УДК 004.3

Модификация алгоритма кодирования полей совместимых микроопераций

Баркалов А.А.[1], Ковалёв С.А.[2], Зеленёва И.Я.[2], Мирошкин А.Н.[2]
[1] Зеленогурский университет, г. Зеленая гора, Польша;
[2] Донецкий национальный технический университет, Украина.
A.Barkalov@iie.uz.zgora.pl, irina@cs.dgtu.donetsk.ua, MiroshkinAN@gmail.com

Abstract

Barkalov A.A., Kovalyov S.A., Zelenyova I.Y., Miroshkin A.N. Modification of algorithm of encoding of sets of compatible microinstructions. *Modification of algorithm of encoding of sets of compatible microinstructions is proposed. Modification is based on control microinstructions copies using. Such method ensures more flexible forming of sets of compatible microinstructions. Possibility conditions of method modification usage are defined; example of method application is given.*

Введение

При проектировании цифровых управляющих устройств (УУ) одной из актуальных задач является минимизация аппаратурных затрат, необходимых для его реализации [1]. В настоящее время базис программируемых логических интегральных схем (ПЛИС) [2, 3] широко используется для реализации схем устройств управления [4, 5]. Современные семейства микросхем FPGA содержат настраиваемые блоки встроенной памяти (БВП), что дает возможность реализовать КМУУ на одной микросхеме. Поскольку объем встроенной памяти относительно невелик, актуальной становится проблема сокращения разрядности слова микропрограммы. В настоящей работе предлагается модификация метода кодирования полей совместимых микроопераций, что при выполнении определенных условий приводит к уменьшению разрядности управляющего слова.

Целью исследования является уменьшение аппаратурных затрат в схеме композиционного микропрограммного устройства управления за счёт модификации алгоритма кодирования полей совместимых микроопераций. Задачей исследования является разработка метода синтеза, позволяющего уменьшить аппаратурные затраты, необходимые для реализации устройства управления.

Основные теоретические положения

С момента предложения М. Уилксом в 1951 году принципа микропрограммного управления [6] появилась проблема оптимального кодирования операционной части микрокоманд. Широкое применение получили следующие стратегии кодирования: максимальное, унитарное, сегментация микрокоманд, кодирование полей совместимых микроопераций [7].

При максимальном кодировании операционная часть представляется в виде кода разрядности

$$m = \lceil \log_2 T \rceil, \qquad (1)$$

где T - число наборов микроопераций (МО). При помощи дешифратора такой код преобразуется в унитарный код $Y_1,...,Y_T$, из которого по логике «ИЛИ» формируются сигналы микроопераций.

Недостатком данного подхода является значительная временная задержка, вносимая дешифратором и схемой «ИЛИ» при большом количестве МО.

При унитарном кодировании каждой МО соответствует один разряд в операционной части микрокоманды (МК). Преимуществом схемы является максимальная гибкость микропрограммирования и максимальное быстродействие. Однако разрядность МК линейно растет при увеличении количества МО.

При рассмотрении методов сегментации микрокоманд необходимо отличать классический метод сегментации [8] от предложенного в работе [7]: классический метод предполагает разбиение общего адреса микрокоманды на адрес сегмента и адрес микрокоманды в ней. В предлагаемом методе [7] разбивается не адрес, а операционная часть микропрограммы. При этом под сегментом понимается совокупность микрокоманд с совпадающими m_1 старшими разрядами кода $K(Y_t)$. В этом случае, как и при классической

сегментации, возникает задача оптимального распределения микрокоманд по сегментам.

Преимуществом метода сегментации МК в устройствах управления является постоянство временных характеристик выполнения микропрограммы, то есть, быстродействие операционного устройства не снижается. Недостатком метода является высокая трудоемкость алгоритмизации и программирования в рамках автоматизированной системы проектирования задачи разбиения.

Компромиссным решением является кодирование полей совместимых микроопераций, при котором множество микроопераций делится на подмножества совместимых МО. Совместимыми называются такие МО, которые никогда не встречаются вместе в микрокомандах. В каждом подмножестве выполняется максимальное кодирование микроопераций. Полученный код размещается в соответствующем поле операционной части. Для получения сигналов МО каждому полю ставится в соответствие дешифратор.

Классический метод кодирования полей совместимых микроопераций предполагает разбиение множества микроопераций $Y = \{y_1,..., y_N\}$ на подмножества (классы) $G_1,..., G_I$. Каждой группе G_i ставится в соответствие поле в МК разрядности

$$B_i = \lceil \log_2(|G_i| + C_i) \rceil, \qquad (2)$$

где $|G_i|$ – количество МО в классе G_i; C_i - переменная, принимающая единичное значение в том случае, если существует такой набор Y_k, для которого $Y_k \cap G_i = \varnothing$.

Разрядность операционной части МК определяется как

$$B = \sum_{i=1}^{I} B_i . \qquad (3)$$

Предложенный С. Шварцем алгоритм разбиения микроопераций на подмножества [7] и дальнейшие модификации этого алгоритма [10] базируются на следующих ограничениях:

1. Каждая МО $y_n \in Y$ содержится только в одном классе G_i $(i = 1,..., I)$.
2. Микрооперации y_n, y_m, принадлежащие одному набору Y_t $(t = 1,..., T)$, должны быть включены в разные классы G_i.

На данных ограничениях основывается и известный метод ветвей и границ [9], который дает решение задачи разбиения при меньших затратах времени. Алгоритм использует общую идею поиска с возвращением и включает четыре этапа:

1. Определение корня дерева поиска.
2. Выбор очередной размещаемой микрооперации.
3. Размещение очередной микрооперации.
4. Вычисление нижней границы решения.

На первом шаге алгоритма выбирается МК Y_t, содержащая максимальное количество микроопераций. Далее формируются корень дерева поиска, включающий m одноэлементных множеств, где $m = |Y_t|$. После выбора корня остается распределить $N - m$ микроопераций. Для упорядочивания процесса выбора очередной микрооперации задается очередь МО, основанная на показателе совместимости E_n.

$$E_n = |Q_n|, \qquad (4)$$

где Q_n - множество МО, совместимых с y_n. Первой в очередь размещается МО, имеющая минимальное значение показателя E_n. Классы G_i в свою очередь также упорядочиваются в соответствии с показателем Z_i, который выражает степень близости микроопераций из класса G_i с оставшимися нераспределенными МО.

$$Z_i = \left| \bigcap_{y_n \in G_i} Q_n \right|, \qquad (5)$$

то есть, первый в очереди стоит класс, позволяющий разместить максимальное количество нераспределенных МО. Если при выборе очередной МК Y_t принадлежащая ей МО y_n не может быть включена ни в одну из m групп из-за нарушения ограничения 2, то число групп разбиения увеличивается на единицу и производится попытка разбиения множества МО Y на $m+1$ группу. Процедура продолжается до нахождения разбиения множества МО на $m+q$ групп, удовлетворяющего условиям 1, 2 при минимально возможном значении q.

Рассмотрим применение данного метода на примере. Пусть некоторая ГСА Γ_1 состоит из микрокоманд, приведенных в табл. 1.

Таблица 1. Список микрокоманд ГСА Γ_1

МК	МО	МК	МО	МК	МО
Y_1	y_1, y_2, y_3	Y_6	y_2, y_5, y_8	Y_{11}	y_1, y_2, y_8, y_9
Y_2	y_1, y_4	Y_7	y_1, y_4, y_6, y_8	Y_{12}	y_2, y_7, y_9
Y_3	y_1, y_4, y_5	Y_8	y_1, y_2, y_3, y_7	Y_{13}	y_3, y_4, y_8
Y_4	y_1, y_2, y_6	Y_9	y_1, y_4, y_9, y_{10}	Y_{14}	y_3, y_7, y_{10}
Y_5	y_4, y_6, y_7	Y_{10}	y_2, y_5, y_{10}	Y_{15}	y_1, y_6, y_7, y_{10}

На первом шаге алгоритма из набора Y_7 с максимальным количеством МО создаются четыре класса: $G_1=\{y_1\}$, $G_2=\{y_4\}$, $G_3=\{y_6\}$, $G_4=\{y_8\}$. Для оставшихся МО определены множества совместимости Q_i:

$$Q_2=\{y_4\};\quad Q_7=\{y_5,y_8\};$$
$$Q_3=\{y_5,y_6,y_9\};\quad Q_9=\{y_3,y_5,y_6\};\quad (6)$$
$$Q_5=\{y_3,y_6,y_7,y_9\};\ Q_{10}=\{y_8\}.$$

Показатели совместимости для указанных множеств $E_2=E_{10}=1$, $E_7=2$, $E_3=E_9=3$, $E_5=4$, следовательно, очередь МО на распределение выглядит так:

$$y=(y_2,y_{10},y_7,y_3,y_9,y_5). \quad (7)$$

Для классов G_i определим показатели Z_i, означающие количество нераспределенных МО, попарно совместимых со всеми МО из класса G_i:

$$Z_1=0;\ Z_2=1;\ Z_3=3;\ Z_4=2. \quad (8)$$

Из (8) следует, что для размещения очередной МО на втором шаге алгоритма классы следует расположить в следующем порядке: G_3, G_4, G_2, G_1. Поскольку в классе G_3 и G_4 присутствуют несовместимые с y_2 микрооперации, распределение выполним в класс G_2. Продолжим распределение оставшихся микроопераций в классы, на каждом шаге алгоритма выполняя их упорядочивание в порядке убывания параметра Z_i. В том случае, если при распределении МО y_n нарушается ограничение 2 для всех имеющихся классов, то образуется новое одноэлементное множество $S_q=\{y_n\}$, после чего необходимо продолжить поиск решения с учетом нового множества. В таблице 2 приведено решение для разбиения множества микроопераций ГСА Γ_1 по описанному выше алгоритму.

Таблица 2. Анализ разбиения на множества совместимых микроопераций ГСА Γ_1

Множество, G_i	Эл-ты, y_i	Кол-во эл-тов, $\lvert G_i\rvert$	Разрядность кода, $\lceil\log_2\lvert G_i\rvert\rceil$
G_1	y_1	1	1
G_2	$\varnothing$, y_2, y_4	3	2
G_3	$\varnothing$, y_3, y_5, y_6, y_9	5	3
G_4	$\varnothing$, y_8, y_{10}	3	2
G_5	y_7	1	1

Общая разрядность операционной части составляет $B=1+2+3+2+1=9$ бит.

Современные микросхемы FPGA [5,6] включают блоки встроенной памяти, конфигурация которых (число слов N_W и число выходов N_O) может меняться при сохранении постоянной емкости

$$V_{EMB}=N_W\cdot N_O. \quad (9)$$

У типичных представителей семейства FPGA $V_{EMB}=4K$ бит при конфигурациях 4К*1, 2К*2, 1К*4, 512*8, то есть $N_O=\{1,2,4,8\}$. Для реализации схемы формирования микроопераций (СФМО) необходимы блоки памяти с количеством слов $N_W=2^{R_A}$, где $R_A=\lceil\log_2 M\rceil$, при этом число выходов t_F определяется как

$$t_F=\frac{V_{EMB}}{2^{R_A}}. \quad (10)$$

Таким образом, для реализации СФМО необходимо

$$n_1=\left\lceil\frac{B}{t_F}\right\rceil \quad (11)$$

блоков EMB, где B - общая разрядность операционной части.

Пусть ГСА Γ_1 содержит $M=300$ операторных вершин, тогда разрядность адреса $R_A=\lceil\log_2 300\rceil=9$. Число выходов блока EMB $t_F=4K/2^9=8$. Для реализации схемы блока БМО необходимо $n_1=\lceil 9/8\rceil=2$ блока EMB. Для определения эффективности использования блоков встроенной памяти введем показатель

$$\omega=\frac{V_{MP}}{V_{RAM}}, \quad (12)$$

где V_{MP} - объём микропрограммы, V_{RAM} - объём используемой памяти (в битах).

$$V_{MP}=M\cdot B, \quad (13)$$

$$V_{RAM}=n_1\cdot V_{EMB}. \quad (14)$$

Для рассмотренного примера объем микропрограммы равен $V_{MP}=300\cdot 9=2700$ бит, а используется $V_{RAM}=2\cdot 4K=8K=8192$ бита, $\omega=2700/8192=0.33$. Следовательно, лишь 33% от выделенного объема памяти используется.

Основная идея предлагаемого метода

Модификация данного метода разбиения множества микроопераций на подмножества совместимых микроопераций основана на том, что ограничения 1 и 2 снимаются, в наборах совместимых микроопераций искусственно создается некоторая избыточность, не приводящая к увеличению разрядности для кодирования подмножеств. Это определяет

возможность формировать одну микрооперацию из различных множеств при необходимости. Для дальнейшего описания методики предлагается ввести несколько определений.

Определение 1. ***Независимой*** называется микрооперация, которая входит только в один из классов G_i. Соответственно, микрооперация, которая входит больше, чем в один класс, называется ***зависимой.*** Множество независимых МО обозначим как Y_I, а зависимых как Y_D.

Определение 2. Микрооперации y_i и y_j называются ***условно-совместимыми***, если принадлежат к различным множествам ($y_i \in Y_I$, $y_j \in Y_D$, $i \neq j$), относятся к одному классу ($y_i, y_j \in G_k$), встречаются вместе в микрокоманде Y_n, и существует такой класс G_m, что $Y_n \cap G_m = y_j$.

Определение 3. Микрооперации y_i и y_j называются ***связанными***, если во всех вершинах ГСА они встречаются только вместе.

Модифицированный алгоритм разбиения множества МО на классы включает следующие шаги:

1. Выделить группы связанных микроопераций. В каждой группе оставить по одному элементу для уменьшения количества микроопераций и, как следствие, сложности алгоритма разбиения.

2. Найти микрокоманду Y_t такую, что $|Y_t| = \max(|Y_1|,...,|Y_N|) = m$. Сформировать m классов, в которые включить микрооперации $y_i \in Y_t$, $(i = 1,...,m)$.

3. Для оставшихся $N - m$ микроопераций определить множества совместимых микроопераций. Расположить нераспределенные микрооперации в порядке увеличения мощности соответствующих множеств. Зависимые МО расположить в конце списка.

4. Выполнять распределение согласно классическому алгоритму деления на классы совместимых микроопераций, выполняя анализ наборов микроопераций, которые содержат зависимые МО, в последнюю очередь.

5. При распределении зависимой МО распределить ее копии в минимально необходимое количество классов.

Применение алгоритма продемонстрируем на примере ГСА.

Использование предлагаемого метода возможно как для нахождения решения, начиная с первого шага, так и для модификации готового решения, найденного одним из известных методов. Применим его к решению, приведенному в табл. 2.

Эффективность предлагаемого алгоритма зависит от правильности выбора зависимых МО. В качестве предполагаемых зависимых микроопераций авторы статьи предлагают выбирать такие, которые образуют одноэлементные множества. В приведенном примере такими являются МО y_1 и y_7. Показатели совместимости для этих микроопераций $E_1 = 0$ и $E_7 = 2$. Выберем y_7 в качестве зависимой МО, поскольку больший показатель совместимости.

Для обеспечения возможности применения метода необходимо, чтобы предполагаемая микрооперация была попарно совместимой или условно-совместимой со всеми МО из множества Y. Микрооперация y_7 встречается в микрокомандах Y_5, Y_8, Y_{12}, Y_{14}, Y_{15}. Для каждой микрооперации найдем класс G_i такой, чтобы $Y_t \cap G_i = \varnothing$:

$Y_5 : \{G_1, G_4\}$, $Y_8 : \{G_4\}$, $Y_{12} : \{G_1, G_4\}$, $Y_{14} : \{G_1, G_2\}$, $Y_5 : \{G_2\}$.

Из рассмотрения необходимо убрать те множества, включение дополнительного элемента в которые увеличит разрядность, необходимую для кодирования. Поскольку класс G_1 является одноэлементным множеством, то его необходимо исключить:

$Y_5 : \{G_4\}$, $Y_8 : \{G_4\}$, $Y_{12} : \{G_4\}$, $Y_{14} : \{G_2\}$, $Y_5 : \{G_2\}$.

В результате выполнения алгоритма не получено пустых множеств, следовательно, МО y_7 является условно-совместимой с остальными элементами множества Y. После включения ее в классы G_2 и G_4 решение изменится так, как показано в табл. 3.

Таблица 3. Модификация разбиения на множества совместимых микроопераций ГСА

Множество, G_i	Эл-ты, y_i	Кол-во эл-тов, $\|G_i\|$	Разрядность кода, $\lceil \log_2 \|G_i\| \rceil$
G_1	y_1	1	1
G_2	$\varnothing$, y_2, y_4, y_7	3	2
G_3	$\varnothing$, y_3, y_5, y_6, y_9	5	3
G_4	$\varnothing$, y_8, y_{10}, y_7	3	2

Общая разрядность операционной части составляет $B = 1 + 2 + 3 + 1 = 8$ бит. Для ГСА, которая содержит $M = 300$ вершин, разрядность

адреса $R_A = \lceil \log_2 300 \rceil = 9$. Число выходов блока EMB $t_F = 4K/2^9 = 8$. Для реализации схемы блока БМО необходим $n_1 = \lceil 8/8 \rceil = 1$ блок EMB. Размер микропрограммы составляет $V_{MP} = 300 \cdot 8 = 2400$ бит, $V_{RAM} = 1 \cdot 4K = 4K$, показатель эффективности использования ресурсов памяти $\omega = 2400/4096 = 0.586$. Следовательно, в сравнении с применением классического метода снизились затраты ресурсов памяти (используется на 1 блок EMB меньше). Одновременно с этим повысилась эффективность использования занимаемых ресурсов (с 33 до 58,6 %).

Структурная схема блока формирования микроопераций для ГСА приведена на рис. 1.

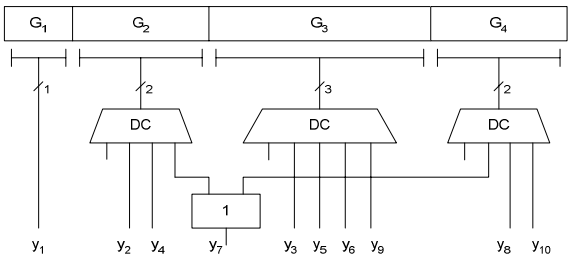

Рисунок 1 – Структурная схема формирования МО ГСА

Заключение

Предлагаемая модификация метода разбиения на классы основана на включении копий определенных микроопераций в различные классы совместимых микроопераций. При выполнении условий такой подход позволяет уменьшить разрядность операционной части микрокоманды в сравнении с классической стратегией кодирования полей совместимых микроопераций, что позволяет уменьшить аппаратурные затраты на реализацию таких автоматов на заказных микросхемах.

Необходимо отметить, что имеет место некоторое ухудшение временных показателей, поскольку в схему вводится подсхема формирования зависимых микроопераций. В ряде случаев такое ухудшение допустимо, если применение предложенного метода даст существенную экономию аппаратурных затрат при реализации схемы устройства. Также такой недостаток может не являться критичным при реализации небыстродействующих устройств, например, калькуляторов.

Практическая значимость предложенного метода заключается в уменьшении аппаратурных затрат в сравнении с известными методами реализации устройства управления.

Дальнейшие направления исследований связаны с формализацией алгоритма выбора зависимых МО с целью повышения эффективности предложенного метода.

Литература

1. De Micheli G. Synthesis and Optimization of Digital Circuits. – NY: McGraw-Hill, 1994. – 636 pp.
2. Грушвицкий Р.И., Мурсаев А.Х., Угрюмов Е.П. Проектирование систем с использованием микросхем программируемой логики. – СПб: БХВ. – Петербург, 2002. – 608 с.
3. Соловьев В.В. Проектирование цифровых схем на основе программируемых логических интегральных схем. – М.: Горячая линия-ТЕЛЕКОМ, 2001. – 636 с.
4. Baranov S. Logic Synthesis for Control Automata – Boston: Kluwer Academic Publishers, 1994 – 312 pp.
5. Баркалов А.А, Бабаков Р.М, Ахмад Бадер. Исследование аппаратурных характеристик автомата Мили с кодированием фрагмента микроопераций по VHDL-моделям // Исскуственный интеллект. – 2007, №1, - С. 117-122.
6. Баркалов О.О. Синтез операційних пристроїв. – Донецьк: РВА ДонНТУ, 2003. – 305 с.
7. Баркалов А.А., Палагин А.В. Синтез микропрограммных устройств управления. – Киев, 1997. – 135 с.
8. Майоров С.А., Новиков Г.И., Структура электронных вычислительных машин. – Л.: Машиностроение, 1979. – 384 с.
9. Bayer J., Koyama B. On the minimization of the width of the control memory of microprogramed processors // Ibid. – 1979. – №4. – P. 310-316.
10. Модифікація алгоритму Шварца для оптимізації операційної частини композиційних мікропрограмних пристроїв керування з урахуванням обмежень базису FPGA. Зеленьова І.Я. Мірошкін О.М., Казачанський А.В. // Наукові праці Донецького національного технічного університету. Серія «Обчислювальна техніка та автоматизація». Випуск 147 (16). / Редкол. Башков Є.О. (голова) та ін. – Донецьк: ДонНТУ, 2009 г. – С. 105-115.

Поступила в редакцию 12.03.10