УДК 004.383.3

В.Н. Ручкин, В.А. Романчук

АЛГОРИТМЫ АНАЛИЗА ВЫЧИСЛИТЕЛЬНЫХ СТРУКТУР НА БАЗЕ НЕЙРОПРОЦЕССОРОВ

Разрабатываются алгоритмы анализа интеллектуальных вычислительных структур на базе отечественных нейропроцессоров семейства NM 640X в результате анализа автоматных моделей связей элементов вычислительных структур с целью определения вида структуры нейропроцессорной системы и предлагаются программные средства комплекса «НейроКС» реализации нового класса интеллектуальных вычислительных систем.

Ключевые слова: *нейропроцессор, нейропроцессорная система, интеллектуальная вычислительная структура, алгоритмы, автоматные модели связей, программный комплекс «НейроКС»*

Введение

В настоящее время для микропроцессоров наступает так называемый ”технологический предел”, означающий что они достигли максимального уровня повышения быстродействия. Все разработки в данное время направлены на повышение числа процессоров на кристалле. Одним из выходов из данной ситуации является новая элементная база, например использование нейрокомпьютеров. В области нейрокомпьютеров в настоящее время ведутся разработки с использованием новых технологий, перспективными можно назвать технологии создания оптических, молекулярных и нанонейрокомпьютеров [1-2].

Вычислительные системы на базе нейропроцессоров – нейропроцессорные системы (НПС) отличаются высокой эффективностью при их использовании вследствие следующих причин:

- алгоритмы нейроинформатики высокопараллельны за счет использования нейросетевого базиса;
- НПС можно обучить устойчивости к помехам и в дальнейшем к самообучению.

Но для дальнейшего развития области нейропроцессорных технологий имеется ряд проблем, основными из которых являются [3]:

1. Невысокая частота нейрочипов (30-150 МГц) не позволяет получить конкурирующую с обычными микропроцессорами производительность нейропроцессорных устройств.
2. Недостаточное программное обеспечение для нейропроцессоров по сравнению с общераспространенными микропроцессорами.
3. Коммерческая недоступность (секретность) информационных материалов в данной области.
4. Слишком большая цена перехода от существующих процессоров к нейропроцессорам (изменение не только аппаратных, но и программных средств).

Одним из способов решения первой проблемы является организаци***я интеллектуальных вычислительных структур*** [4] . В настоящее время в области нейропроцессорных технологий ведутся исследования в части многопроцессорности, уже разработаны модули, включающие несколько процессоров с различными связями. Однако имеются проблемы, мешающие созданию эффективных мультимикропроцессорных структур на базе нейропроцессоров, одной из которых является проблема описания связей элементов НПС и описание различных структур НПС исходя их описания связей ее элементов.

Цель работы

Целью работы является разработка алгоритмов описания интеллектуальных вычислительных структур на базе отечественных нейропроцессоров.

Постановка задачи

В соответствии с целью работы были определены две задачи: разработка алгоритмов определения связей элементов интеллектуальных вычислительных структур на базе нейропроцессоров и разработка алгоритмов определения вида структуры НПС на основе описания связей ее элементов.

Для дальнейших исследований, в качестве примера было выбрано семейство нейропроцессоров NM640X, по следующим причинам:

- Процессоры семейства NM640X обладают функциональными возможностями, наиболее полно отражающими принципы функционирования всего класса нейропроцессоров.
- Информация о процессе функционирования размещена в открытом доступе и содержит необходимый для дальнейшего исследования теоретический материал.

Результаты исследования могут быть адаптированы и для других нейропроцессоров, соответствующих общим принципам функционирования.

Теоретические исследования

Разработка алгоритмов определения связей элементов вычислительных структур на базе нейропроцессоров

Нейропроцессорная система состоит из множества элементов – процессорных модулей (ПМ). Под процессорным модулем будем понимать нейромикропроцессор и ряд вспомогательных интегральных схем.

Результаты определения связей элементов НПС будем представлять в виде матрицы связи ПМ $M = [M_{ij}]$. Ее размерность $q \times q$, где q - число ПМ. Элементами матрицы M являются: ‘0’ - нет связи между ПМ; ‘1’ - есть связь между ПМ (в подпрограмме с большим порядковым номером необходимы данные подпрограммы с меньшим порядковым номером); ‘X’ - запрещенные ячейки (между подпрограммами не может быть связей).

Для того чтобы определить элементы матрицы M введем вспомогательную матрицу $M' = [M'_{ij}]$ размерности $q \times E$, в которой число столбцов E - это количество элементов процессора. Для нейропроцессоров семейства NM640X $E = 44$ (дополнительно заносятся области памяти, занятые переменными). Элементами матрицы M' являются: ‘0’ - элемент не использовался в данной подпрограмме; ‘1’ - элемент был присвоен в данной подпрограмме; ‘2’ - элемент был использован в данной подпрограмме; ‘3’ - элемент был использован, а затем присвоен в данной подпрограмме.

Случай, когда элемент сначала присваивается в подпрограмме, а затем используется - не рассматривается, т.к. важен только случай присваивания элемента в текущей подпрограмме, т.е. случай с обозначением ‘1’.

Данная матрица описывает состояние каждого элемента процессора в каждой подпрограмме и позволяет определить элемент матрицы M_{ij} следующим образом:

$$((M'_{ik} = 1) \wedge (M'_{jk} = 2) \vee (M'_{ik} = 3)) \vee$$
$$((M'_{nk} = 1) \vee (M'_{nk} = 3)) \rightarrow M_{ij} = 1; n = \overline{i+1, j};$$
$$((M'_{ik} = 3) \wedge (M'_{jk} = 2) \vee (M'_{ik} = 3)) \wedge .$$
$$((M'_{nk} = 1) \vee (M'_{nk} = 3)) \rightarrow M_{ij} = 1; n = \overline{i+1, j}.$$

Тогда целью алгоритма является заполнение элементов матрицы связей ПМ для дальнейшего определения вида структуры. На входе имеем кортеж подпрограмм, например $PR = < RO_1, RO_2, RO_5, RO_1, ... >$. На выходе - матрица связи ПМ M и вспомогательная матрица связи M' с описанием связей между элементами НПС.

Для того чтобы определить состояние элемента процессора R в подпрограмме P был использован математический аппарат конечных автоматов. Пусть $K = (S, Z, W, \delta, \lambda, S_1)$ - конечный автомат описания (рисунок 1), где

S - множество состояний автомата K;

Z - множество входных сигналов автомата K;

W - множество выходных сигналов автомата K;

δ - функция переходов автомата K;

λ - функция выходов автомата K;

S_1 - начальное состояние автомата K.

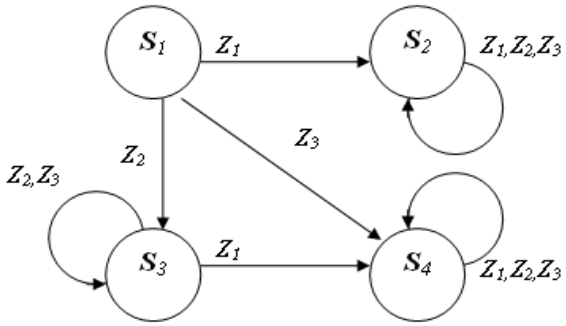

Рисунок 1 – Конечный автомат K определения состояния элемента ПМ R

Множество состояний S автомата K: S_1 - состояние M'_{RP}=0; S_2 - состояние M'_{RP}=1; S_3 - состояние M'_{RP}=2; S_4 - состояние M'_{RP}=3;

Входной алфавит Z автомата K: Z_1 - элементу R было присвоено значение; Z_2 - значение элемента R было использовано; Z_3 - значение элемента R было сначала использовано, потом присвоено.

Пусть q - число ПМ, E - число элементов процессора. Рассмотрим *алгоритм заполнения матрицы связей ПМ*. Целью алгоритма является получение значений $M_{ij}; i = \overline{1, q}; j = \overline{1, q}$.

Шаг 1. Если $p \geq q$, то переход на шаг 4.

Шаг 2. Если $i \geq E$, то $p = p + 1$, переход на шаг 1.

Шаг 3. Выборка микрокоманды MK_j. Заполнение элементов матрицы M' (алгоритм рассматривается далее), $i = i + 1$, переход на шаг 2.

Шаг 4. Если $p \geq q$, то переход на шаг 7.

Шаг 5. Если $i \geq q$, то $p = p + 1$ и переход на шаг 4.

Шаг 6. Заполнение элементов матрицы M (алгоритм рассматривается далее), $i = i + 1$ и переход на шаг 5.

Шаг 7. Конец алгоритма.

Рассмотрим *алгоритм, необходимый для того, чтобы определить значения элементов матрицы* M'. Целью алгоритма является получение значений $M'_{in}; i = \overline{1,q}; n = \overline{1,E}$.

Для реализации алгоритмов используется понятие лексемы Lex – последовательности допустимых символов языка нейроассемблера, имеющей смысл для транслятора. Алгоритмы распознавания лексем Lex приведены далее.

Далее показан *алгоритм распознавания лексем для языка нейроассемблера*, используемого для семейства нейропроцессоров NM640X. На входе алгоритма некоторая микрокоманда MK_i из кортежа $PR =< MK_1, MK_2, ..., MK_m, ..., MK_M >$. На выходе элемент матрицы M' для микрокоманды MK_i.

Шаг 1. Разделение микрокоманды MK_i на левую и правую часть.

Шаг 2. Если микрокоманда скалярная, то переход на шаг 12.

Шаг 3. Если в отладочном коде MK_i нет лексемы wtw, то на шаг 5.

Шаг 4. Заполнение элементов матрицы M', связанных с операцией wtw.

Шаг 5. Если в отладочном коде MK_i нет лексемы ftw, то переход на шаг 7.

Шаг 6. Заполнение элементов матрицы M', связанных с операцией ftw.

Шаг 7. Если в отладочном коде MK_i нет лексемы $vregs$, то на шаг 9.

Шаг 8. Заполнение элементов матрицы M', связанных с операцией $vregs$.

Шаг 9. Если в отладочном коде MK_i нет лексем выборки памяти, то шаг 11.

Шаг 10. Заполнение элементов матрицы M', связанных с операцией wtw.

Шаг 11. Заполнение элементов матрицы M', связанных с микрокомандой MK_i. Переход на шаг 23.

Шаг 12. Если в отладочном коде MK_i нет лексем перехода, то на шаг 14.

Шаг 13. Заполнение элементов M', связанных с обработкой перехода.

Шаг 14. Если в отладочном коде MK_i нет лексемы $pswr$, то на шаг 16.

Шаг 15. Заполнение элементов M', связанных с обработкой блока $pswr$.

Шаг 16. Если в отладочном коде MK_i нет лексем выборки памяти, то на шаг 18.

Шаг 17. Заполнение элементов M', связанных с обработкой памяти.

Шаг 18. Если в отладочном коде MK_i нет лексем изменения флагов N,Z,V,C, то переход на шаг 20.

Шаг 19. Заполнение элементов M', связанных с обработкой флагов.

Шаг 20. Если в отладочном коде MK_i нет лексем изменения регистров $ar0..ar7, gr0..gr7$, то переход на шаг 22.

Шаг 21. Заполнение элементов M', связанных с обработкой регистров.

Шаг 22. Заполнение элементов матрицы M', связанных с микрокомандой MK_i.

Шаг 23. Конец алгоритма.

Обработка микрокоманды реализуется с помощью конечного автомата.

Пусть: $Z = \{1,2,3\}$ - входящий сигнал конечного автомата,

$C = \{1,2,3\}$ - текущее состояние элемента процессора.

Одним из шагов разрабатываемого алгоритма является шаг заполнения матрицы M. Для его реализации необходимо использовать значения элементов матрицы связей ПМ M' и определить отображение

$$\varphi: M_{ij} \to M'_{nj}; i = \overline{1,q}; j = \overline{1,q}; n = \overline{1,E}.$$

Пусть: k - элемент процессора (для семейства процессоров NM640X k =44)

$M'_{ij} = \{1,2,3\}$ - текущее состояние элемента матрицы M'.

$M_{ij} = \{0,1\}$ - текущее состояние элемента матрицы M.

Тогда можно описать *алгоритм определения значения элементов матрицы* M *исходя из значений элементов матрицы* M', который является реализацией конечного автомата K, показанного на рисунке 1. На входе алгоритма множество элементов матрицы M'. На выходе множество элементов матрицы M.

Шаг 1. Начальная инициализация элемента $M_{ij} = 0$.

Шаг 2. Если $k \geq E$, то переход на шаг 12.

Шаг 3. Если $j \le i$, то $M_{ij} = 'X'$, переход на шаг 12.

Шаг 4. Если $(M'_{ik} = 1) \wedge (M'_{jk} = 2) \vee (M'_{ik} = 3)$, то переход на шаг 8.

Шаг 5. Если $(n > i+1) \wedge (n < j)$, то переход на шаг 7.

Шаг 6. Если $(M'_{nk} = 1) \vee (M'_{nk} = 3); n = \overline{i+1, j}$, то $flag = true$, $n = n+1$, на шаг 5.

Шаг 7. Если $flag = false$, то $M_{ij} = 1$.

Шаг 8. Если $(M'_{ik} = 3) \wedge (M'_{jk} = 2) \vee (M'_{ik} = 3)$, то $k = k+1$, переход на шаг 3.

Шаг 9. Если $(n > i+1) \wedge (n < j)$, то переход на шаг 11.

Шаг 10. Если $(M'_{nk} = 1) \vee (M'_{nk} = 3); n = \overline{i+1, j}$, то $flag = true$, $n = n+1$, на шаг 9.

Шаг 11. Если $flag = false$, то $M_{ij} = 1$.

Шаг 12. Конец алгоритма.

Таким образом, имеем матрицу M размерности $q \times q$, где q - число ПМ в НПС. Она содержит элементы, обозначающие зависимости ПМ по всем элементам процессора семейства NM640X. На основе полученной матрицы связей между ПМ можно определить вид нейропроцессорной структуры.

Разработка алгоритмов определения вида структуры НПС на основе описания связей ее элементов

На данном этапе необходимо определить вид многопроцессорной структуры исходя из матрицы связей ПМ. Для этого проверяется соответствие полученной матрицы связей ПМ всем признакам матриц для каждого типа архитектур. Если данная матрица не обладает всеми признаками ни одного из этих типов архитектур, то НПС является структурой произвольного вида. На вход алгоритма поступает матрица связей M, полученная ранее. На выходе имеем флаги вида структуры ($FKonv, FVect, VKV, FVK, FN$) - значения, равные *true,* если НПС имеет структуру конвейерного, векторного, векторно-конвейерного, конвейерно-векторного и произвольного типа соответственно.

Шаг 1. Определение соответствия матрицы всем признакам матрицы для конвейерной архитектуры.

Шаг 2. Если матрица обладает всеми признаками конвейерной архитектуры, то $FKonv = true$, переход на шаг 10.

Шаг 3. Определение соответствия матрицы всем признакам матрицы для векторной архитектуры.

Шаг 4. Если матрица обладает всеми признаками векторной архитектуры, то $FVect = true$, переход на шаг 10.

Шаг 5. Определение соответствия матрицы всем признакам матрицы для конвейерно-векторной архитектуры.

Шаг 6. Если матрица обладает всеми признаками конвейерно-векторной архитектуры, то $FKV = true$, переход на шаг 10.

Шаг 7. Определение соответствия матрицы всем признакам матрицы для векторно-конвейерной архитектуры.

Шаг 8. Если матрица обладает всеми признаками векторно-конвейерной архитектуры, то $FVK = true$, переход на шаг 10.

Шаг 9. Установка флага $FN = true$.

Шаг 10. Конец алгоритма.

Рассмотрим матрицы и алгоритмы для определения каждого вида структур:

1. Конвейерная структура

Отличием матрицы для данной структуры является наличие значений ‘1’ по диагонали над главной диагональю матрицы. В остальных ячейках должны быть значения ‘0’. Пример матрицы конвейерной структуры для пяти ПМ представлен на рисунке 2.:

$$M = \begin{bmatrix} X & 1 & 0 & 0 & 0 \\ X & X & 1 & 0 & 0 \\ X & X & X & 1 & 0 \\ X & X & X & X & 1 \\ X & X & X & X & X \end{bmatrix}$$

Рисунок 2 – Матрица связи конвейерной структуры

Рассмотрим *алгоритм определения конвейерной структуры*. На вход алгоритма поступает матрица связей M. На выходе имеем флаг $FKonv = \{true, false\}$ конвейерной структуры.

Шаг 1. Начальная инициализация $FKonv = true$.

Шаг 2. Если $i \ge q$, то переход на шаг 8.

Шаг 3. Если $j \ge q$, то $i = i+1$, переход на шаг 2.

Шаг 4. Если $i \le j$, то $j = j+1$, переход на шаг 3.

Шаг 5. Если $i = j-1$, то переход на шаг 7.

Шаг 6. Если $M_{ij} \neq 0$, то $FKonv = false$, $j = j + 1$, переход на шаг 3.

Шаг 7. Если $M_{ij} \neq 1$, то $FKonv = false$, $j = j + 1$, переход на шаг 3.

Шаг 8. Конец алгоритма.

2. Векторная структура

Отличием матрицы для данной структуры является то, что все значения ячеек равны ‘0’. Пример матрицы для пяти ПМ представлен на рисунке 3.

$$M = \begin{bmatrix} X & 0 & 0 & 0 & 0 \\ X & X & 0 & 0 & 0 \\ X & X & X & 0 & 0 \\ X & X & X & X & 0 \\ X & X & X & X & X \end{bmatrix}$$

Рисунок 3 – Матрица связи векторной структуры

Рассмотрим *алгоритм определения векторной структуры*. На вход алгоритма поступает матрица связей M. На выходе имеем флаг $FVect = \{true, false\}$ векторной структуры.

Шаг 1. Начальная инициализация $FVect = true$.

Шаг 2. Если $i \geq q$, то переход на шаг 6.

Шаг 3. Если $j \geq q$, то $i = i + 1$, переход на шаг 2.

Шаг 4. Если $i \leq j$, то $j = j + 1$, переход на шаг 3.

Шаг 5. Если $M_{ij} = 1$, то $FVect = false$, $j = j + 1$, переход на шаг 3. В противном случае $j = j + 1$, переход на шаг 3**.**

Шаг 6. Конец алгоритма.

3. Векторно-конвейерная структура

Отличием матрицы структуры является то, что все значения ячеек равны ‘0’, кроме некоторых (не всех) значений ‘1’ над главной диагональю матрицы. Пример матрицы для пяти ПМ представлен на рисунке 4.

$$M = \begin{bmatrix} X & 1 & 0 & 0 & 0 \\ X & X & 0 & 0 & 0 \\ X & X & X & 1 & 0 \\ X & X & X & X & 0 \\ X & X & X & X & X \end{bmatrix}$$

Рисунок 4 – Матрица связи векторно-конвейерной структуры

Рассмотрим *алгоритм определения векторно-конвейерной структуры*. На вход алгоритма поступает матрица связей M. На выходе имеем флаг $FVK = \{true, false\}$ векторно-конвейерной структуры.

Шаг 1. Начальная инициализация $FVK = true$.

Шаг 2. Если $i \geq q$, то переход на шаг 8.

Шаг 3. Если $j \geq q$, то $i = i + 1$, переход на шаг 2.

Шаг 4. Если $i \leq j$, то $j = j + 1$, переход на шаг 3.

Шаг 5. Если $i \neq j - 1$, то переход на шаг 7.

Шаг 6. Если $M_{ij} = 1$, то $FVK = false$, $j = j + 1$, переход на шаг 3.

Шаг 7. $j = j + 1$, переход на шаг 3.

Шаг 8. Конец алгоритма.

4. Конвейерно-векторная структура

Отличием матрицы для данной структуры является то, что все значения ‘1’ сгруппированы в прямоугольные контуры, которые упорядочены в матрицы по типу лестницы и нет рядов и строк матрицы без хотя бы одного значения ‘1’. Пример матрицы векторной структуры для пяти ПМ представлен на рисунке 5.

$$M = \begin{bmatrix} X & 1 & 1 & 1 & 0 \\ X & X & 0 & 0 & 1 \\ X & X & X & 0 & 1 \\ X & X & X & X & 1 \\ X & X & X & X & X \end{bmatrix}$$

Рисунок 5 – Матрица связи конвейерно-векторной структуры

Рассмотрим *алгоритм определения конвейерно-векторной структуры*. На вход алгоритма поступает матрица связей M. На выходе имеем флаг $FKV = \{true, false\}$ конвейерно-векторной структуры.

Шаг 1. Начальная инициализация $FKV = true$.

Шаг 2. Если $i \geq q$, то переход на шаг 8.

Шаг 3. Если $j \geq q$, то $i = i + 1$ переход на шаг 2.

Шаг 4. Проверка условия: элемент входит в строго прямоугольный контур, состоящий из значений, равных ‘1’.

Шаг 5. Проверка условия: все элементы справа и сверху прямоугольного контура равны ‘0’.

Шаг 6. Проверка условия: все элементы слева и снизу граничного элемента прямоугольного контура имеют значение ‘X’.

Шаг 7. Проверка условия: все прямоугольные контуры упорядочены в виде “лест-

ницы”, “ступени” которой не прерываются по вертикали и горизонтали, $j = j + 1$, переход на шаг 3.

Шаг 8. Конец алгоритма.

Процедура “Проверка условия: элемент входит в прямоугольную область” реализуется методом обхода границ области (рисунок 6):

$$M = \begin{bmatrix} X & 0 & 1 & 1 & 0 \\ X & X & 1 & 1 & 0 \\ X & X & X & 0 & 0 \\ X & X & X & X & 0 \\ X & X & X & X & X \end{bmatrix}$$

Рисунок 6 – Метод обхода границ для матрицы связи

1. Маркер двигается вправо до ‘0’, пройденный путь обозначим как x_1.

2. Маркер двигается вниз до ‘0’, пройденный путь обозначим как y_1.

3. Маркер двигается влево до ‘0’, пройденный путь обозначим как x_2.

4. Маркер двигается вниз до ‘0’, пройденный путь обозначим как y_2.

Если $x_1 = x_2$ и $y_1 = y_2$, то контур прямоугольный.

Практические исследования

Программные средства описания многопроцессорной вычислительной структуры на базе нейропроцессоров были реализованы в программном комплексе «НейроКС» [4,5].

Алгоритм определения связей элементов вычислительной структуры на базе нейропроцессоров с целью определения вида НПС реализован следующим образом: после вставки директив в текстовом редакторе становится возможным режим «Генерация матрицы связи подпрограмм» для визуального представления матрицы M (рисунок 7).

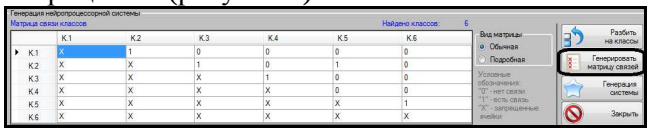

Рисунок 7 – Визуальное представление матрицы M

После генерации матрицы M возможна генерация матрицы M' (рисунок 8).

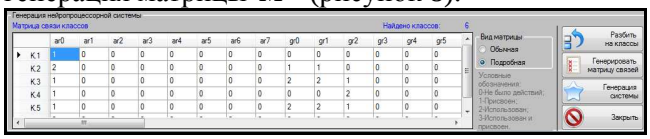

Рисунок 8 – Визуальное представление матрицы M'

Алгоритм определения вида структуры НПС на основе описания связей ее элементов с целью использования оценок эффективности для этого вида структуры реализован следующим образом: после вызова соответствующей процедуры на основании матрицы связей M' происходит запуск подсистемы «Конструктор систем» для визуального представления вычислительной структуры (рисунок 9).

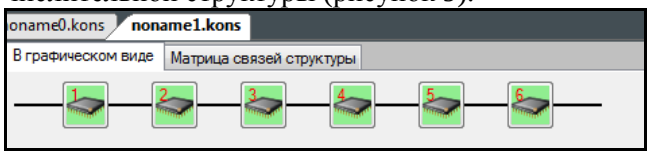

Рисунок 9 – Визуальное представление интеллектуальной вычислительной системы

Было произведено экспериментальное исследование разработанного программного комплекса на примере реализации алгоритма шифрования по методу ГОСТ 28147-89. Реализованы НПС конвейерной структуры с различным числом ПМ, что позволило в дальнейшем сравнить время выигрыша относительно однопроцессорного варианта (в данной статье не рассматривается) и получить следующие результаты: 2 ПМ – выигрыш на 50.31%, 3 ПМ - на 98.07%, 4 ПМ - на 152.59%, 5 ПМ - на 195.53%, 6 ПМ - на 244.63%, 7 ПМ - на 291.48%, 8 ПМ - на 393.62%.

Результаты исследования внедрены в НТЦ «Модуль», который является производителем нейропроцессоров семейства NM 640X.

Заключение

Таким образом, для проектирования и анализа вычислительных структур получены следующие основные результаты:

- алгоритм определения связей элементов вычислительной структуры на базе нейропроцессоров с целью определения вида НПС;
- алгоритм определения вида структуры НПС на основе описания связей ее элементов;
- программные средства анализа интеллектуальных вычислительных структур на базе нейропроцессоров семейства NM 640X.

Библиографический список

1. *Галушкин А.И.*, Нейрокомпьютеры. Кн.3. – М: ИПРЖР, 2000. - 528 с.
2. *Злобин В.К., Ручкин В.Н.* Нейросети и нейрокомпьютеры: Учеб. пособие. / В.К., Злобин. В.Н. Ручкин. - СПб.: БХВ-Петербург, 2011. - 256 с..
3. *Головкин Б.А.* Вычислительные системы с большим числом процессоров. М.: Радио и связь, 1995. - 320 с.
4. *Калинкина Т.И., Костров Б.В., Ручкин В.Н.* Телекоммуникационные и вычислительные сети. Архитектура, стандарты и технологии. – СПб.: БХВ-Петербург, 2010. – 288 с.
5. *Романчук В.А., Ручкин В.Н.* Разработка программных средств анализа нейропроцессорных систем // Вестник РГРТУ. 2010. №2. Вып.32. С.61-67.