УДК: 681.325
В.П.Тарасенко, О.К.Тесленко, Я.М.Клятченко
Національний технічний університет України «Київський політехнічний інститут»

СТРУКТУРИ ДЛЯ ПЛІС РЕАЛІЗАЦІЙ ДЕТАЛЬНОГО АДАПТИВНОГО ПОРІВНЯННЯ ПОСЛІДОВНОСТЕЙ СИМВОЛІВ

В статті розглянуті апаратні структури для створення пристрою на базі сучасних програмованих логічних інтегральних схем, що реалізує алгоритм детального адаптивного порівняння послідовностей символів.

Ключові слова: *логічні інтегральні схеми, алгоритми порівняння, адаптивне порівняння, послідовність символів.*

Вступ

Розвиток інформаційних технологій забезпечує суттєве покращення якості освіти, але привносить також додаткові проблеми, до яких належить контроль запозичень, які можуть широко використовуватись (і використовуються) в студентських роботах. На даний час існує ряд програмних систем, наприклад [1-3], призначених автоматизувати процес знаходження запозичень. Логіка розвитку подібних систем, безумовна перспективність та практична важливість даного напрямку в інформаційних технологіях, особливо освітніх, стимулюють дослідження апаратних реалізацій алгоритмів виявлення запозичень. Практичне використання таких досліджень ґрунтується на розвитку та широкому застосуванні програмованих логічних інтегральних схем (ПЛІС).

Постановка задачі

Одним з алгоритмів виявлення запозичень є алгоритм адаптивного порівняння [4], де запозичення виявляються шляхом порівняння окремих символів (наприклад – байтів) з обмеженням деякої довжини співпадання, яка не береться до уваги. Оскільки в даному алгоритмі особливості семантичної структури даних не використовуються, то він є універсальним щодо інформаційних об’єктів різної природи. Тобто, за його допомогою можна виявляти запозичення в україномовних текстах, так і більшості інших мов, запозичення в аудіо та відео файлах, програмах на мовах програмування та виконавчих програмах і т.п. В алгоритмі адаптивного порівняння, файл даних, який перевіряється на наявність запозичень, розглядається як послідовність B символів із множини Q = {0,1,…,255} довжиною n_b, а файл даних із бази зразків – як послідовність A символів з Q довжиною n_a. Крім того, встановлюється значення d – мінімальна довжина збігу двох підпослідовностей символів, яка береться до уваги. Тут і надалі підпослідовністю деякої послідовності називається безперервна її частина. Значення d може підбиратись із врахуванням особливостей подання даних в інформаційних об’єктах.

Основною проблемою систем виявлення запозичень є мінімізація часу, необхідного для обробки піддослідного файлу. Верхньою оцінкою часу на порівняння двох файлів є час пропорційний $n_a \times n_b$. Враховуючи великий обсяг бази зразків, який із кожним роком збільшується, для виявлення запозичень може знадобитись великий проміжок часу, недопустимий для широкого практичного використання. В алгоритмі адаптивного порівняння був запропонований етап швидкісного порівняння [4] (швидкісне адаптивне порівняння). Основна ідея швидкісного етапу полягає у формуванні не точного, а приблизного рівня запозичень, але який гарантовано не менший за реально існуючий. На етапі швидкісного порівняння значення мінімальної довжини збігу, яка береться до уваги, визначається наступним чином $d = n_b$ div const. При цьому верхня оцінка затрат часу на швидкісне порівняння пропорційна const× n_a. Конкретне значення константи const обумовлюється особливостями структури піддослідного інформаційного об’єкта. Швидкісне адаптивне порівняння дозволяє попередньо «відсіяти» значний обсяг даних із бази зразків за значно коротший обсяг часу. Швидкісне порівняння є своєрідним аналогом порівняння двох однакових за розміром файлів на повне співпадання за методом контрольних сум. У цьому випадку нерівність контрольних сум гарантує неспівпадання файлів, при рівності контрольних сум необхідні додаткові дослідження.

Для точного визначення рівня запозичень використовується етап детального адаптивного порівняння піддослідної послідовності з «підозрілими», виділеними на швидкісному етапі. В [5]

запропоновано варіанти ПЛІС реалізацій детального адаптивного порівняння з оцінками апаратних затрат та швидкодії. Для практичного застосування вказаних результатів необхідно провести більш детальний аналіз можливих апаратних структур, який може бути використаний для формування відповідної методики.

Метод розв'язання задачі

Процес виявлення запозичень у піддослідній послідовності полягає в багатократному її порівнянні з різними послідовностями зразками. Згідно з [5] для реалізації детального адаптивного порівняння необхідне виконання наступних операцій.

1. Знаходження елементарних збігів.
2. Виділення суттєвих збігів.
3. Формування результату як просторового коду.
4. Згортка просторового коду.

Розглянемо конкретизацію одного з варіантів запропонованих в [5] ПЛІС реалізацій детального адаптивного порівняння. При цьому будемо мати на увазі, що пошук запозичень характеризується багатократним порівнянням піддослідної послідовності з послідовностями з бази зразків.

Для одержання загальних порівняльних оцінок апаратних витрат та швидкодії різних варіантів реалізації в [5] було прийнято, що $n_a = n_b = n$. У реальних умовах це, звичайно, не так, тому при $n_a \geq n_b$ першопочаткова структура пристрою буде мати наступний вигляд (Рис.1)

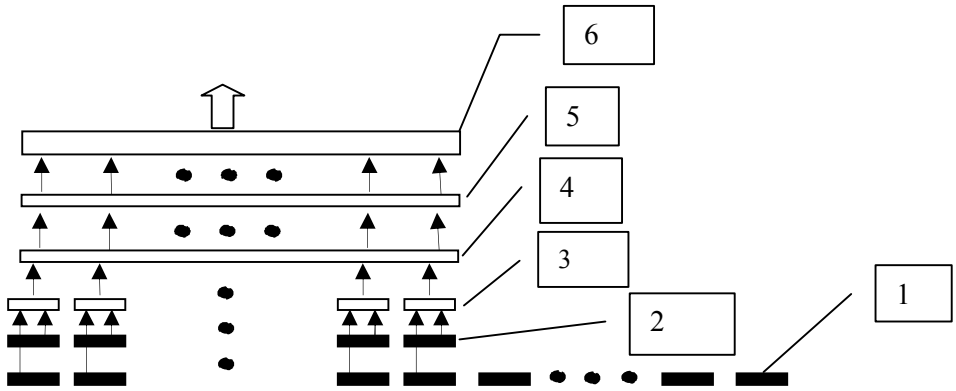

Рис. 1. Деталізація при $n_a \geq n_b$

Цифрами на рис. 1 позначено:

1-регістр циклічного зсуву на n_a символів, у який попередньо завантажується оригінальна послідовність;

2-регістр на n_b символів, у який попередньо завантажується піддослідна послідовність;

3-схеми порівняння символів (знаходження елементарних збігів);

4-схема виділення суттєвих збігів;

5-регістр згортки просторового коду;

6-схема згортки просторового коду.

Пристрій працює наступним чином. На першому такті роботи (після завантаження послідовностей) виконується по символьне порівняння послідовності B із частиною послідовності A. Якщо в тій чи іншій частині результату посимвольного порівняння кількість підряд розташованих одиниць менше за d, то такі одиниці схемою виділення суттєвих збігів замінюються на 0. Реалізація схем порівняння та виділення суттєвих збігів наведена в [5]. Регістр згортки просторового коду працює наступним чином. Першопочатково всі його розряди скидаються в 0, а в подальшому в розряди регістру можуть записуватись лише 1. Наступні такти відрізняються від першого циклічним зсувом вмісту регістру 1 вліво на один символ. Загальна кількість тактів дорівнює n_a. Реалізація схем згортки просторового коду досліджена в [6-7]. В результаті на виході схеми згортки просторового коду формується двійковий код числа, який відображує кількісну характеристику запозичення або кількісну характеристику оригінальності піддослідної послідовності B .B Табл. 1 показано приклад функціонування пристрою у випадку коли B =<abcde>, n_b =5 , A = <acd4abc8e>, n_a = 9, d = 2.

Таблиця 1

	№ регістру згідно з рис.1	Вміст регістру								
1-й такт	2	a	b	c	d	e				
	1	a	c	d	4	a	b	c	8	e
	5	0	0	0	0	0				
2-й такт	2	a	b	c	d	e				
	1	c	d	4	a	b	c	8	e	a
	5	0	0	0	0	0				
3-й такт	2	a	b	c	d	e				
	1	d	4	a	b	c	8	e	a	c
	5	0	0	0	0	0				
4-й такт	2	a	b	c	d	e				
	1	4	a	b	c	8	e	a	c	d
	5	0	0	0	0	0				
5-й такт	2	a	b	c	d	e				
	1	a	b	c	8	e	a	c	d	4
	5	1	1	1	0	0				
6-й такт	2	a	b	c	d	e				
	1	b	c	8	e	a	c	d	4	a
	5	1	1	1	0	0				
7-й такт	2	a	b	c	d	e				
	1	c	8	e	a	c	d	4	a	b
	5	1	1	1	0	0				
8-й такт	2	a	b	c	d	e				
	1	8	e	a	c	d	4	a	b	c
	5	1	1	1	0	0				
9-й такт	2	a	b	c	d	e				
	1	e	a	c	d	4	a	b	c	8
	5	1	1	1	1	0				

Як видно із прикладу перший та останній символи піддослідної послідовності не вважаються сусідніми, що в деякій мірі спрощує схему виділення суттєвих збігів.

В розглянутому пристрої кількість звернень до пам'яті пропорційна $n_a + n_b$, а затрати часу на порівняння пропорційні n_a.

В випадку $n_a < n_b$ (рис.2) – регістр 1 циклічного зсуву для послідовності A з одного боку не може мати довжину меншу за довжину регістру 2, а з іншого боку результат роботи пристрою буде залежать від наповнення тими чи іншими даними тих розрядів регістру 1, які не містять символи послідовності A.

Загальновідомий підхід, який полягає в заповнені нулями тих розрядів регістру, що не використовуються, не дає стовідсоткової гарантії безпомилкового визначення запозичень. Для забезпечення такої гарантії можна скористуватись, наприклад, допоміжним регістром маски елементарних порівнянь. Такий регістр є регістром кільцевого зсуву з розрядністю, яка дорівнює кількості символів піддослідної послідовності.

На відміну від попереднього пристрою тут введено регістр маски 7, який є регістром циклічного зсуву та забезпечує ігнорування можливих випадкових збігів піддослідної послідовності з вмістом тих розрядів регістру 1, які не зайняті оригінальною послідовністю. В табл. 2 показано приклад функціонування пристрою в випадку коли B =<abcdcfahb>, n_b =9 , A = <cfbcdq>, n_a = 6, d = 2.

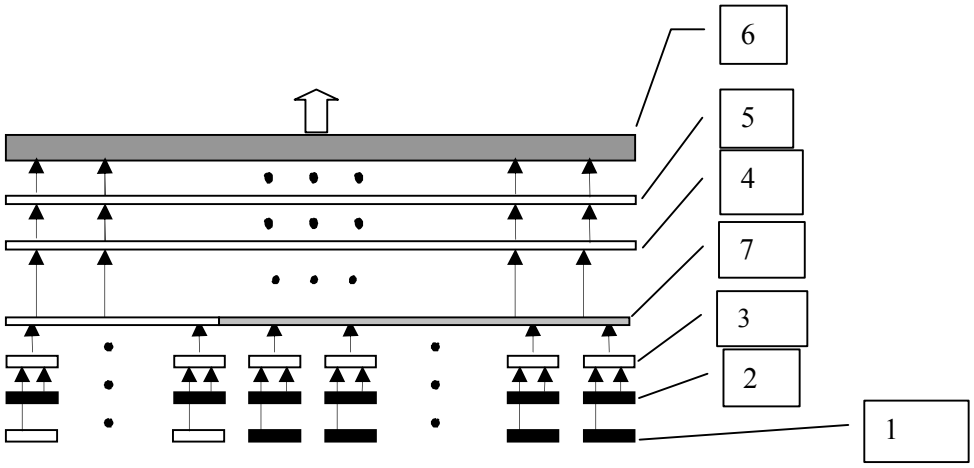

Рис. 2. Деталізація при $n_a < n_b$

В розглянутому пристрої кількість звернень до пам'яті пропорційна $n_a + n_b$, а затрати часу на порівняння пропорційні n_b. В випадку $n_a < n_b$ структура пристрою могла бути дещо інша – виконання порівнянь шляхом циклічного зсуву вмісту регістрів 2 та 5 та зменшенні кількості схем для виявлення елементарних збігів, скороченні довжини регістра 1 та відсутності регістра маски. В цьому випадку пристрій при $n_a < n_b$ буде суттєво відрізнятись від пристрою на рис.1. Оскільки при виявленні запозичень піддослідна послідовність порівнюється з багатьма послідовностями зразками різної довжини, то виникне необхідність реалізації та використання двох пристроїв.

Таблиця 2

	№ регістру згідно з рис.2	Вміст регістру								
1-й такт	2	a	b	c	d	c	f	a	h	b
	1				c	f	b	c	d	q
	7	0	0	0	1	1	1	1	1	1
	5	0	0	0	0	0	0	0	0	0
2-й такт	2	a	b	c	d	c	f	a	h	b
	1			c	f	b	c	d	q	
	7	0	0	1	1	1	1	1	1	0
	5	0	0	0	0	0	0	0	0	0
3-й такт	2	a	b	c	d	c	f	a	h	b
	1		c	f	b	c	d	q		
	7	0	1	1	1	1	1	1	0	0
	5	0	0	0	0	0	0	0	0	0
4-й такт	2	a	b	c	d	c	f	a	h	b
	1	c	f	b	c	d	q			
	7	1	1	1	1	1	1	0	0	0
	5	0	0	0	0	0	0	0	0	0
5-й такт	2	a	b	c	d	c	f	a	h	b
	1	f	b	c	d	q				c
	7	1	1	1	1	1	0	0	0	1
	5	0	1	1	1	0	0	0	0	0
6-й такт	2	a	b	c	d	c	f	a	h	b
	1	b	c	d	q				c	f
	7	1	1	1	1	0	0	0	1	1
	5	0	1	1	1	0	0	0	0	0
7-й такт	2	a	b	c	d	c	f	a	h	b
	1	c	d	q				c	f	b
	7	1	1	1	0	0	0	1	1	1
	5	0	1	1	1	0	0	0	0	0

8-й такт	2	a	b	c	d	c	f	a	h	b
	1	d	q				c	f	b	c
	7	1	1	0	0	0	1	1	1	1
	5	0	1	1	1	0	0	0	0	0
9-й такт	2	a	b	c	d	c	f	a	h	b
	1	q				c	f	b	c	d
	7	1				1	1	1	1	1
	5	0	1	1	1	1	1	0	0	0

На базі ж поданих на рис. 1 та рис. 2 пристроїв можна створити універсальний пристрій, який буде роботоздатним як при $n_a < n_b$, так і при $n_a \geq n_b$. Структура такого пристрою показана на рис. 3, де в складі пристрою безпосередньо використовується блок 8 завантаження оригінальної послідовності. Пристрій працює наступним чином. В кожному такті роботи пристрою із блоку 8 поступає черговий символ оригінальної послідовності та шляхом зсуву записується в регістр 1. Одночасно в регістр маски 7 шляхом зсуву записується 1. Якщо $n_a > n_b$, то перші символи оригінальної послідовності А виштовхуються із регістру 1, тобто, кільцевий зсув в даному пристрої не використовується. Після запису в регістр 1 останнього символу послідовності А виконується ще n_b - d додаткових тактів. В кожному з цих тактів відбувається зсув вмісту регістру 1 на один символ, а в регістр маски шляхом зсуву записується 0. В регістрі маски 7 кільцевий зсув також не використовується. Значення, якими заповнюються звільнені розряди регістру 1 при додаткових тактах, не суттєве. В іншому робота даного пристрою аналогічна роботі попереднього.

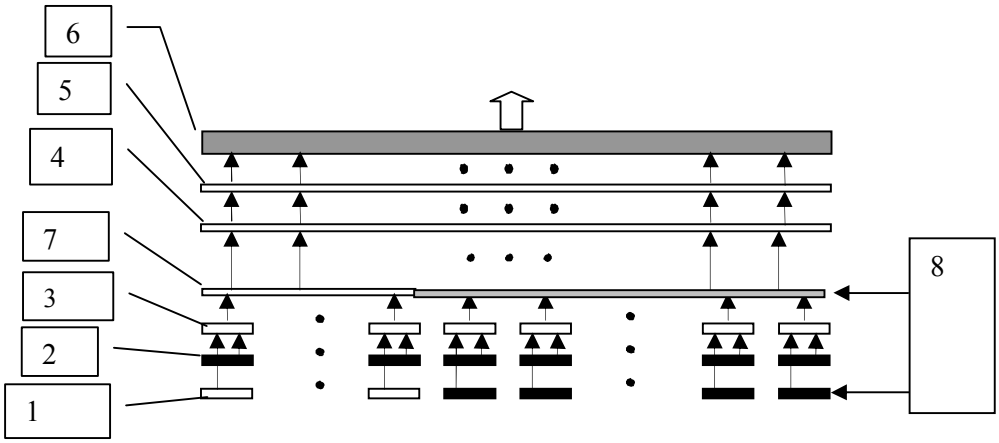

Рис. 3. Структура універсального пристрою

В таблицях 3 та 4 показано приклади функціонування пристрою при послідовностях А та В згідно табл. 1 та табл. 2.

Таблиця 3

	№ регістру згідно з рис.3	Вміст регістру				
1-й такт	2	a	b	c	d	e
	1					a
	7	0	0	0	0	1
	5	0	0	0	0	
2-й такт	2	a	b	c	d	e
	1				a	c
	7	0	0	0	1	1
	5	0	0	0	0	0
3-й такт	2	a	b	c	d	e
	1			a	c	d
	7	0	0	1	1	1

	5	0	0	0	0	0
4-й такт	2	a	b	c	d	e
	1		a	c	d	4
	7	0	1	1	1	1
	5	0	0	1	1	0
5-й такт	2	a	b	c	d	e
	1	a	c	d	4	a
	7	1	1	1	1	1
	5	0	0	1	1	0
6-й такт	2	a	b	c	d	e
	1	c	d	4	a	b
	7	1	1	1	1	1
	5	0	0	1	1	0
7-й такт	2	a	b	c	d	e
	1	d	4	a	b	c
	7	1	1	1	1	1
	5	0	0	1	1	0
8-й такт	2	a	b	c	d	e
	1	4	a	b	c	8
	7	1	1	1	1	1
	5	0	0	1	1	0
9-й такт	2	a	b	c	d	e
	1	a	b	c	8	e
	7	1	1	1	1	1
	5	1	1	1	1	0
10-й такт	2	a	b	c	d	e
	1	b	c	8	e	
	7	1	1	1	1	0
	5	1	1	1	1	0
11-й такт	2	a	b	c	d	e
	1	c	8	e		
	7	1	1	0	0	0
	5	1	1	1	1	0
12-й такт	2	a	b	c	d	e
	1	8	e			
	7	1	1	0	0	0
	5	1	1	1	1	0

Таблиця 4

	№ регістру згідно з рис.3	Вміст регістру								
1-й такт	2	a	b	c	d	c	f	a	h	b
	1									c
	7	0	0	0	0	0	0	0	0	1
	5	0	0	0	0	0	0	0	0	0
2-й такт	2	a	b	c	d	c	f	a	h	b
	1								c	f
	7	0	0	0	0	0	0	0	1	1
	5	0	0	0	0	0	0	0	0	0
3-й такт	2	a	b	c	d	c	f	a	h	b
	1							c	f	b

	7	0	0	0	0	0	0	1	1	1
	5	0	0	0	0	0	0	0	0	0
4-й такт	2	a	b	c	d	c	f	a	h	b
	1						c	f	b	c
	7	0	1	1	1	1	1	1	1	1
	5	0	0	0	0	0	0	0	0	0
5-й такт	2	a	b	c	d	c	f	a	h	b
	1					c	f	b	c	d
	7	0	0	0	0	1	1	1	1	1
	5	0	0	0	0	1	1	0	0	0
6-й такт	2	a	b	c	d	c	f	a	h	b
	1				c	f	b	c	d	q
	7	0	0	0	1	1	1	1	1	1
	5	0	0	0	0	1	1	0	0	0
7-й такт	2	a	b	c	d	c	f	a	h	b
	1			c	f	b	c	d	q	
	7	0	0	1	1	1	1	1	1	0
	5	0	0	0	0	1	1	0	0	0
8-й такт	2	a	b	c	d	c	f	a	h	b
	1		c	f	b	c	d	q		
	7	0	1	1	1	1	1	1	0	0
	5	0	1	1	1	0	0	0	0	0
9-й такт	2	a	b	c	d	c	f	a	h	b
	1	c	f	b	c	d	q			
	7	1	1	1	1	1	1	0	0	0
	5	0	0	0	0	1	1	0	0	0
10-й такт	2	a	b	c	d	c	f	a	h	b
	1	f	b	c	d	q				
	7	1	1	1	1	1	0	0	0	0
	5	0	1	1	1	1	1	0	0	0
11-й такт	2	a	b	c	d	c	f	a	h	b
	1	b	c	d	q					
	7	1	1	1	1	0	0	0	0	0
	5	0	1	1	1	1	1	0	0	0
12-й такт	2	a	b	c	d	c	f	a	h	b
	1	c	d	q						
	7	1	1	1	0	0	0	0	0	0
	5	0	1	1	1	1	1	0	0	0
13-й такт	2	a	b	c	d	c	f	a	h	b
	1	d	q							
	7	1	1	0	0	0	0	0	0	0
	5	0	1	1	1	1	1	0	0	0

Як видно з наведених прикладів, результат роботи універсального пристрою відповідає результатам роботи пристроїв, поданих на рис.1 та рис.2. В розглянутому пристрої кількість звернень до пам'яті пропорційна $n_a + n_b$, а затрати часу на порівняння пропорційні $n_a + n_b$- d. Враховуючи, що в поточному такті порівняння блок 8 може одночасно підготовлювати дані до наступного такту, то часові затрати на звернення до пам'яті для читання символів послідовності A можна виключити.

Універсальність поданого на рис.3 пристрою дозволяє одержувати інтегральну оцінку запозичень – обернену до оцінки оригінальності піддослідної послідовності. Дійсно, якщо після обробки першої послідовності із бази регістр 5 не обнуляти, а продовжувати обробку наступної послідовності із бази і т.д., то легко видно, що в результаті на виході пристрою буде сформована

інтегральна оцінка запозичень. Наприклад, одна частина піддослідної послідовності запозичена в одній послідовності із бази, інша – в другій.

Висновки

Застосування розглянутих пристроїв дозволяє суттєво зменшити верхню оцінку числа порівнянь до величини, пропорційної $n_a + n_b$ на відміну від $n_a \times n_b$. Це досягається за рахунок одночасних елементарних (посимвольних) порівнянь. Запропонований універсальний пристрій може бути використаний для одержання кількісної характеристики оригінальності піддослідної послідовності. Враховуючи високий рівень регулярності розглянутих структур, їх реалізація на ПЛІС шляхом створення параметричного (параметр - n_b) модуля на мові VHDL не є проблематично. Сучасні ПЛІС дозволяють практично реалізовувати модулі з досить великим значенням n_b, достатнім для контролю запозичень в студентських роботах незначного обсягу, таких, наприклад, як реферати, звіти з лабораторних робіт, домашні контрольні роботи.

З ростом можливостей ПЛІС перспективними є дослідження в напрямку ієрархічного адаптивного порівняння, суть якого полягає в використанні розглянутих пристроїв замість схем 3 – схем порівняння окремих символів. Тобто, елементарним порівнянням вважати адаптивне порівняння окремих складових інформаційного об'єкту, наприклад, речень, абзаців або параграфів в текстових даних.

1. http://www.antiplagiat.ru
2. http://searchinform.com
3. http://www.plagiarism-detector.com
4. В.П. Тарасенко, А.Ю. Михайлюк, О.К. Тесленко, О.С. Осипов Автоматизація оцінки оригінальності інформації. Наукові записки українського наково-дослідного інституту зв'язку.- №1.- 2007.
5. Клятченко Я.М., Тарасенко В.П., Тесленко О.К. Ефективність ПЛІС - реалізацій адаптивного порівняння послідовностей символів // Науковий вісник Чернівецького університету: Збірник наук. праць.- Вип. 446: Комп'ютерні системи та компоненти. – Чернівці: ЧНУ, 2009.-С. 23-28.
6. В.П.Тарасенко, О.В.Тарасенко-Клятченко, О.К.Тесленко. Оцінка складності оптимізованих пірамідальних структур для додавання розрядних добутків // Інформаційні технології та комп'ютерна інженерія.- 2007.- №3(10).- С.19-24
7. В.П.Тарасенко, О.В.Тарасенко-Клятченко, О.К.Тесленко. Оптимізація структур пристроїв згортки рівноважних однорозрядних операндів // Інформаційні технології та комп'ютерна інженерія.- 2008, №3(13).- С.14-17.