Алгоритм компенсації кутової похибки, який пропонується в цій статті, позбавлений цих недоліків. Він полягає в наступному:

1. Під час калібрування вимірювальної частини комплексу у зовнішніх колах встановлюється фазовий зсув між напругою та струмом $\varphi = 0$ і визначаються косинусний K_{COS} та синусний K_{SIN} коефіцієнти:

$$\begin{aligned} K_{cos} &= \cos\Delta\gamma = \frac{P_\gamma}{UI} \\ K_{sin} &= \sin\Delta\gamma = \frac{Q_\gamma}{UI} \end{aligned} . \qquad (6)$$

2. Під час обчислень значень потужностей кутова похибка компенсується відповідно до таких співвідношень:

$$\begin{aligned} P &= P_\gamma K_{cos} + Q_\gamma K_{sin} \\ Q &= Q_\gamma K_{cos} - P_\gamma K_{sin} \end{aligned} . \qquad (7)$$

Дійсно, якщо підставити в (7) значення P_γ і Q_γ із (2) та K_{COS} і K_{SIN} із (6), отримаємо

$$\begin{aligned} P &= UI\cos(\varphi+\Delta\gamma)\cos\Delta\gamma + UI\sin(\varphi+\Delta\gamma)\sin\Delta\gamma = UI\cos\varphi \\ Q &= UI\sin(\varphi+\Delta\gamma)\cos\Delta\gamma - UI\cos(\varphi+\Delta\gamma)\sin\Delta\gamma = UI\sin\varphi \end{aligned} \qquad (8)$$

Запропонований алгоритм набагато простіший і ефективніший, ніж розглянутий перед ним, досить легко реалізується і не вимагає великих об'ємів обчислень. Він реалізований в програмному забезпеченні телекомплексів СПРУТ, впроваджених на підприємствах Західної енергосистеми.

1. Волынский Б.А., Зейн Е.Н., Шатерников В.Е.. Электротехника. – М., 1987.

УДК 681.3

Дунець Б.Р., Почаєвець О.М.
ДУ "Львівська політехніка", кафедра ЕОМ

МОДЕЛЬ АСОЦІАТИВНОЇ МАШИНИ ТА ЇЇ ЗАСТОСУВАННЯ ДЛЯ ЗАДАЧ МІНІМІЗАЦІЇ ЛОГІЧНИХ ФУНКЦІЙ

Запропоновано модифікацію асоціативної моделі, яка розширює можливості та усуває недоліки вже існуючих архітектур. Наведені фрагменти програм, які демонструють переваги запропонованої архітектури. Розглянуті питання практично реалізовані за допомогою програми-симулятора, в середовищі якої працює алгоритм мінімізації.

Вступ. Мінімізація ФАЛ є актуальною при створенні цифрових автоматів, що значною мірою стосується розробки інтелектуальної цифрової вимірювальної апаратури. Асоціативні машини з огляду мінімізації функцій алгебри логіки (ФАЛ) порівняно з машинами Ноймана мають такі переваги:

1. Зменшення часу мінімізації за рахунок великого ступеня паралельності апаратних ресурсів.

2. Зменшення обсягу програми та спрощення програмування внаслідок відсутності циклів обробки масивів даних.

При побудові модифікованої моделі асоціативної машини за основу беруться розглянуті в [1] три моделі асоціативних машин.

Модифікована модель. Недоліком існуючих моделей асоціативних машин є наявність тільки одного тегового розряду. В існуючих архітектурах цей недолік долається так: частина розрядів асоціативної комірки відводиться не під запам'ятовування корисної інформації, а під збереження копій тегового розряду. Під час роботи алгоритму ці розряди мають обмінюватись з реальним теговим бітом, внаслідок чого знижується швидкодія та штучно зменшується розрядність асоціативної комірки.

Перелічені недоліки долає запропонована в статті модифікована модель 32-розрядної асоціативної машини (рис.1). Блок службових регістрів асоціативної машини складається:

1. Comparand - регістр компаранду.
2. Comp_mask - регістр маски компаранду.
3. Tag_mask - регістр тегової маски.
4. F - теговий прапорець, що вказує на операцію перетворення багатьох тегових ознак на однобітний результуючий тег: "І" (F==0) або "ЧИ" (F==1).
5. R.Counter - лічильник кількості комірок-відповідачів.

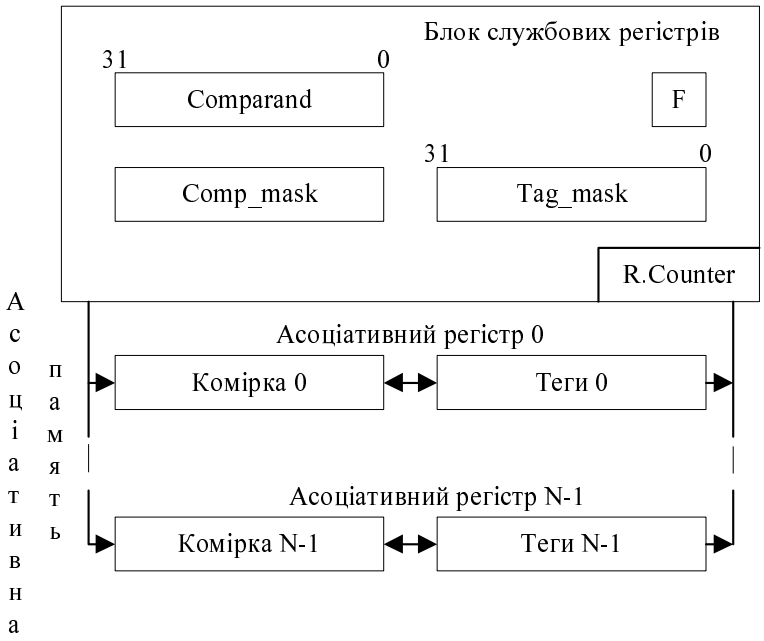

Рис.1. Модифікована модель асоціативної машини

Операції, що виконує асоціативна машина:

1. *SET* - одночасно встановлює в "1" лише ті тегові розряди усіх комірок, на які вказує тегова маска;

2. *COMPARE* - одночасно порівнює вміст усіх комірок з компарандом за розрядами, на які вказує маска компаранду; при негативному результаті порівняння тегові розряди комірки, що вказані у теговій масці, будуть скинуті в "0";

3. *WRITE* - копіює вміст компаранду паралельно в усі комірки – відповідачі за розрядами, на які вказує маска компаранду;

4. *FIRST* - серед комірок-відповідачів залишає лише одну; тегові розряди решти комірок, що вказані у теговій масці, будуть скинуті в "0";

5. *COUNT* - підраховує кількість комірок-відповідачів;

6. MOVE - паралельно переміщує тегові розряди усіх комірок, на які вказує тегова маска, циклічно на одну позицію в одному з двох протилежних напрямків.

Комірка є відповідачем, якщо:

1. При F==0 усі тегові розряди комірки, що вказані у теговій масці, встановлені в "1".

2. При F==1 хоча б один теговий розряд комірки серед тих, на які вказує тегова маска, встановлений в "1".

Запропонована асоціативна машина працює як окремий блок під керуванням машини Ноймана. Для цього машина Ноймана здійснює:

1. Запис значень до службових регістрів власними командами: =comparand, =comp_mask, =tag_mask, =flag.

2. Ініціалізацію виконання операцій асоціативної машини: set, compare, write, first, count, move.

3. Зчитування кількості комірок-відповідачів.

Реалізація методу Квайна-Мак-Класкі на асоціативній машині. З [2] відомо, що довільна ФАЛ подається у вигляді кубів з подальшою її мінімізацією за аналітичним методом Квайна-Мак-Класкі. При цьому вхідна ФАЛ задається в досконалій нормальній формі (ДНФ) як набір термів певної розрядності.

Мінімізація для вхідної комбінації (Рис.2) 01001, 01010, 01011, 01100, 01101, 01110, 01111, 11000, 11001, 11010, 11011, 11100, 11101, 11110 складається з двох кроків:

Крок 1. Знаходимо всі склейки за таблицею, кожна наступна колонка якої є результатом склейок термів з попередньої колонки. Усі несклеєні терми - це прості імпліканти для наступного кроку алгоритму.

Крок 2. Будуємо таблицю простих імплікантів, де вказуємо перекриття вхідних термів одержаними імплікантами. В даній таблиці знаходимо мінімальний варіант перекриття, попередньо вилучивши з розгляду особливі імпліканти, для яких є характерне те, що вхідний терм перекритий тільки одним із імплікантів.

Кожен з цих кроків можна паралельно обробити на асоціативній машині. Перший крок алгоритму базується на пошуку склейки серед N термів. Послідовне перебирання вимагає $N^2/2$ порівнянь, тоді як за допомогою асоціативної машини це можна реалізувати за N порівнянь. Тобто, маємо прискорення у часі в N/2 рази, а також відсутність одного вкладеного циклу. Причому, при отриманні склейок одного рівня з них треба вилучити тотожні – застосування асоціативної машини дає аналогічне прискорення в N/2 рази.

Другий крок алгоритму є головним. Його метою є вибір мінімальної комбінації імплікантів, яка перекриває усі вхідні терми. Перебираючи усі можливі варіанти, для кожного з них треба перевірити перекриття в середньому серед N/2 вхідних термів, що вимагає послідовного циклічного перебирання. Асоціативна машина робить цю перевірку за один крок. Тут також забезпечуємо прискорення в N/2 рази.

Отже, застосування універсальної асоціативної структури дає для задачі мінімізації загальну перевагу за швидкодією в N/2 рази, де N - кількість вхідних термів.

Крок 1

01001√	010-1√	01—1
01001√	01-01√	-10-1
01010√	-1001√	-1-01
01100√	0101-√	01-1-
11000√	01-10√	-101-
-----	-1010√	-1-10
01011√	0110-√	011--
01101√	011-0√	-110-
01110√	-1100√	-11-0
11001√	1100-√	110--
11010√	110-0√	11-0-
11100√	11-00√	11—0
-----	-----	
01111√	01-11√	
11011√	-1011√	
11101√	011-1√	
11110√	-1101√	
	0111-√	
	-1110√	
	110-1√	
	11-01√	
	1101-√	
	11-10√	
	1110-√	
	111-0√	

Крок 2

	11--0	11-0-	110--	-11-0	-110-	011--	-1-10	-101-	01-1-	-1-01	-10-1	01--1
01001	-	-	-	-	-	-	-	-	-	*	*	*
01010	-	-	-	-	-	-	*	*	*	-	-	-
01100	-	-	-	*	*	*	-	-	-	-	-	-
11000	*	*	*	-	-	-	-	-	-	-	-	-
01011	-	-	-	-	-	-	-	*	*	-	*	*
01101	-	-	-	-	*	*	-	-	-	*	-	*
01110	-	-	-	*	-	*	*	-	*	-	-	-
11001	-	*	*	-	-	-	-	-	-	*	*	-
11010	*	-	*	-	-	-	*	*	-	-	-	-
11100	*	*	-	*	*	-	-	-	-	-	-	-
01111	-	-	-	-	-	*	-	-	*	-	-	*
11011	-	-	*	-	-	-	-	*	-	-	*	-
11101	-	*	-	-	*	-	-	-	-	*	-	-
11110	*	-	-	*	-	-	*	-	-	-	-	-

Рис.2. Два кроки мінімізації ФАЛ за Квайном-Мак-Класкі

Для запропонованої асоціативної машини створена програма-симулятор, яка наочно демонструє виконання мінімізації функцій до 15 змінних з великим ступенем паралелізму (рис 3, 4). Ось лише кілька фрагментів виконання програми на прикладі мінімізації вже розглянутої функції (склейка 11-0-, отримана з терму 11101, подається в асоціативній пам'яті двома двійковими наборами, перший з яких вказує на несклеєні біти: 11010 і 11101):

Фрагмент 1: Знаходження склейки 01**0**-1 і 01**1**-1 => 01**--**1:

```
;в асоціативних комірках список склейок першого рівня
readnext     temp1      ;перший двійковий набір 11101
readnext     temp2      ;другий двійковий набір 01001
=tag_mask    1
set
=comparand   temp1*65536+temp2
=comp_mask   (temp1 & 11011b)*65537
compare
count        responder
if found     responder>=2
. . . . . .
found:                  ;склейку знайдено
```

Результат виконання дій (рис.3) – в молодших тегових розрядах помічено терми, що дали склейку.

```
|--------------Comparand---------------|  A.Regs:24                      |   R=2  |
0000_0000_0001_1101_0000_0000_0000_1001   Flag = 0 => AND
|--------------Comp_mask---------------|  |---------------Tag_mask--------------|
0000_0000_0001_1001_0000_0000_0001_1001   0000_0000_0000_0000_0000_0000_0000_0001

0000_0000_0001_1101_0000_0000_0000_1001|  |0000_0000_0000_0000_0000_0000_0000_0001
0000_0000_0001_1011_0000_0000_0000_1001|  |0000_0000_0000_0000_0000_0000_0000_0000
0000_0000_0000_1111_0000_0000_0000_1001|  |0000_0000_0000_0000_0000_0000_0000_0000
0000_0000_0001_1110_0000_0000_0000_1010|  |0000_0000_0000_0000_0000_0000_0000_0000
0000_0000_0001_1011_0000_0000_0000_1010|  |0000_0000_0000_0000_0000_0000_0000_0000
0000_0000_0000_1111_0000_0000_0000_1010|  |0000_0000_0000_0000_0000_0000_0000_0000
0000_0000_0001_1011_0000_0000_0000_1011|  |0000_0000_0000_0000_0000_0000_0000_0000
0000_0000_0000_1111_0000_0000_0000_1011|  |0000_0000_0000_0000_0000_0000_0000_0000
0000_0000_0001_1110_0000_0000_0000_1100|  |0000_0000_0000_0000_0000_0000_0000_0000
0000_0000_0001_1101_0000_0000_0000_1100|  |0000_0000_0000_0000_0000_0000_0000_0000
0000_0000_0000_1111_0000_0000_0000_1100|  |0000_0000_0000_0000_0000_0000_0000_0000
0000_0000_0001_1101_0000_0000_0000_1101|  |0000_0000_0000_0000_0000_0000_0000_0001
0000_0000_0000_1111_0000_0000_0000_1101|  |0000_0000_0000_0000_0000_0000_0000_0000
0000_0000_0001_1110_0000_0000_0000_1110|  |0000_0000_0000_0000_0000_0000_0000_0000
0000_0000_0000_1111_0000_0000_0000_1110|  |0000_0000_0000_0000_0000_0000_0000_0000
0000_0000_0001_1110_0000_0000_0001_1000|  |0000_0000_0000_0000_0000_0000_0000_0000
0000_0000_0001_1101_0000_0000_0001_1000|  |0000_0000_0000_0000_0000_0000_0000_0000
0000_0000_0001_1011_0000_0000_0001_1000|  |0000_0000_0000_0000_0000_0000_0000_0000
```

Рис.3. Результат виконання дій фрагменту 1

Фрагмент 2: Побудова таблиці простих імплікантів:

```
  ;в асоціативних комірках список вхідних термів
  =ii          0
s_loop:
  =tag_mask    2^ii
  set
  readnext     temp1     ;перший двійковий набір імпліканту
  readnext     temp2     ;другий двійковий набір імпліканту
  =comparand   temp2
  =comp_mask   temp1
  compare
  =ii          ii+1
  if s_loop    ii<m_quant ;<кількості простих імплікантів
```

Результат виконання (рис.4) – тегові розряди комірок є перекриттям таблиці простих імплікантів.

Рис.4. Результат виконання дій фрагменту 2

Висновок. Запропонована структура асоціативної машини є універсальною і її сумісне використання з машиною Ноймана надає значні переваги при розв'язанні задач, обробки великих масивів даних.

1. Фостер К. Ассоциативные параллельные процессоры. М., 1981. 2 Миллер Р. Теория переключательных схем. М., 1970.