наблюдения, слежения, бортовой авиации и т.п.) в связи с огромным количеством вариантов перебора. При этом использование программных средств также невозможно из-за длительного времени решения. В этом случае следует использовать специализированные аппаратные средства, для чего необходимо найти новые методы и подходы размещения параллельных программ ММ-системах.

Непосредственно перед размещением первоначально целесообразно выполнить разбиение последовательных программ на слабосвязные фрагменты с целью их дальнейшего по возможности независимого назначения на отдельные процессорные модули [3].

Исходная программа в последовательной форме представляется в виде $|Q|_i$ строк, $|Vi|_k$ входных переменных, $|Vo|_\kappa$ выходных переменных, где $i=\overline{1,N}$, $k=\overline{1,M}$, N – количество строк исходной последовательной программы, M – общее количество переменных. Каждой строки Q_i сопоставляются коэффициенты $|ki|_M$ и $|ko|_M$, показывающие наличие в ней входных переменных Vi и выходных переменных Vo, где $ki_p=\{0,1\}$, $ko_p=\{0,1\}$, $p=\overline{1,M}$.

Тогда для Q_i будет выполняться равенство:

$$ko_f \cdot Vo_k = ki_1 \cdot Vi_1 \Xi ki_2 \cdot Vi_2 \Xi \ldots \Xi ki_M \cdot Vi_M, \quad (1)$$

где Ξ - признак выполняемой операции, f – номер единичного коэффициента ko для i-й строки.

Исходная программа задается множеством участков $U^V = \{U_1^V, U_2^V, ..., U_Z^V\}$, характеризующихся разными значениями V уровней вложенности операторов, где $Z = \overline{1, N_u}$, $V = \overline{1, MV}$. Она задается матрицами:

– следования для участков $Ms\_U = \|Ms\_U\|_{N_u \times N_u}$,

где $Ms\_U_{zg} = \begin{cases} 1, & \text{если } U_z \ \varphi \ U_g \\ 0, & \text{в противном случае.} \end{cases}$, $g = \overline{1, N_u}$.

– входных переменных для участков $I\_U = \|I\_U\|_{N_u \times M}$,

где $|I\_U|_z = |I|_A \vee |I|_{A+1} \vee ... \vee |I|_B$.

– выходных переменных для участков $O\_U = |O\_U|_{N_u \times M}$,

где $|O\_U|_z = |O|_A \vee |O|_{A+1} \vee ... \vee |O|_B$.

– достижимости для участков $Md\_U = \|Md\_U\|_{N_u \times N_u}$. В ее ячейку записывается 1, если из участка U_z можно достигнуть участка U_g:

$$Md\_U_{zg} = \begin{cases} 1, & \text{если } U_z \mapsto U_g \\ 0, & \text{в противном случае.} \end{cases},$$

Если $U_z \mapsto U_g$, то существует подмножество U', такое что

$$U' \subseteq U, U_z \ \varphi \ U_1', U_1' \ \varphi \ U_2', ..., U_{N'-1}' \ \varphi \ U_{N'}', U_{N'}' \ \varphi \ U_g.$$

Выявление параллелизма между участками U_z и U_g программы одинакового уровня вложенности V можно свести к выявлению информационной независимости между этими участками, используя условие:

$$F\_U(z,g) = (I\_U_z \wedge O\_U_g) \vee (I\_U_g \wedge O\_U_z) \vee (O\_U_z \wedge O\_U_g) \qquad (2)$$

Выявление параллелизма между операторами R_i и R_k участка программы можно свести к выявлению информационной независимости между ними через условие:

$$F(i,k) = (I_i \wedge O_k) \vee (I_k \wedge O_i) \vee (O_i \wedge O_k) \qquad (3)$$

Полный набор информационных зависимостей между всеми участками программы задается матрицей неполного параллелизма $MNP\_U = \|MNP\_U\|_{N_u \times N_u}$. Пользуясь (2), получаем

$$MNP\_U_{zg} = \begin{cases} 0, & \text{если } (Md\_U_{zg} = 0) \vee ((Md\_U_{zg} = 1) \wedge (F\_U(z,g) = 0), \\ 1, & \text{если } (Md\_U_{zg} = 1) \wedge (F\_U(z,g) = 1). \end{cases}$$

Полный набор информационных зависимостей между всеми операторами программы задаётся матрицей неполного параллелизма $MNP = \|MNP\|_{N \times N}$. Используя (3), получаем

$$MNP_{ik} = \begin{cases} 0, & \text{если } (Md_{i,k} = 0) \vee ((Md_{i,k} = 1) \wedge (F(i,k) = 0)), \\ 1, & \text{если } (Md_{i,k} = 1) \wedge (F(i,k) = 1). \end{cases}$$

На основе вышеприведённой формализации разработан метод распараллеливания программ, состоящий из следующих этапов:

1. Преобразование матриц входных и выходных переменных, а также матрицы следования для операторов программы к аналогичным матрицам для участков программы.

2. Вычисление бинарной матрицы достижимости для участков программы на основе матрицы следования.

3. Вычисление бинарной матрицы неполного параллелизма на основе матриц достижимости, входных и выходных переменных.

4. Преобразование множества U участков программы в последовательной форме к множеству участков Up программы в параллельной форме с помощью матрицы неполного параллелизма.

После разбиения, на следующем этапе выполняется этап планирования размещения параллельных программ в ММ-системах.

Пакет взаимодействующих программ, запланированных к обработке в базовом блоке, представлен графом взаимодействия задач вида:

$$\text{G=<X,E>, где } X=\{x_1,x_2,\dots x_g,\dots x_N\} - \quad (4)$$

множество вершин графа G, вершины $x_g \in X$ соответствуют программам, а дуги $e_{ij} \in E$ – связям между ними, взвешиваемыми объемами данных m_{ij}, передаваемыми между задачами и сводятся в матрицу обмена информацией (МОИ) $M=\|m_{ij}\|_{N\times N}$, где $N=n^2=|X|$, $i,j=\overline{1,N}$.

Матричный блок представлен топологической моделью в виде графа H=<P,V>, где $P=\{p_1,p_2,\dots,p_{n^2}\}$ – множество идентификаторов процессорных модулей блока, организованных в матрицу $|P|_{n\times n}$, где $|P|=N=n^2$ - процессоров; V – множество связей, описываемых матрицей смежности $\|W\|_{N\times N}$ размером $n^2\times n^2$.

Размещение пакета программ (задач), описываемых графом G (4) аналитически описывается отображением:

$$\beta_S=\begin{pmatrix} x_1^{(S)} & x_2^{(S)} & \dots & x_N^{(S)} \\ p_1 & p_2 & \dots & p_N \end{pmatrix}, \quad (5)$$

где s – это номер очередной перестановки по процессорным модулям p_N, соответствующий s-у варианту размещения. Мощность множества $\psi=\{\beta_S\}$ всевозможных отображений (5) равна числу всевозможных перестановок задач $x_n^{(s)}$ в матрице $X: |\psi|=N!$. Введем матрицу минимальных расстояний (ММР) $D=\|d_{ij}\|_{N\times N}$, $N=n^2=|P|$, которую можно построить по матрице смежности, необходимую для отображения длин d_{ij} кратчайших маршрутов передачи данных.

Планирование размещения сводится к поиску максимальной коммуникационной задержки при обмене между процессорными модулями $p_{a,b}$ и $p_{x,y}$, соответствующей отображению $\beta^*\in\Psi$ и вычислению показателя $A=\max\limits_{\beta_S\in\psi}\{T_{\beta_S}(p_{a,b},p_{x,y})\}$ с последующей его минимизацией. В результате:

$$T_{\beta^*}=\min_{\psi}\{A\}. \quad (6)$$

Отображение β_s вычисляется как произведение

$$T_{\beta_s}\left(p_{a,b}, p_{x,y}\right) = d_{ij} \cdot m_{ij}, \tag{7}$$

где $i = (a-1)\cdot n + b$ и $j = (x-1)\cdot n + y$,

$$\text{а} \max_{\beta_s \in \psi}\left\{T_{\beta_s}\left(p_{a,b}, p_{x,y}\right)\right\} = \max\left\{d_{ij} m_{ij}\right\}. \tag{8}$$

Поиск варианта размещения по (6) сложная переборная задача, в итоге сходящаяся к n! перестановкам, что для критических систем неприемлемо. Следовательно возникает необходимость сокращения количества переборных перестановок. Для этого были предложены дополнительные критерии, ускоряющие время поиска.

$$d_{\alpha k} < d_{\alpha\beta}, \tag{9}$$

где $d_{\alpha k}, d_{\alpha\beta}$ – одноименные элементы матрицы ММР; $m_{\alpha\beta}$ – элемент МОИ, которому соответствует $max\left\{m_{ij} \cdot d_{ij}\right\}$, найденный в предыдущем шаге перестановок.

$$m_{\alpha k} \cdot d_{\alpha k} < m_{\alpha\beta} \cdot d_{\alpha\beta}, \tag{10}$$

$$m_{\alpha k} < m_{\alpha\beta}. \tag{11}$$

Разработана методика ускорения выполнения процедур планирования размещения программ, которая начинается с вычисления недостижимости минимальной оценки размещения T_{inf} при допущении, что топологии графов G и H тождественны. Соответствующая методика включает следующие шаги: 1. Составляются две матрицы: обмена информацией между задачами (МОИ) и кратчайших маршрутов (ММР) между процессорами в коммуникационной среде базового блока. 2. Вычисляются минимум коммуникационной задержки T_{inf} и коэффициент эффективности исходного произвольного размещения задач $\eta_н$. 3. По порогу эффективности $\eta_н > 2$ принимается решение о целесообразности инициализации процедуры поиска субоптимального размещения. Коэффициент эффективности перестановок $\eta = T / T_{\text{inf}}$ определяется как отношение полученной величины задержки (5) к T_{inf}. 4. Выполняются шаги целенаправленных перестановок столбцов и строк матрицы обмена информацией. Находится максимальное значение коммуникационной задержки (5) по предыдущему варианту перестановок программ. 5. Находится минимум (3) из максимумов задержек по всем вариантам перестановок и вычисляется коэффициент эффективности η. 6. Если η оказывается менее установленного порога эффективности $\eta \le 2$, шаги поиска прекращаются и найденный вариант матрицы обмена информацией считается соответствующим полученному варианту размещения.

С другой стороны в ММ-системах могут возникать ситуации отказов внутренних процессорных модулей и межпроцессорных связей. В этом случае при условии использования критических систем реакция со стороны ММ-системы должна быть сведена к минимуму. С этой целью был предложен следующий подход.

Множество подпрограмм каждой программы описывается графом взаимодействия задач:

$$\Phi = \left\langle X, E \right\rangle, \tag{12}$$

где $X = \left\{x_i\right\}$ – множество вершин, которые соответствуют отдельным подпрограммам, $E = \left\{e_{ij}\right\}$ – множество дуг, отражающих связи между подпрограммами. Граф Ф представляется матрицей смежности вершин: $\text{AM} = \left\|m_{ij}\right\|_{N\times N}$, где $N = \left|X\right|$. Значение

элемента m_{ij} определяется числом сообщений (объемом данных), передаваемых между подпрограммами x_i и x_j.

Топология мультипроцессора задается графом $H = \langle P_1, V \rangle$, где множество вершин P_1 соответствует процессорным модулям, а множество дуг V – межмодульным связям. Множество P_1 разбивается на два непересекающихся подмножества $P_1 = P \cup L$, где $P = \{p_{ij}\}$ – множество основных процессоров, $L = \{l_{ij}\}$ – множество резервных процессоров, причем фиксируется $|P| = |L| = n^2$, $n = 2,3,4,\ldots$. Упорядочим множества процессоров P и L в виде матриц $\|p_{ij}\|_{n\times n}$ и $\|l_{ij}\|_{n\times n}$ соответственно.

Размещение множества взаимосвязанных подпрограмм, описываемого графом Ф, в мультикомпьютере задается отображением

$$\beta : X \to P_1, \tag{13}$$

которое ставит в соответствие каждой программе один из процессоров (основной либо резервный).

Для описания множества кратчайших маршрутов передач данных вводится матрица длин $\mathrm{LM} = \|d_{ij}\|_{N\times N}$, элемент d_{ij} которой численно равен длине кратчайшего маршрута между процессорами, в которых размещены подпрограммы x_i и x_j.

Пусть Υ – множество всевозможных отображений вида (13). Тогда задача размещения программ в мультипроцессоре будет заключаться в выборе такого отображения $\beta_s \in \Upsilon$, которое соответствует следующему критерию:

$$\varsigma_s = \min_{\beta\in\Upsilon}\left\{ \max_{i=1,N,\,j=1,N}\{d_{ij}\cdot m_{ij}\}\right\}, \tag{14}$$

где максимум в фигурных скобках представляет собой наибольшую частную коммуникационную задержку для заданного отображения β, а минимум – последующую ее минимизацию.

Например, при отказе процессора $p_{1.2}$ выполняется перебор отображений вида:

$$\begin{Bmatrix} x_{s_{1.}}\, x_{s_{1.2}} \ldots x_{s_{1.\upsilon}} \ldots x_{s_{1.n}} \\ x_{s_{2.1}}\, x_{s_{2.2}} \ldots x_{s_{2.\upsilon}} \ldots x_{s_{2.n}} \\ \ldots \\ x_{s_{q.1}}\, x_{s_{q.1}} \ldots x_{s_{q.\upsilon}} \ldots x_{s_{q.n}} \\ \ldots \\ x_{s_{n.1}}\, x_{s_{n.2}} \ldots x_{s_{n.\upsilon}} \ldots x_{s_{n.n}} \end{Bmatrix} \to \begin{Bmatrix} p_{1.1} l_{1.1} \boxtimes l_{1.2} \ldots p_{1.\upsilon} l_{1.\upsilon} \ldots p_{1.n} l_{1.n} \\ p_{2.1} l_{2.1}\, p_{2.2} l_{2.2} \ldots p_{2.\upsilon} l_{2.\upsilon} \ldots p_{2.n} l_{2.n} \\ \ldots \\ p_{q.1} l_{q.1}\, p_{q.2} l_{q.2} \ldots p_{q.\upsilon} l_{q.\upsilon} \ldots p_{q.n} l_{q.n} \\ \ldots \\ p_{n.1} l_{n.1}\, p_{n.2} l_{n.2} \ldots p_{n.\upsilon} l_{n.\upsilon} \ldots p_{n.n} l_{n.n} \end{Bmatrix}, \tag{15}$$

где $\boxtimes$ показывает ситуацию отказа процессора $p_{1.2}$.

В этом случае учитывается, что отказавший процессор замещается резервным, например, процессором $l_{1,2}$ и изменяется множество допустимых маршрутов передачи данных. Для этого была предложена процедура замещения отказавшего процессорного модуля резервным, формализующаяся следующим образом.

Состояние основных процессоров $p_{ij} \in P$ описывается матрицей $Z = \|z_{ij}\|_{n\times n}$:

$$z_{ij} = \begin{cases} 1, \text{ если } p_{ij} \text{ неисправен;} \\ 0, \text{ если } p_{ij} \text{ исправен.} \end{cases}$$

Состояние резервных процессоров $l_{ij} \in L$ описывается матрицей $\Theta = \left\| \theta_{ij} \right\|_{n \times n}$:

$$\theta_{ij} = \begin{cases} 1, \text{ если } l_{ij} \text{ неисправен;} \\ 0, \text{ если } l_{ij} \text{ исправен.} \end{cases}$$

Матрица $K = \left\| k_{ij} \right\|_{n \times n}$ содержит признаки доступности ближайшего резервного модуля к процессору $p_{ij} \in P$ для замены:

$$k_{ij} = \begin{cases} 1, \text{ если } l_{ij} \text{ недоступен;} \\ 0, \text{ если } l_{ij} \text{ доступен.} \end{cases}$$

Тогда с учетом введенных обозначений процедура замещения отказавшего процессорного модуля резервным будет состоять из следующих шагов:

ЗАГРУЗИТЬ матрицы AM и LM;

УСТАНОВИТЬ $z_{ij} = \theta_{ij} = k_{ij} = 0, i = \overline{1,n}, j = \overline{1,n}$;

ЦИКЛ по всем процессорам $p_{ij} \in P$

ЕСЛИ $\exists p_{ij} \in P : z_{ij} = 1$, то найти в матрице Θ элемент $\theta_{st} = 0$ такой, что $\left(\left|i - s\right| \le 1\right) \wedge \left(\left|j - t\right| \le 1\right)$;

ЕСЛИ такой θ_{st} найден, то принять $\theta_{st} = 1$, $l_{st} = p_{ij}$ (замещение успешно), иначе необходим останов системы;

КОНЕЦ ЦИКЛА

В случае отказа канала связи нарушаются маршруты транзитной передачи данных и необходимо найти новые кратчайшие маршруты обхода отказавшего канала. Для этого случая обоснована целесообразность использования алгоритма Дейкстры.

Для предложенных методов и аппаратно-ориентированных алгоритмов планирования размещения параллельных программ в масштабируемых мультипроцессорах была предложена структурная организация, обосновывающая целесообразность предложенных подходов, представленная на рисунке 1.

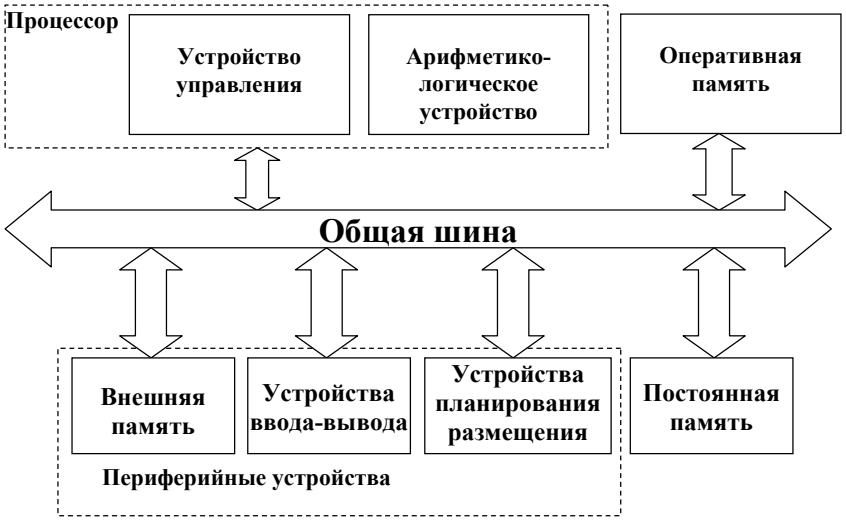

Рис. 1. Применение специализированных средств планирования размещения

Предложенные аппаратные средства могут быть использованы при выполнении задач разбиения последовательных программ, планирования размещения в мультикомпьютерных системах и/или планирования размещения параллельных программ в отказоустойчивых масштабируемых мультипроцессорных системах.

Разработаны функциональные схемы специализированных аппаратных средств планирования размещения в ММ-системах, представленные на рисунках 2,3.

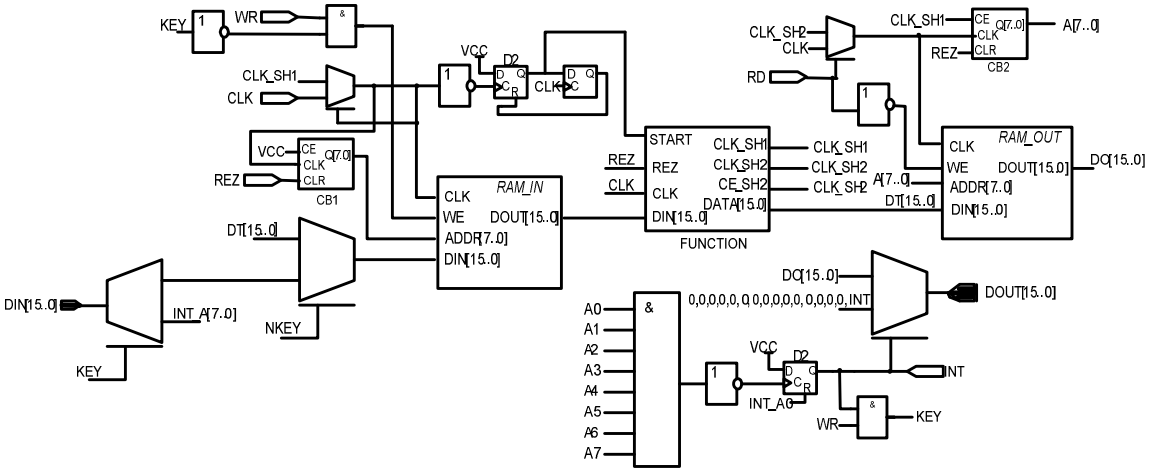

Рис. 2. Функциональная схема типового вычислительного блока устройства разбиения

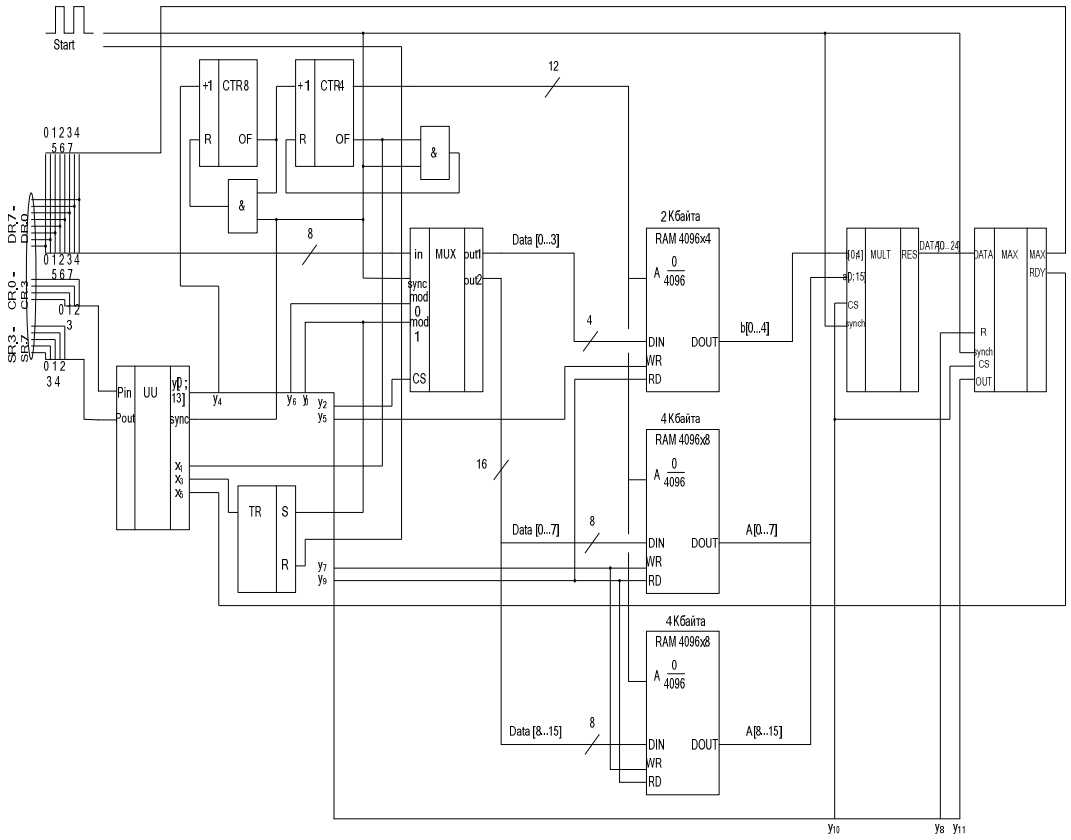

Рис. 3. Функциональная схема акселератора вычисления показателя коммуникационной задержки

Здесь на рисунке 2 используются элементы**:** входное RAMIN и выходное RAMOUT ОЗУ для сохранения заданий, поступающих от микропроцессорного ядра. Такая структура позволяет блокам параллельно выполнять отдельные стадии разбиения участков программы. Также типовой блок включает интерфейс взаимодействия с микропроцессором и схему генерации прерывания. Процессор считывает результаты выполнения заданий из выходных ОЗУ типовых блоков по поступающим прерываниям, организованным в векторную систему.

Самую частую операцию вычисления максимальной коммуникационной задержки (рис. 3) предполагается выполнять в аппаратном ускорителе: акселераторе вычисления показателя коммуникационной задержки (АВПКЗ). Блок АВПКЗ отличается тем, что в нем применены конвейерный и матричный подходы для поэлементного перемножения векторов с одновременным подсчетом максимального произведения. В составе АВПКЗ можно выделить шесть основных функциональных блоков (рис. 3). Данные с параллельного порта поступают в блок специализированного демультиплексора (DMX). В зависимости от режима работы DMX загружает одно из ОЗУ данными соответствующей разрядности. Если идет загрузка в $ОЗУ_2$, из 8-ми разрядного порта за один цикл приема MUX принимает два 4-х разрядных слова. Если же идет загрузка в $ОЗУ_1$, то MUX принимает два байта 16-ти разрядного слова и после их склеивания помещает целое слово в $ОЗУ_1$. Блок умножения матричной–конвейерный (БУМК) осуществляет перемножение синхронно считанных из $ОЗУ_1$ и $ОЗУ_2$ двух слов и выдает результат в блок нахождения максимума (MAX). Умножение происходит конвейерно за один такт. Блок MAX находит максимум и по сигналу об окончании расчета выдает результат за три цикла вывода в порт контроллера. Устройство управления (УУ) выдает управляющие сигналы, обозначенные на рис. 3 как множество микроопераций (МО), а также значения адресов для $ОЗУ_1$ и $ОЗУ_2$ в режимах загрузки и вычисления.

Для структурной организации ММ-системы, представленной на рисунке 1, были предложены специализированные устройства планирования размещения, аналогичные представленным на рисунках 2 и 3, которые подробно изложены в [4]

В дальнейших исследованиях предполагается исследование подходов к размещению параллельных программ для программируемых систем на кристалле (ПЛМ), как подобных, представленным в данной работе.

Список литературы: 1. Цилькер Б.Я., Орлов С.А. Организация ЭВМ и систем: учебник для вузов / СПб.: Питер, 2004. 668 с. **2.** Данильченко Н.В., Макеев С.Н. Масштабируемый мультипроцессор для цифровой обработки сигналов / МЭС-2012. Носсия. Москва. октябрь 2012. **3**. Борзов Д.Б., Дюбрюкс С.А., Миронов Д.А. Устройство обеспечения распараллеливающей компиляции последовательных программ для вычислительных систем / Сборник трудов XVIII Международной научно-технической конференции «Машиностроение и техносфера XXI века. Т1». – Донецк, 2012. – С. 95-98. **4**. Борзов Д.Б., Титов В.С. Параллельные вычислительные системы (архитектура, принципы размещения задач) / Монография. – Курс. гос. тех. ун-т. Курск, 2009. 159 с.