5. *Бухарин С.Н.* Безопасность бизнеса-информационные войны. М.: Униар, 2006. 401 с.
6. *Бухарин С.Н., Глушков А.Г., Ермолаев И.Д.* Информационное противоборство Кн.1, Кн.2 М.: Полиори, 2005. 600 с.
7. *Чалдин Р.* Психология влияния. СПб.: ПитерКом, 1999. 346 с.

Поступила 18.02.2013р.

УДК 62.507

В.В. Мінзюк, Львів

МЕТОД ПОШУКУ ПРОСТИХ КОН'ЮНКТИВНИХ ТЕРМІВ БУЛОВИХ ФУНКЦІЙ ПОБІТОВИМ РОЗБИТТЯМ МНОЖИНИ

Abstract. In this paper the bitwise dividing method for searching prime conjuncterms of Boolean functions has been proposed. The method based on procedure of searching boundary in rank-ordered set between conjuncterms wich has 0 vs 1 in some bit.

Вступ

У процесі логікового синтезу комбінаційних мереж та цифрових автоматів виникає проблема мінімізації булових функцій n змінних. Її невід'ємною складовою є задача швидкого пошуку множини усіх простих кон'юнктивних термів (кон'юнктермів). При застосуванні відомих методів [1-6] у процесі пошуку виникає тавтологія, що неминуче спричиняє надлишкові витрати обчислювальних ресурсів. Описаний у [7] метод порозрядного вирощування трійкового дерева пошуку простих кон'юнктермів позбавлений цього недоліку. А його модифікація [8] використовуючи певні закономірності двійкового простору дає змогу у ряді випадків зменшити кількість порівнянь. Однак у методі порозрядного вирощування необхідно здійснювати сортування кожної одержаної множини по нулях і одиницях у розглядуваному біті. З урахуванням того факту, що від самого початку пошуку множина кон'юнктермів є упорядкованою, це виглядає не раціональним. Причина цього недоліку криється у порядку аналізу бітів. А саме, на кожному етапі розглядається наймолодший біт. Подальші розвідки у напрямку зменшення обчислювальних витрат призвели до появи методу, запропонованого у цій роботі.

Метод побітового розбиття

Нехай задано булову функцію множиною мінтермів, на яких функція приймає значення “1”, та множиною мінтермів, на яких функція недовизначена. Впорядкуємо змінні у кон'юнктермах за наростанням номеру справа наліво. Кодуємо прямі та інверсні змінні відповідно одиницею та нулем. Крім того кожний кон'юнктерм описується кодом помітки. Встановлюємо нулі в усіх бітах коду помітки мінтермів, на яких функція приймає значення "1". У коді помітки кон'юнктермів, на яких функція недовизначена, встановлюємо одиниці в усіх бітах. Об'єднуємо ці дві множини та впорядковуємо одержану за природним порядком чисел (за наростанням). Розраховуємо код помітки вихідної множини як побітову кон'юнкцію кодів помітки усіх її елементів.

1. Перевірка потужності множини.

Якщо у множині є лише один елемент і код помітки множини дорівнює нулю, то цей елемент заноситься у множину простих кон'юнктермів. Якщо код помітки відмінний від нуля, то множина вилучається з розгляду.

Якщо у множині є тільки два елементи, то пара перевіряється на склеювання.

2 Процедура розбиття множини. Ця процедура застосовується до множин, що містять більше двох елементів.

Нехай біт із номером m є найстаршим із тих, що ще не розглядалися для кон'юнктермів вибраної множини. (Нумерація бітів у кон'юнктермі починається з крайнього правого біта із номером 0.) За бітом із номером m розіб'ємо множину на дві підмножини. У першу заносимо усі кон'юнктерми, що містять нуль у вибраному біті, у другу – з одиницею у цьому біті.

3. Якщо кількість елементів в одній з одержаних підмножин становить 2^m , то ця підмножина може бути замінена кон'юнктермом, у якого поглинуті усі біти молодші від m. Тоді у другій підмножині розглядуваний біт замінюють символом поглинання і визначають коди помітки одержаних множин. Інакше переходять до пункту 4.

Для визначення кодів помітки у випадку склеювання множини в один кон'юнктерм візьмемо крайній лівий елемент множини, у кон'юнктермах якої біт із номером m замінений символом поглинання. У другій множині починаючи з крайнього лівого елемента шукаємо такий, що міститься у вибраному елементі першої множини. Знайшовши, розраховуємо побітову кон'юнкцію кодів помітки цих кон'юнктермів. Це і є новий код помітки вибраного елемента першої множини. При цьому в коді помітки знайденого елемента другої множини встановлюємо одиницю у розглядуваному біті. Тепер беремо наступний елемент першої множини та здійснюємо пошук у другій множині так само починаючи з наступного після останнього знайденого. Процедура повторюється, доки у першій множині не закінчаться елементи. Тоді переходимо до пункту 5.

4. Якщо кількість елементів у жодній з одержаних підмножин не дорівнює 2^m , то здійснюємо пошук множини кон'юнктермів, у яких розглядуваний біт поглинуто.

Процедура пошуку склеювань кон'юнктермів розбитої множини.

Одержано впорядковану множину кон'юнктермів, що містить нуль у біті з номером m, а також впорядковану множину кон'юнктермів, що містить одиницю у цьому біті. Тепер знайдемо множину кон'юнктермів, що склеюються у розглядуваному біті.

Візьмемо крайні ліві кон'юнктерми із одержаних під час розбиття множин.

У кон'юнктермі із другої множини, де розглядуваний біт має значення 1, обнулюємо цей біт. В обох кон'юнктермах замінюємо символи поглинання нулями. Тепер знаходимо різницю між одержаними числами (друге число віднімаємо від числа, що одержано з першої множини).

Якщо різниця дорівнює нулю, то кон'юнктерми склеюються. Заносимо відповідний кон'юнктерм у результуючу множину. Код помітки одержаного кон'юнктерма дорівнює побітовій кон'юнкції кодів помітки вихідних кон'юнктермів. А у кодах помітки вихідних кон'юнктермів встановлюємо одиницю у біті з номером m .

Якщо різниця є меншою від нуля, то беремо наступний кон'юнктерм із першої множини і знов виконуємо обнулення бітів із подальшим порівнянням чисел.

Якщо різниця є більшою від нуля, то беремо наступний кон'юнктерм із другої множини і знов виконуємо обнулення бітів із подальшим порівнянням чисел.

Процедура завершується, коли хоча б в одній з множин закінчаться елементи.

В результаті аналізу вихідної множини одержано три нові множини: кон'юнктерми першої містять нуль у розглядуваному біті, другої – одиницю, а третьої – символ поглинутої змінної.

5. Розраховуємо коди помітки кожної одержаної множини як побітову кон'юнкцію кодів помітки її елементів.

Якщо множина замінюється одним кон'юнктермом, код помітки останнього встановлюється рівним коду помітки заміненої множини.

Множину вилучають із подальшого розгляду у разі, якщо її код помітки відмінний від нуля. У подальшому кожну із множин аналізують окремо по біту, молодшому від розглянутого.

Кон'юнктерм вважається простим, якщо після завершення розгляду усіх утворених множин його код помітки дорівнює нулю.

Приклад

Нехай задано недовизначену булову функцію чотирьох змінних

$$\begin{cases} Y^1 = \{(1),(2),(4),(6),(7),(8),(9),(11)\}^1 \\ Y^\sim = \{(0),(3),(15)\}^\sim \end{cases}$$

Розв'язання:

Для наочності проілюструємо розв'язування у числовій формі, а також у псевдотрійковому зображенні. Для псевдотрійкового зображення позначатимемо двійкові коди помітки над відповідними розрядами цього зображення.

Об’єднаємо множини кон’юнктермів Y^1 та $Y^{\sim}$.

В одержаній множині упорядкуємо числа за їх природним порядком.

Для кон’юнктермів із множини Y^1 встановимо коди помітки 0 (0000 – нулі в усіх бітах). Це означає, що кон’юнктерм ще не приймав участі у склеюванні і на даному етапі є істотним.

Для кон’юнктермів із множини $Y^{\sim}$ встановимо коди помітки 15 (1111 – одиниці в усіх бітах). Це означає, що кон’юнктерм є неістотним (тобто не обов’язковий для врахування у кінцевому результаті).

Розрахуємо код помітки об’єднаної множини як побітову кон’юнкцію кодів помітки її елементів. Одержимо 0 (0000 – нулі в усіх бітах коду помітки множини). Числове та двійкове зображення об’єднаної упорядкованої множини кон’юнктермів із кодами помітки подано на рис. 1.

Здійснимо розбиття множини за найстаршим (третім) бітом. Для цього у послідовності кон’юнктермів у псевдотрійковому зображенні знайдемо межу, де третій біт змінює своє значення. Це кон’юнктерми (0111) та (1000). В інший спосіб можна у послідовності кон’юнктермів у числовому зображенні знайти межу між кон’юнктермами, що є меншими 8 (2^3) і не меншими від 8. Це, відповідно, кон’юнктерми (7) та (8) вихідної множини.

Ми одержали множину кон’юнктермів, що містять нуль у третьому біті, а також множину кон’юнктермів, що містять одиницю у цьому біті (рис. 2). Тепер знайдемо множину кон’юнктермів, що склеюються у розглядуваному біті.

Візьмемо крайні ліві кон’юнктерми із одержаних множин. Це (0000) та (1000). У числовій формі запису це (0) та (8) (наймолодші у своїх множинах). У кон’юнктермі із другої множини, де розглядуваний біт має значення 1, обнулюємо цей біт. Тепер порівнюємо одержані кон’юнктерми: (0000) і (0000). Оскільки вони однакові, то у шукану множину заносимо кон’юнктерм (-000). У числовій формі описана процедура набуває наступного вигляду: від першого числа слід відняти друге і додати 8 (2^3). Остання дія аналогічна обнуленню третього біта у кон’юнктермі, що містить одиницю у цьому біті. Одержимо 0-8+8=0, значить результуюча множина містить кон’юнктерм (0,8).

Код помітки одержаного кон’юнктерма (-000) дорівнює побітовій кон’юнкції кодів помітки вихідних кон’юнктермів: 1111&0000=0000, де “&” – символ бінарної операції побітової кон’юнкції двійкових чисел. Оскільки вихідні кон’юнктерми прийняли участь у склеюванні по третьому біту, то у їх кодах помітки встановлюємо одиницю у третьому біті. В результаті одержимо $\overset{1111}{(0000)}$, $\overset{1000}{(1000)}$, $\overset{0000}{(-000)}$. У числовій формі маємо $(0)^{15}$, $(8)^{8}$, $(0,8)^{0}$.

Візьмемо для порівняння наступний елемент із кожної множини: $\overset{0000}{(0001)}$ та $\overset{0000}{(1001)}$. В результаті одержимо $\overset{1000}{(0001)}$, $\overset{1000}{(1001)}$, $\overset{0000}{(-001)}$.

$$\{(0)^{15},\quad (1)^{0},\quad (2)^{0},\quad (3)^{15},\quad (4)^{0},\quad (6)^{0},\quad (7)^{0},\quad (8)^{0},\quad (9)^{0},\quad (11)^{0},\quad (15)^{15}\}^{0}$$

$$\{\overset{1111}{(0000)},\ \overset{0000}{(0001)},\ \overset{0000}{(0010)},\ \overset{1111}{(0011)},\ \overset{0000}{(0100)},\ \overset{0000}{(0110)},\ \overset{0000}{(0111)},\ \overset{0000}{(1000)},\ \overset{0000}{(1001)},\ \overset{0000}{(1011)},\ \overset{1111}{(1111)}\}^{0000}$$

Рис. 1 Вихідна множина кон’юнктермів

$$\{(0)^{15},\quad (1)^{0},\quad (2)^{0},\quad (3)^{15},\quad (4)^{0},\quad (6)^{0},\quad (7)^{0}\},\quad \{(8)^{0},\quad (9)^{0},\quad (11)^{0},\quad (15)^{15}\}$$

$$\{\overset{1111}{(0000)},\ \overset{0000}{(0001)},\ \overset{0000}{(0010)},\ \overset{1111}{(0011)},\ \overset{0000}{(0100)},\ \overset{0000}{(0110)},\ \overset{0000}{(0111)}\},\ \{\overset{0000}{(1000)},\ \overset{0000}{(1001)},\ \overset{0000}{(1011)},\ \overset{1111}{(1111)}\}$$

Рис. 2 Результат розбиття вихідної множини по третьому біту

$$\{(0)^{15},\quad (1)^{8},\quad (2)^{0},\quad (3)^{15},\quad (4)^{0},\quad (6)^{0},\quad (7)^{8}\}^{0},\quad \{(8)^{8},\quad (9)^{8},\quad (11)^{8},\quad (15)^{15}\}^{8}$$

$$\{\overset{1111}{(0000)},\ \overset{1000}{(0001)},\ \overset{0000}{(0010)},\ \overset{1111}{(0011)},\ \overset{0000}{(0100)},\ \overset{0000}{(0110)},\ \overset{1000}{(0111)}\}^{0000},\ \{\overset{1000}{(1000)},\ \overset{1000}{(1001)},\ \overset{1000}{(1011)},\ \overset{1111}{(1111)}\}^{1000}$$

$$\{(0,8)^{0},\quad (1,9)^{0},\quad (3,11)^{0},\quad (7,15)^{0}\}^{0}$$

$$\{\overset{0000}{(-000)},\ \overset{0000}{(-001)},\ \overset{0000}{(-011)},\ \overset{0000}{(-111)}\}^{0000}$$

Рис. 3 Результат склеювання по третьому біту

Знов візьмемо наступні елементи: $\overset{0000}{(0010)}$ та $\overset{0000}{(1011)}$. У числовій формі їм відповідють $(2)^0$ та $(11)^0$. Оскільки 2-11+8=-1 < 0, то елемент із першої множини замінюємо наступним і повторюємо операцію: 3-11+8=0. Отже одержимо: $(3)^{15}$, $(11)^8$, $(3,11)^0$.

Продовжуємо порівняння доки в одній із множин не закінчаться елементи. В результаті одержимо три множини. Перша містить кон’юнктерми із нулем у розглядуваному біті, друга – із одиницею, а третя – із символом поглинання (рис. 3).

Для кожної із трьох множин, одержаних на етапі розбиття та склеювання по третьому біту, розрахуємо код помітки як побітову кон’юнкцію кодів помітки усіх елементів множини. Для множини кон’юнктермів, що містять одиницю у розглянутому біті, одержимо код помітки 1000. Оскільки код помітки множини відмінний від нуля, ця множина вилучається із подальшого розгляду. Для множин, кон’юнктерми яких містять нуль чи символ поглинання у третьому біті, код помітки становить 0 (0000 – нулі в усіх бітах коду помітки множини). У подальшому ці множини розглядатимемо окремо.

Наступною розглянемо множину $\{(0)^{15}, (1)^8, (2)^0, (3)^{15}, (4)^0, (6)^0, (7)^8\}^0$. Після розбиття по другому біту одержимо:

$$\{(0)^{15},\quad (1)^{8},\quad (2)^{0},\quad (3)^{15}\},\quad \{(4)^{0},\quad (6)^{0},\quad (7)^{8}\}$$

$$\{\overset{1111}{(0000)},\ \overset{1000}{(0001)},\ \overset{0000}{(0010)},\ \overset{1111}{(0011)}\},\quad \{\overset{0000}{(0100)},\ \overset{0000}{(0110)},\ \overset{1000}{(0111)}\}$$

Множина $\{(0)^{15}, (1)^8, (2)^4, (3)^{15}\}$ містить 2^2 елементів, значить, може бути замінена кон’юнктермом (0,1,2,3) (у псевдотрійковому зображенні – (00- -)). Тоді у псевдотрійковому зображенні елементів другої множини другий біт замінюється знаком поглинання. (У числовому зображенні кожен кон’юнктерм доповнюється числами, меншими на 2^2 від тих, що уже містяться в кон’юнктермі.) Друга множина набуває вигляду

$$\{(0,4)^{0},\quad (2,6)^{0},\quad (3,7)^{8}\}$$

$$\{\overset{0000}{(0-00)},\ \overset{0000}{(0-10)},\ \overset{1000}{(0-11)}\}$$

Знайдемо коди помітки кон’юнктермів двох одержаних множин. Для цього візьмемо крайній лівий елемент другої множини $\overset{0000}{(0-00)}$ ($(0,4)^0$). У першій множині починаючи з крайнього лівого елемента шукаємо такий, що міститься у вибраному елементі другої множини. Знайшовши $\overset{1111}{(0000)}$ ($(0)^{15}$), розраховуємо побітову кон’юнкцію кодів помітки цих кон’юнктермів: 0000&1111=0000. Це і є новий код помітки вибраного елемента другої множини. При цьому в коді помітки знайденого елемента першої множини встановлюємо одиницю у розглядуваному біті. Тепер беремо наступний елемент другої множини та здійснюємо пошук у першій множині так само починаючи з наступного після останнього знайденого. Процедура повторюється, доки у другій множині не закінчаться елементи. Тоді розраховуємо коди помітки одержаних множин. Таким чином одержимо:

$$\{(0)^{15},\quad (1)^{8},\quad (2)^{4},\quad (3)^{15}\}^{0},\quad \{(0{,}4)^{0},\quad (2{,}6)^{0},\quad (3{,}7)^{8}\}^{0}$$

$$\{\overset{1111}{(0000)},\ \overset{1000}{(0001)},\ \overset{0100}{(0010)},\ \overset{1111}{(0011)}\}^{0000},\quad \{\overset{0000}{(0-00)},\ \overset{0000}{(0-10)},\ \overset{1000}{(0-11)}\}^{0000}$$

Код помітки кон'юнктерма (0,1,2,3) дорівнює коду помітки заміненої множини. Отже цей кон'юнктерм є простим, а відповідна множина вилучається із подальшого розгляду.

Розглянемо другу з одержаних на попередньому етапі множин $\{(0,4)^0, (2,6)^0, (3,7)^8\}^0$. Розіб'ємо її на підмножини по першому біту. У псевдотрійковому зображення ця операція проводиться так само як і на попередніх етапах. А у числовому зображенні у кожному кон'юнктермі необхідно проаналізувати найменше число. У першу множину слід помістити кон'юнктерми, у яких наймолодше число менше 2^1, а решта – у другу.

$$\{(0{,}4)^{0}\},\quad \{(2{,}6)^{0},\quad (3{,}7)^{8}\}$$

$$\{\overset{0000}{(0-00)}\},\quad \{\overset{0000}{(0-10)},\quad \overset{1000}{(0-11)}\}$$

Друга множина містить 2^1 елементів, тому замінюється кон'юнктермом (0-1-) (у числовому зображенні (2,3,6,7)). Тоді в кон'юнктермі першої множини перший біт замінюємо символом поглинання. Після розрахунку кодів помітки одержимо два простих кон'юнктерми:

$$(0{,}2{,}4{,}6)^{0},\quad (2{,}3{,}6{,}7)^{0}$$

$$\overset{0000}{(0--0)},\quad \overset{0000}{(0-1-)}$$

Тепер розглянемо множину, одержану при склеюванні по третьому біту (рис. 3): $\{(0,8)^0, (1,9)^0, (3,11)^0, (7,15)^0\}^0$. Після розбиття маємо

$$\{(0{,}8)^{0},\quad (1{,}9)^{0},\quad (3{,}11)^{0}\},\quad \{(7{,}15)^{0}\}$$

$$\{\overset{0000}{(-000)},\quad \overset{0000}{(-001)},\quad \overset{0000}{(-011)}\},\quad \{\overset{0000}{(-111)}\}$$

Перевірка склеювання здійснюється так само як і на попередніх етапах лише з одним уточненням. У числовому зображенні необхідно взяти для перевірки наймолодше число із кожного кон'юнктерма. Внаслідок склеювання одержимо:

$$\{(0{,}8)^{0},\quad (1{,}9)^{0},\quad (3{,}11)^{4}\}^{0},\quad \{(7{,}15)^{4}\}^{4}\quad \{(3{,}7{,}11{,}15)^{0}\}^{0}$$

$$\{\overset{0000}{(-000)},\quad \overset{0000}{(-001)},\quad \overset{0100}{(-011)}\}^{0},\quad \{\overset{0100}{(-111)}\}^{0100}\quad \{\overset{0000}{(--11)}\}^{0000}$$

Код помітки другої множини відмінний від нуля, тому вона надалі не розглядатиметься. Третя множина містить лише один елемент (- -11) (у числовому зображенні – (3,7,11,15)) із кодом помітки 0. Тому відповідний кон'юнктерм є простим.

Розглянемо множину $\{(0,8)^0, (1,9)^0, (3,11)^4\}^0$. Після розбиття по першому біту одержимо:

$$\{(0,8)^0 \quad (1,9)^0\}, \quad \{(3,11)^4\}$$

$$\{\overset{0000}{(-000)} \quad \overset{0000}{(-001)}\}, \quad \{\overset{0100}{(-011)}\}$$

Перша з одержаних множин містить 2^1 елементів, тому замінимо її відповідним кон'юнктермом (-00-) (у числовому зображенні – (0,1,8,9)). Тоді у кон'юнктермі другої множини перший біт замінюємо символом поглинання. Після розрахунку кодів помітки одержимо два прості кон'юнктерми:

$$(0,1,8,9)^0, \quad (1,3,9,11)^0$$

$$\overset{0000}{(-00-)}, \quad \overset{0000}{(-0-1)}$$

У підсумку множина простих кон'юнктермів заданої функції у псевдотрійковому зображенні:

{(00- -), (0-1-) , (0- -0), (- -11), (-00-), (-0-1)},

а у числовому зображенні:

{(0,1,2,3), (2,3,6,7), (0,2,4,6), (3,7,11,15), (0,1,8,9), (1,3,9,11)}.

Висновки

На відміну від [7] запропонований метод пошуку простих кон'юнктермів не потребує сортування кожної розглядуваної множини. Натомість виконується лише пошук межі між нулями та одиницями у заданому біті.

Аналіз кожної множини призводить до декомпозиції задачі, що дає змогу ефективно використовувати багатопроцесорні системи для пошуку розв'язку.

Для реалізації обчислень також можна використати маскове зображення кон'юнктермів [9]. При цьому спрощується реалізація порівнянь кон'юнктермів. А використаня методу побітового сортування зі склеюванням [10] дасть змогу одержати частину кон'юнктермів нижчого рангу вже на етапі впорядкування вихідної множини.

1. *McCluskey E.J.* Minimization of Boolean Functions / E.J. McCluskey // Bell System Technical Journal. – 1956. – Vol. 35. – №6. – P. 1417-1444.

2. *Quine W.V.* The Problem of Simplifying Truth Functions / W.V. Quine // The American Mathematical Monthly. – The Mathematical Association of America, Oct. 1952. – Vol. 59. – No. 8. – P. 521-531.

3. *Закревский А.Д.* Алгоритмы синтеза дискретных автоматов. / А.Д. Закревский. – М.: Наука, 1971. – 512 с.

4. *Рицар Б.Є.* Мінімізація булових функцій методом розчеплення кон'юнктермів /

Б.Є. Рицар // Управляющие системы и машины. – 1998. – №5. – С.14-22.
5. *Hwa H.R.* A method for generating prime implicants of a boolean expressions. / H.R. Hwa // IEEE Transactions on Electronic Computers. – New York: The Institute of Electrical and Electronics Engineers, Inc., Jun. 1974. – Vol. C-23. – №6. – P. 637-641.
6. *Svoboda A.* Ordering of implicants. / A. Svoboda // IEEE Transactions on Electronic Computers. – New York: The Institute of Electrical and Electronics Engineers, Inc., Feb. 1967. – Vol. EC-16. – №1. – P. 100-105.
7. *Лузин С.Ю.* Асимптотически оптимальный метод получения простых импликант. / С.Ю. Лузин // Автоматика и вычислительная техника. – 2000. – №1. – С. 80-84.
8. *Мінзюк В.В.* Модифікація методу порозрядного вирощування простих кон'юнктивних термів булових функцій. / В.В. Мінзюк // Збірник наукових праць Інституту проблем моделювання в енергетиці ім. Г.Є. Пухова. – К.: ІПМЕ ім. Г.Є. Пухова НАН України, 2012. – Випуск 65. – С. 129-134.
9. *Мінзюк В.В.* Спосіб синтезування кон'юнктермів булових функцій. / В.В. Мінзюк // Вісник Національного університету "Львівська політехніка" Радіоелектроніка та телекомунікації. – 2004. – № 508. – С. 256-262.
10. *Мінзюк В.В.* Спосіб сортування цілих чисел для задач мінімізації булових функцій. / В.В. Мінзюк // Вісник Національного університету "Львівська політехніка" Радіоелектроніка та телекомунікації. – 2011. – № 705. – С. 135-137.

Поступила 27.02.2013р.

УДК 621.3

Р.В. Бачинський, НУ "Львівська політехніка", ІКТА, каф. ЕОМ, м. Львів

ВИМІРЮВАННЯ НАПРУГИ В БАГАТОКОМІРКОВИХ БАТАРЕЯХ ЖИВЛЕННЯ

В статті розглянуто особливості побудови пристрою для вимірювання напруги окремих комірок в багатокоміркових батареях живлення, на основі схеми з плаваючим потенціалом землі.

В статье рассмотрены особенности устройства для измерения напряжения отдельных ячеек в многоячеечных батареях питания, на основе схемы с плавающим потенциалом земли.

This article describes device functionality for measuring individual cell voltage level in multi-cell batteries. The device uses schematic with floating ground potential.

Вступ

Високовольтні батареї живлення з великою кількістю комірок використовуються в різних медичних та промислових системах. Для забезпечення надійного функціонування таких систем необхідно контролювати абсолютне значення напруги кожної комірки та різницю напруг