ИНФОРМАТИКА

2005 **апрель-июнь** **№ 2**

УДК 681.324

А.А. Баркалов, А.Ф. Буковец

СИНТЕЗ МИКРОПРОГРАММНОГО АВТОМАТА С МНОЖЕСТВЕННЫМ КОДИРОВАНИЕМ НАБОРОВ МИКРООПЕРАЦИЙ

Предлагается метод оптимизации аппаратурных затрат в логической схеме микропрограммного автомата Мили, основанный на присвоении различных кодов одинаковым наборам микроопераций из разных подтаблиц прямой структурной таблицы автомата. Дан пример применения предложенного метода.

Введение

Устройство управления, которое является одним из важнейших блоков любой цифровой системы и во многом определяет ее характеристики [1], может быть реализовано как микропрограммный автомат (МПА) Мили [2]. Для синтеза логических схем современных микропрограммных автоматов широко используются программируемые логические устройства (ПЛУ), такие как CPLD (complex programmable logic devices) или FPGA (field-programmable gate arras) [3, 4]. Подобные устройства отличаются высокой стоимостью, что делает актуальной задачу уменьшения аппаратурных затрат (числа БИС) в схеме микропрограммного автомата. Один из способов решения этой задачи – представление схемы МПА в виде многоуровневой структуры [5], для чего могут использоваться такие методы, как кодирование логических условий, максимальное кодирование наборов микроопераций, кодирование полей совместимых микроопераций, кодирование строк прямой структурной таблицы, преобразование кодов объектов [5, 6]. Каждый из них позволяет уменьшить аппаратурные затраты при выполнении определенных условий. В настоящей работе предлагается один из методов решения задачи уменьшения аппаратурных затрат при синтезе логической схемы МПА Мили на ПЛУ. В основе метода лежит присвоение разных кодов одинаковым наборам микроопераций, что приводит к уменьшению числа функций, формируемых на основе прямой структурной таблицы автомата.

1. Основная идея предлагаемого метода

Логическая схема микропрограммного автомата Мили задается системой булевых функций [2]

$$Y = Y(T, X);$$
$$\Phi = \Phi(T, X). \qquad (1)$$

Здесь $X = \{x_1, \ldots, x_L\}$ – множество логических условий, записанных в условных вершинах исходной граф-схемы алгоритма (ГСА); $Y = \{y_1, \ldots, y_N\}$ – множество выходных сигналов (микроопераций) автомата, записанных в операторных вершинах исходной ГСА; $T = \{T_1, \ldots, T_{R1}\}$ – множество внутренних переменных, кодирующих внутренние состояния МПА $a_m \in A = \{a_1, \ldots, a_{M1}\}$, где $R_1 =]\log_2 M_1[$; $\Phi = \{\phi_1, \ldots, \phi_{R1}\}$ – множество функций возбуждения памяти МПА. Система функций (1) формируется по прямой структурной таблице (ПСТ) МПА (табл. 1), имеющей столбцы a_m, $K(a_m)$, a_s, $K(a_s)$, X_h, Y_h, Φ_h, h. Здесь $a_m \in A$ – исходное состояние МПА; $K(a_m)$ – код состояния a_m; $a_s \in A$ – состояние перехода; $K(a_s)$ – код состояния прехода a_s; X_h – конъюнкция некоторых элементов множества логических условий X, определяющая переход $\langle a_m, a_s \rangle$; $Y_h \subseteq Y$ – множество микроопераций (микрокоманд), формируемых при переходе $\langle a_m, a_s \rangle$; $\Phi_h \subseteq \Phi$ – множество функций возбуждения памяти МПА, принимающих единичное значение для переключения памяти из состояния с кодом $K(a_m)$ в состояние с кодом $K(a_s)$; h – номер перехода и одновременно номер строки ПСТ, $h = 1, \ldots, H$.

Исходная ПСТ является основой для формирования системы (1), которая представляется в следующем виде:

$$y_n = \bigvee_{h=1}^{H} C_{nh} A_m^h X_h, \ (n = 1, \ldots, N); \tag{2}$$

$$\phi_r = \bigvee_{h=1}^{H} C_{rh} A_m^h X_h, \ (r = 1, \ldots, R_1). \tag{3}$$

Здесь C_{nh} (C_{rh}) – булева переменная, равная единице, если и только если функция y_n (ϕ_r) записана в h-й строке исходной ПСТ; A_m^h – конъюнкция внутренних переменных, соответствующая коду $K(a_m)$ исходного состояния $a_m \in A$ из строки h, $h = 1, \ldots, H$.

Таблица 1

Прямая структурная таблица МПА Мили S_1

a_m	$K(a_m)$	a_s	$K(a_s)$	X_h	Y_h	Φ_h	h
a_1	000	a_2	001	x_1	y_1y_2	D_3	1
		a_3	010	$\neg x_1 x_2$	y_3	D_2	2
		a_2	001	$\neg x_1 \neg x_2$	y_2y_4	D_3	3
a_2	001	a_3	010	x_2	y_3y_4	D_2	4
		a_4	011	$\neg x_2$	y_5	D_2D_3	5
a_3	010	a_5	100	x_3	y_1y_2	D_1	6
		a_4	011	$\neg x_3$	y_3y_5	D_2D_3	7
a_4	011	a_1	000	x_1	y_2	–	8
		a_5	100	$\neg x_1$	y_1y_2	D_1	9
a_5	100	a_1	000	1	y_3y_4	–	10

Пусть ПСТ содержит M_2 попарно различных наборов микроопераций $Y_q \subseteq Y$. Поставим в соответствие каждой микрокоманде $Y_q \subseteq Y$ двоичный код $K(Y_q)$, имеющий $R_2 =]\log_2 M_2[$ разрядов ($q = 1, \ldots, M_2$). Используем для кодирования микрокоманд переменные $z_r \in Z = \{ z_1, \ldots, z_{R2}\}$. В этом случае МПА Мили может быть реализован в виде двухуровневой схемы (рис. 1), называемой в дальнейшем PY-автоматом Мили [5, 6].

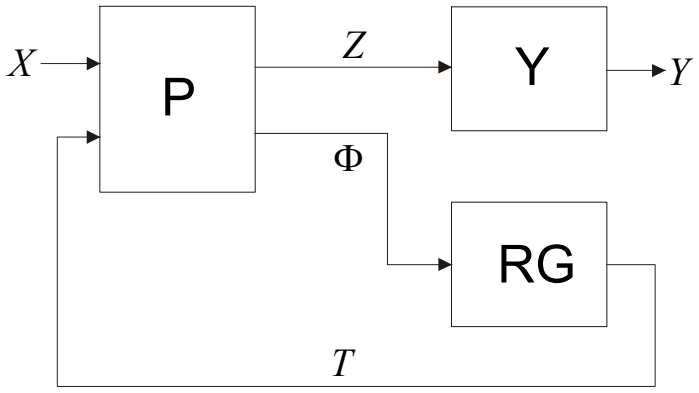

Рис. 1. Структурная схема PY-автомата Мили

Регистр RG реализует память автомата и построен на триггерах типа D, что соответствует практической реализации подобных схем [7], схема Y преобразовывает переменные $z_r \in Z$ в микрооперации автомата Мили $y_n \in Y$. Схема Y реализует систему функций

$$Y = Y(Z). \tag{4}$$

Такой подход позволяет уменьшить число выходов схемы P по сравнению с одноуровневой реализацией автомата. Однако число выходов схемы P по-прежнему остается значительным и определяется как $t(\mathrm{PY})$:

$$t(\mathrm{PY}) = R_1 + R_2 . \tag{5}$$

Предлагаемый в данной работе подход ориентирован на уменьшение этого показателя, что приводит к уменьшению аппаратурных затрат в схеме автомата в целом.

Назовем часть ПСТ, задающую переходы из состояния $a_m \in A$, подтаблицей ПСТ. Пусть M_3 – максимальное число микрокоманд, встречающихся в различных подтаблицах исходной ПСТ. Закодируем микрокоманды $Y_q \subseteq Y$ двоичными кодами $K_m(Y_q)$, имеющими $R_3 =]\log_2 M_3[$ разрядов, и используем для такого кодирования переменные из множества $\Psi = \{\psi_1, \ldots, \psi_{R3}\}$. В этом случае МПА Мили может быть представлен в виде структуры, называемой в дальнейшем PY_0-автоматом (рис. 2).

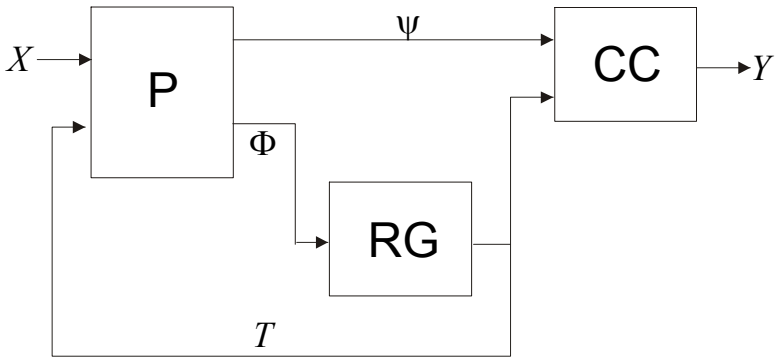

Рис. 2. Структурная схема PY_0-автомата Мили

В данной структуре преобразователь кодов CC формирует функции Y, представленные в виде

$$Y = Y(T, \Psi) , \tag{6}$$

а схема P реализует систему функций (2) и систему

$$\Psi = \Psi(T, X) . \tag{7}$$

Из рис. 2 видно, что схема P требует теперь только $t(\mathrm{PY}_0)$ выходов, где

$$t(\mathrm{PY}_0) = R_1 + R_3 . \tag{8}$$

Если выполняется условие

$$R_3 < R_2 , \tag{9}$$

то предлагаемый в работе подход позволяет уменьшить аппаратурные затраты в логической схеме МПА по сравнению с PY-автоматом Мили.

2. Метод синтеза PY_0-автомата Мили

В настоящей работе предлагается метод синтеза PY_0-автомата Мили, иллюстрируемый на примере МПА Мили S_1 (см. табл. 1). Предлагаемый метод включает следующие шаги.

1. *Множественное кодирование наборов микроопераций.* Пусть m-я подтаблица ПСТ МПА Мили содержит H_m различных наборов микроопераций, тогда максимальное число микрокоманд в подтаблице определяется как $M_3 = \max(H_1, \ldots, H_M)$. Закодируем наборы $Y_q \subseteq Y$ из разных подтаблиц произвольными кодами $K_m(Y_q)$, где индекс m соответствует индексу исходного состояния в подтаблице ПСТ.

Автомат Мили S_1 содержит $M_2 = 7$ попарно различных наборов микроопераций: $Y_1 = \{y_1, y_2\}$, $Y_2 = \{y_3\}$, $Y_3 = \{y_2, y_4\}$, $Y_4 = \{y_3, y_4\}$, $Y_5 = \{y_5\}$, $Y_6 = \{y_3, y_5\}$, $Y_7 = \{y_2\}$. Поэтому $R_2 = 3$, $t(\text{PY}) = 6$. Из табл. 1 видно, что $M_3 = 3$ и, следовательно, $R_3 = 2$, $\Psi = \{\psi_1, \psi_2\}$, $t(\text{PY}_0) = 5$. В этом случае условие (9) выполняется, поэтому применение модели PY_0-автомата может быть целесообразно.

Сформируем таблицу множественного кодирования наборов микроопераций (табл. 2), содержащую столбцы $Y_h(a_1)$, $K_1(Y_t)$, …, $Y_h(a_M)$, $K_M(Y_t)$. Четные столбцы этой таблицы содержат коды $K_m(Y_q)$ наборов микроопераций $Y_q \subseteq Y$ для различных подтаблиц исходной ПСТ.

Таблица 2

Таблица множественного кодирования наборов микроопераций МПА S_1

$Y_q(a_1)$	$K_1(Y_q)$	$Y_q(a_2)$	$K_2(Y_q)$	$Y_q(a_3)$	$K_3(Y_q)$	$Y_q(a_4)$	$K_4(Y_q)$	$Y_q(a_5)$	$K_5(Y_q)$
Y_1	00	Y_4	00	Y_1	00	Y_1	00	Y_4	00
Y_2	01	Y_5	01	Y_6	01	Y_7	01	–	–
Y_3	10	–	–	–	–	–	–	–	–
–	–	–	–	–	–	–	–	–	–

2. *Формирование преобразованной ПСТ.* Преобразованная ПСТ является основой для формирования систем функций (2) и (7). Таблица формируется путем замены столбца Y_h исходной ПСТ столбцом Ψ_h, который содержит переменные $\psi_R \in \Psi$, принимающие единичное значение в коде $K_m(Y_q)$ (табл. 3).

Таблица 3

Фрагмент преобразованной ПСТ PY_0-автомата S_1

a_m	$K(a_m)$	a_s	$K(a_s)$	X_h	Ψ_h	Φ_h	h
a_1	000	a_2	001	x_1	–	D_3	1
		a_3	010	$\neg x_1 x_2$	ψ_2	D_2	2
		a_2	001	$\neg x_1 \neg x_2$	ψ_1	D_3	3
a_2	001	a_3	010	x_2	–	D_2	4
		a_4	011	$\neg x_2$	ψ_2	$D_2 D_3$	5
a_3	010	a_5	100	x_3	–	D_1	6
		a_4	011	$\neg x_3$	ψ_2	$D_2 D_3$	7
a_4	011	a_1	000	x_1	ψ_2	–	8
		a_5	100	$\neg x_1$	–	D_1	9
a_5	100	a_1	000	1	–	–	10

3. *Формирование таблицы преобразователя кодов.* Таблица преобразователя кодов СС является основой для формирования системы функций (6), задающей закон формирования выходных сигналов автомата Мили. Предлагаемая в настоящей работе таблица СС включает столбцы $K(a_m)$, $K_m(Y_q)$, Y_q, h, при этом столбец Y_q содержит микрооперации $y_n \in Y_q$. Система (6) имеет регулярный характер, т. е. определена на более чем 50 % всевозможных термов. Как известно [6], для реализации подобных функций целесообразно использовать постоянные запоминающие устройства (ПЗУ). Для реализации системы (6) на ПЗУ таблица СС должна содержать H_0 строк, где

$$H_0 = 2^{R_1 + R_3} \geq M_3 . \tag{10}$$

Таблица преобразователя кодов СС PY_0-автомата S_1 содержит 32 строки ($H_0 = 32$), но в табл. 4 показаны только первые строки этой таблицы.

Таблица 4

Фрагмент таблицы преобразователя кодов PY_0-автомата Мили S_1

$K(a_m)$	$K_m(Y_q)$	Y_q	h
000	00	y_1y_2	1
000	01	y_3	2
000	10	y_2y_4	3
000	11	–	4
001	00	y_3y_4	5
001	01	y_5	6
001	10	–	7
001	11	–	8

4. *Реализация логической схемы PY_0-автомата Мили.* Схема P реализуется на ПЛУ, используя системы функций (3) и (7). Так, из табл. 3 может быть получен фрагмент дизъюнктивных нормальных форм искомых функций:

$$\psi_1 = \overline{T_1}\,\overline{T_2}\,\overline{T_3}\,\overline{x_1}\,\overline{x_2};$$

$$D_1 = \overline{T_1}T_2\,\overline{T_3}x_3 \vee \overline{T_1}T_2T_3\,\overline{x_1}.$$

Схема преобразователя кодов CC реализуется на ПЗУ по соответствующей таблице. Вопросы реализации подобных схем достаточно полно рассмотрены в работе [7] и выходят за рамки данной статьи.

3. Исследование эффективности предложенного метода

Используя результаты работы [8], можно получить зависимость, показывающую относительную стоимость предложенной структуры по отношению к PY-автомату Мили. Эта зависимость имеет вид

$$f(p_1,k_1,k_2,k_3,k_4,K,N) = \frac{S(\mathrm{PY}_0)}{S(\mathrm{PY})}, \tag{11}$$

где $S(\mathrm{PY}_0)$ – общая стоимость PY_0-автомата Мили и $S(\mathrm{PY})$ – общая стоимость PY-автомата Мили; стоимость оценивается как площадь реализации схемы на заказных матрицах [2]. Если $f < 1$, то аппаратурные затраты в логической схеме PY_0-автомата, реализуемой на стандартных ПЛУ, также будут меньше стоимости логической схемы эквивалентного PY-автомата Мили. В исследованиях использовался вероятностный подход, предложенный Г.И. Новиковым [1], при котором рассматриваются общие тенденции сложности реализации за счет выделения классов ГСА. Анализ результатов применения предложенных методов к отдельным ГСА дает оценки только в некоторых точках пространства возможных решений, что не позволяет увидеть общих тенденций. Также в исследованиях использовались результаты работы [8], согласно которым относительные сложности реализации логических схем МПА на заказных матрицах и стандартных схемах совпадают с точностью до 85–90%.

С помощью методики [2] можно получить матричную реализацию PY-автомата Мили (рис. 3).

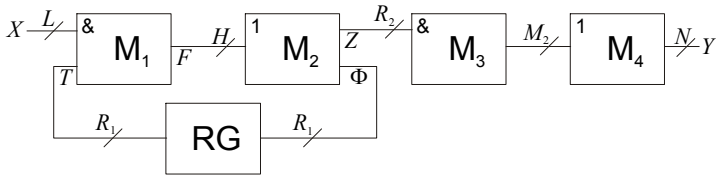

Рис. 3. Матричная реализация PY-автомата Мили

В этой структуре конъюнктивная матрица M_1 и дизъюнктивная матрица M_2 представляют собой двухуровневую ПЛМ-реализацию схемы P, в то время как конъюнктивная матрица M_3 и дизъюнктивная матрица M_4 реализуют схему Y, которая в силу своей регулярности реализуется на ПЗУ. Сложность матричной реализации S(PY) PY-автомата оценивается в условных единицах площади:

$$S(\mathrm{PY}) = S(\mathrm{P}) + S(\mathrm{Y}) = 2(L + R_1)H + H(R_2 + R_1) + k_3 2R_2 M_2 + M_2 N, \tag{12}$$

где S(P) – площадь матричной реализации схемы P; S(Y) – сложность матричной реализации схемы Y; коэффициент k_3 задает отношение между ценой ПЛМ или ПМЛ, с одной стороны, и ПЗУ – с другой; K – число вершин в исходной ГСА. Используя результаты работы [8], параметры L, H, M_2 могут быть представлены как функции от числа вершин K и некоторых коэффициентов:

$$L = \frac{(1 - p_1)}{1{,}4} K\ ; \tag{13}$$

$$H = 10{,}6 + 1{,}44 k_1 p_1 K\ ; \tag{14}$$

$$M_2 = k_2 p_1 K\ . \tag{15}$$

Здесь p_1 – доля операторных вершин по отношению к общему числу вершин ГСА, k_1 – отношение числа состояний автомата к числу операторных вершин ГСА, k_2 – отношение числа различных наборов микроопераций к числу операторных вершин ГСА. Именно параметр p_1 определяет класс ГСА с одинаковой долей операторных вершин среди множества различных. Этот параметр рассматривается как вероятность того, что произвольно взятая вершина ГСА будет операторной вершиной, при этом параметр $1 - p_1$ рассматривается как вероятность появления условных вершин в данной ГСА. Матричная реализация PY_0-автомата Мили строится по той же методике, что и для PY-автомата (рис. 4).

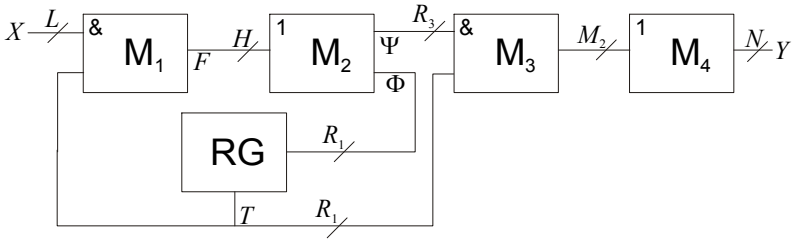

Рис. 4. Матричная реализация PY_0-автомата Мили

Матрицы M_1 и M_2 реализуют схему P, а матрицы M_3 и M_4 – преобразователь кодов СС. Сложность реализации этой структуры оценивается как

$$S(\mathrm{PY}_0) = S(\mathrm{P}) + S(\mathrm{Y}_0) = 2(L + R_1)H + H(R_2 + R_1) + k_3 2(R_3 + R_1)M_3 + M_3 N, \tag{16}$$

где $R_3 = k_4 R_2$, $M_3 = k_4 M_2$ и k_4 – отношение мощности множества Ψ к мощности множества Z.

Исследование эффективности предложенного метода выполнено с использованием системы Matlab. На вид зависимости (11) больше всего влияют коэффициенты k_1, k_2, k_4, p_1 и число микроопераций в исходной ГСА N. Коэффициент $k_3 = 0{,}1$ был выбран на основе прайс-листов, позволяющих найти относительную стоимость ПЛМ и ПЗУ с одинаковым числом входов. Исследуемая функция f анализировалась при изменении числа вершин ГСА K в интервале от 50 до 1000, при этом все параметры фиксировались, а один менялся. Наилучшие результаты в таком случае достигались при числе вершин K порядка 500, если число микроопераций N превышало 200 (рис. 5, *а*).

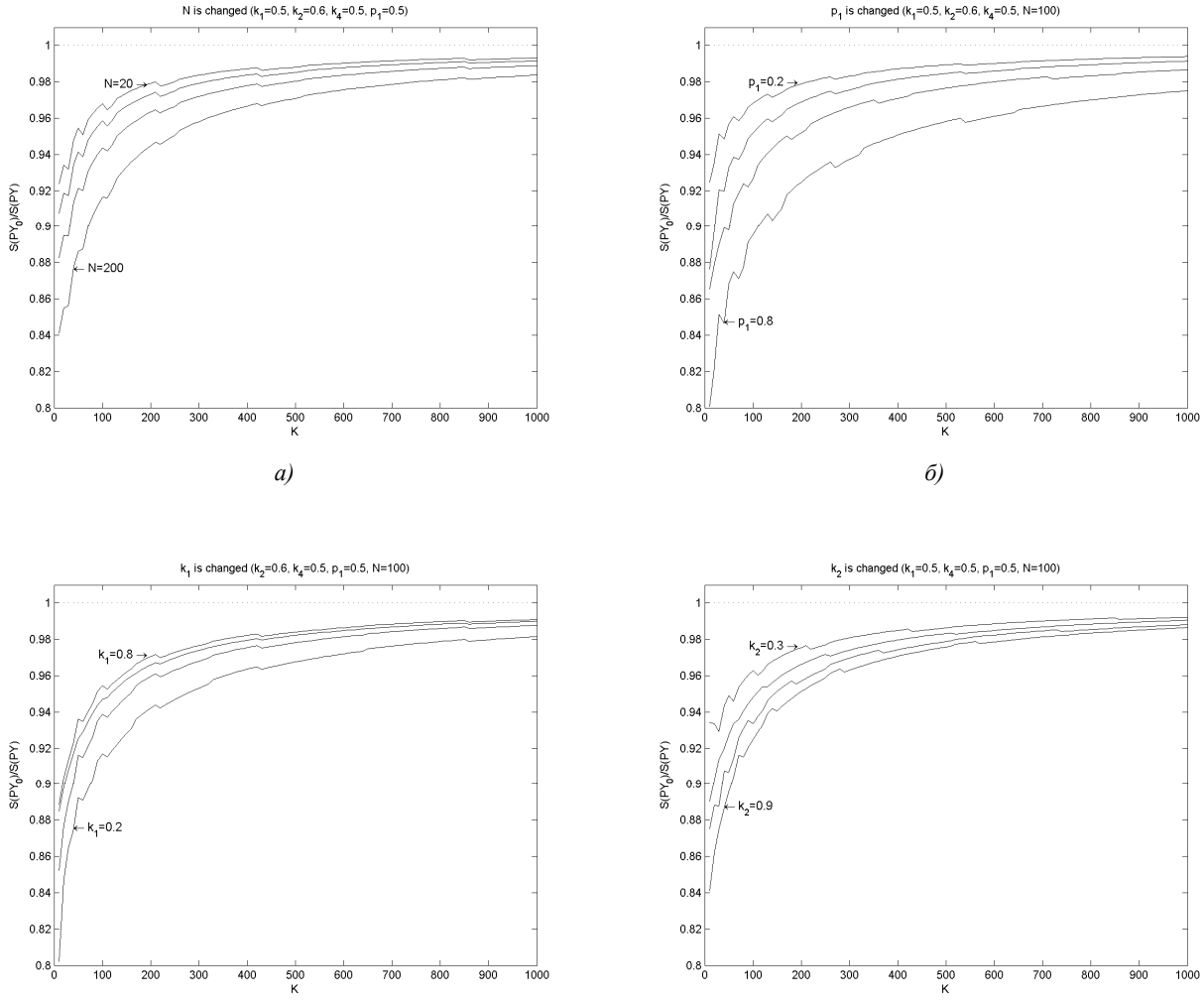

Рис. 5. Графики зависимости $f(K)$ при изменении параметров N, p_1, k_1 и k_2

Выигрыш возрастает по мере роста числа операторных вершин (рост p_1 на рис. 5, *б*). Лучшие результаты достигаются при низких значениях коэффициента k_1 (рис. 5, *в*), что соответствует ситуации, когда число операторных вершин значительно превосходит число состояний, т. е. соответствует разветвленным алгоритмам. Выигрыш также повышается при росте отношения числа микрокоманд (наборов микроопераций в операторных вершинах) к числу операторных вершин, т. е. по мере роста параметра k_2 (рис. 5, *г*).

Заключение

Предложенный в работе метод позволяет уменьшить аппаратурные затраты при двухуровневой реализации логической схемы микропрограммного автомата Мили на программируемых логических устройствах, основанной на максимальном кодировании наборов микроопераций. Проведенные исследования показали, что описанный метод всегда дает лучшие результаты, чем ранее известный. При этом в некоторых случаях, определяемых сочетанием параметров исходной граф-схемы алгоритма, выигрыш может достигать 15 %. Наилучшие результаты получаются при синтезе МПА с низким отношением числа состояний к числу операторных вершин и с высоким отношением числа различных наборов микроопераций к общему числу операторных вершин, т. е. при интерпретации разветвленных ГСА. Выигрыш увеличивается по мере убывания числа вершин исходной ГСА.

Проведенные исследования финансировались Комитетом научных исследований (Польша) – грант № 3 T11C 046 26.

Список литературы

1. Майоров С.А., Новиков Г.И. Структура электронных вычислительных машин. – Л.: Машиностроение, 1979. – 384 с.

2. Baranov S. Logic Synthesis for Control Automata. – Kluwer Academic Publishers, 1994. – 312 p.

3. Jenkins J. Designing with FPGAs and CPLDs. – PTR Prentice Hall, 1994. – 274 p.

4. Грушницкий Р.И., Мурсаев А.Х., Угрюмов Е.П. Проектирование систем на микросхемах программируемой логики. – СПб.: БХВ, 2002. – 636 с.

5. Скляров В.А. Синтез автоматов на матричных БИС. – Мн.: Наука и техника, 1984. – 256 с.

6. Баркалов А.А. Синтез устройств управления на программируемых логических устройствах. – Донецк: ДонНТУ, 2002. – 262 с.

7. Соловьев В.В. Проектирование цифровых систем на основе программируемых логических интегральных схем. – М.: Горячая линия–Телеком, 2001. – 636 с.

8. Баркалов А.А. Разработка фомализованных методов структурного синтеза композиционных автоматов. Автореф. дис. ... д-ра техн. наук. – Донецк: ДГТУ, 1995. – 28 с.

Поступила 16.11.04

Зеленогурский университет,
Польша, Зеленая Гура, Подгорная, 50
e-mail: a.barkalov@iie.uz.zgora.pl
a.bukowiec@iie.uz.zgora.pl

A.A. Barkalov, A.F. Bukowiec

THE SYNTHESIS OF MICROPROGRAM AUTOMAT WITH MULTYSETS OF MICROOPERATIONS

The method of hardware optimization in the logic circuit of Mealy FSM is proposed. Method is based on the matching of different codes with the equal sets of microoperations in the different subtables of direct structural table. An example of application of proposed method is given.