УДК 004.272

І.Д. Яковлєва
Чернівецький національний університет імені Юрія Федьковича

ОЦІНКА ВАРІАНТІВ СИНТЕЗУ ПАРАЛЕЛЬНИХ ОБЧИСЛЮВАЛЬНИХ ПРИСТРОЇВ СОРТУВАННЯ

Розглянуто різні варіанти синтезу паралельних обчислювальних пристроїв сортування та їхнє проектування від програмного опису алгоритму до апаратної реалізації з можливістю зміни ширини паралельної форми.

Various options for the synthesis of parallel computing sorting devices and their design flow from software descriptions of the algorithm to hardware implementation with the possibility of changing the width of the parallel form are considered.

Вступ. Висока продуктивність обробки даних у сучасних комп'ютерах досягається завдяки використанню просторового і часового паралелізму. Особливо цікавою в цьому сенсі є концепція підвищення продуктивності комп'ютерного пристрою шляхом наближення його структури до структури виконуваного алгоритму [1]. Перетворення алгоритму в його апаратну модель відбувається шляхом повного апаратного відображення потокового графа виконуваного алгоритму (ПГА) операційними пристроями (ОП), які виконують функціональні оператори (ФО) алгоритму і з'єднані між собою відповідно до графа алгоритму [2]. Пристрої, структура яких адаптована до ПГА, мають такі переваги: операцію із всіма операндами можна виконувати незалежно від стану інших операцій (не потрібна синхронізація); обмін даними між операціями чітко визначений; управління операціями здійснюється за допомогою передавання даних між ними; алгоритм виконується за один такт, що надає багаторазове прискорення виконання конкретної задачі. Але виникає дуже багато питань, що стосуються, у першу чергу того, як будувати й змінювати графи алгоритмів для можливості їх апаратної реалізації [1, 2, 3, 4].

Розглянуті різні варіанти синтезу паралельних обчислювальних пристроїв сортування та їхнє проектування від програмного опису алгоритму до апаратної реалізації з можливістю зміни ширини паралельної форми.

1. Огляд літератури. Частка сортування серед операцій, виконуваних комп'ютером, є доволі високою, оскільки ця задача є однією з типових проблем обробки даних, і, звичайно, розуміється як задача розміщення елементів невпорядкованого набору значень $\overline{X} = \{x_1, x_2, ..., x_N\}$ в порядку монотонного зростання або убування $\overline{X} \approx \overline{Y} = \{(y_1, y_2, ..., y_N) : y_1 \le y_2 \le ... \le y_N\}$ [5]. Крім того, на основі ПГА сортування будуються комутуючі мережі, які забезпечують здатність до обміну даними між компонентами багатопроцесорної комп'ютерної системи [2]. Також, алгоритми сортування – зручний засіб для визначення переваг тієї або іншої моделі та для їх порівняння між собою.

Обчислювальна трудомісткість процедури впорядкування є доволі високою. Так, для деяких відомих простих методів (бульбашкове сортування, сортування включенням тощо) кількість необхідних операцій визначається квадратичною залежністю від числа даних, що впорядковують $t_s \approx N^2$ [5]. Для ефективніших алгоритмів (сортування злиттям, сортування Шелла тощо) трудомісткість визначається величиною $t_s \approx N \log_2 N$ [5]. Паралельне виконання операцій алгоритму сортування декількома операційними пристроями одночасно значно прискорює час виконання алгоритму. Серед алгоритмів сортування паралельне виконання операцій можна виконувати для алгоритмів, в яких послідовність виконуваних операцій залежить тільки від числа вхідних даних *N* і не залежить від значень їхніх ключів (неадаптивні алгоритми) [6]. Серед алгоритмів сортування неадаптивними є: алгоритм сортування за

методом “парно-непарної” перестановки [7], алгоритм сортування модифікованим методом “бульбашки” [6], алгоритм сортування за методом Бетчера [5]. Але і під час такого проектування з’являються недоліки, які із ускладненням проектів стають доволі значними, наприклад, досить часто приходиться виконувати вручну роботу з розпаралелювання алгоритмів або зміни ширини паралельної форми. Така робота вимагає часу, високої кваліфікації та не гарантує збереження простого розв’язку і, тим самим, призводить до того, що на деякому етапі ускладнення, проект стає нерентабельним. Це змушує шукати нові шляхи прискорення проектування.

Постановка задачі. Отже, оскільки задача сортування є однією з типових проблем обробки даних та актуальною задачею є пошук нових шляхів прискорення проектування граф-алгоритмічних пристроїв із ПГА, то постало завдання автоматизації процесу переходу від програмного опису алгоритмів сортування до їх апаратної реалізації з можливістю зміни паралельної форми алгоритму.

2. Визначення характеристик ПГА. Описані вище методи паралельного сортування засновані на застосуванні однієї й тієї самої базової операції “порівняти й переставити”, яка полягає в порівнянні тієї або іншої пари ключів із набору даних для сортування та перестановки цих значень, якщо їхній порядок не відповідає умовам сортування. Ці методи відрізняються між собою лише порядком порівняння пар, а базова операція “порівняти й переставити” однакова для таких алгоритмів. Отримані ПГА даних алгоритмів для сортування 16 вхідних значень зображені на рис. 1.

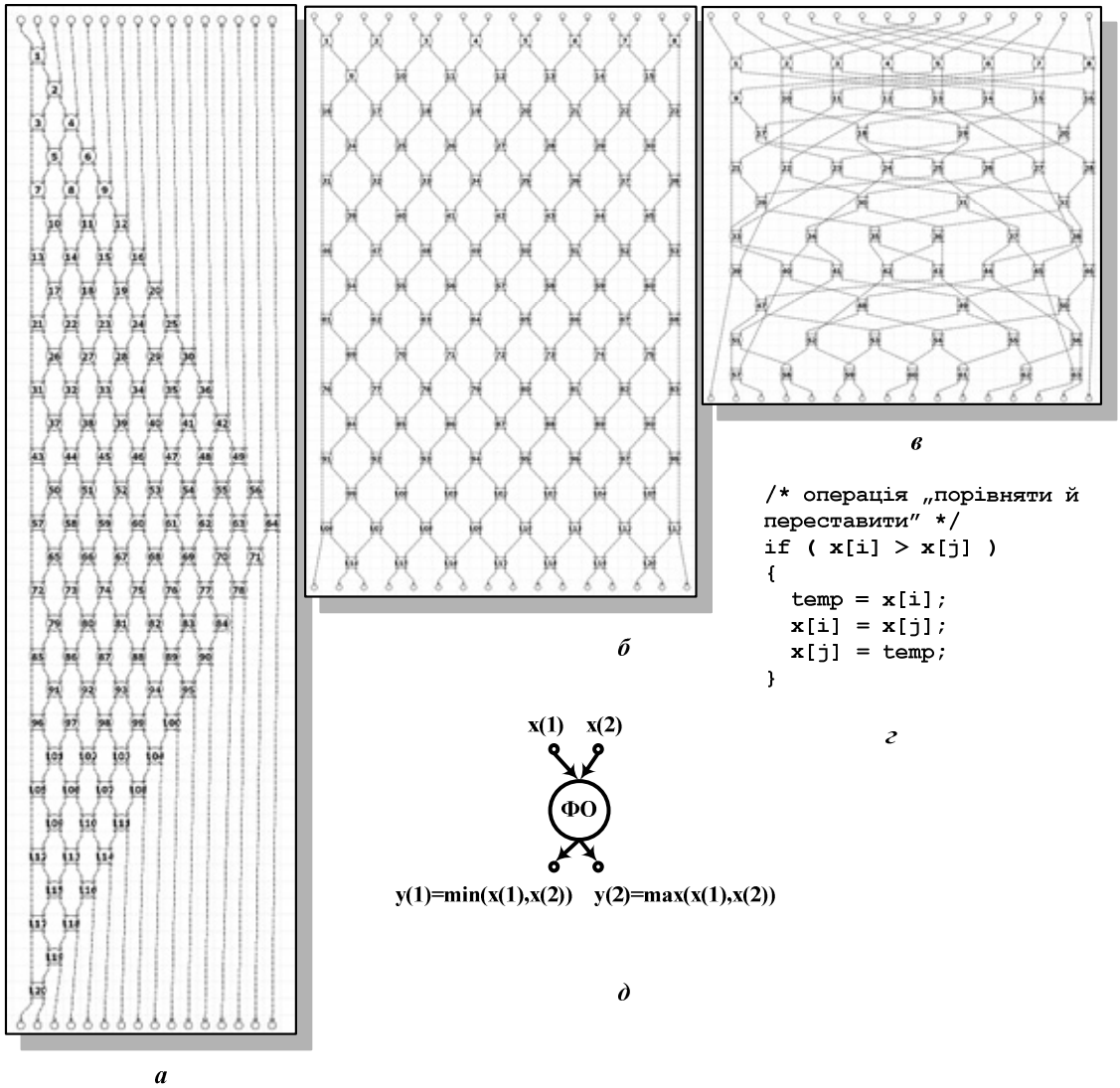

Рис. 1. Алгоритми сортування 16 значень: а – ПГА сортування модифікованим методом “бульбашки”; б – ПГА сортування “парно-непарної” перестановки; в – ПГА сортування Бетчера; г, д – базова операція “порівняти й переставити”

Для однакової кількості вхідних значень N дані алгоритми мають однакову ширину ПГА – $\left\lceil \frac{N}{2} \right\rceil$, але виконують різну кількість операцій „порівняти й переставити” та мають різну кількість ярусів. Загальна кількість ФО ПГА дорівнює загальній кількості операцій „порівняти й переставити”.

Для алгоритмів сортування за методом “парно-непарної” перестановки та модифікованим методом “бульбашки” для *N* вхідних значень кількість ФО дорівнює $N(N-1)/2$ [2], а для алгоритму сортування за методом Бетчера – $0{,}48N\ln^2 N$, що підтверджено результатами моделювання. Висота ПГА для цих алгоритмів різна і є найбільшою для модифікованого алгоритму сортування за методом “бульбашки” – $2N-3$ [2]. Для алгоритму сортування за методом “парно-непарної” перестановки висота ПГА дорівнює кількості вхідних значень N, і для алгоритму сортування за методом Бетчера є найменшою і дорівнює $\frac{1}{2}\lceil\log_2 N\rceil(\lceil\log_2 N\rceil+1)$ [5].

Оскільки всі алгоритми виконують одну операцію „порівняти й переставити”, вважатимемо, що часова затримка всіх ФО однакова і дорівнює одиниці. Апаратна складність цих обчислювальних пристроїв дорівнюватиме кількості ФО ПГА, а часова затримка – кількості ярусів ПГА.

Проаналізувавши табл. 1, зроблено висновок, що з усіх алгоритмів алгоритм сортування за методом Бетчера має найкращі часові властивості за найменших апаратних затрат. Оскільки ПГА сортування за методом Бетчера для N вхідних даних містить $\frac{1}{2}\lceil\log_2 N\rceil(\lceil\log_2 N\rceil+1)$ ярусів [5], то час виконання алгоритму становитиме $\Theta(\frac{1}{2}\lceil\log_2 N\rceil(\lceil\log_2 N\rceil+1))$.

Таблиця 1

Обчислювальні характеристики алгоритмів сортування залежно від кількості вхідних значень

Алгоритм сортування	**Кількість операцій, *w***	**Кількість ярусів, *t***
модифікований “бульбашки”	$N(N-1)/2$	$2N-3$
“парно-непарної” перестановки	$N(N-1)/2$	N
за методом Бетчера	$0{,}48N\ln^2 N$	$\frac{1}{2}\lceil\log_2 N\rceil(\lceil\log_2 N\rceil+1)$

3. Зменшення ширини ПГА. Відомі ситуації, коли кількість операторів в одному ярусі повинна бути обмежена і менша за ширину ПГА. У таких випадках, за рахунок того, що на одних ярусах ПГА ФО розміщено більше, а на інших – менше, можна переміщувати ФО з яруса на ярус, зменшуючи ширину ПГА і, досягаючи при тому, наприклад, максимального завантаження процесорів, на яких виконується алгоритм. Такі графи задають еквівалентні алгоритми, і насправді усі вони збігаються. Відрізняються вони між собою лише топологією. Змінюючи ширину ПГА, автоматично виконується також перевірка на зменшення висоти ПГА за правилом побудови ПГА [1].

Дослідження зміни ширини ПГА показало, що при фіксованій кількості вхідних значень, із зменшенням ширини ПГА збільшується висота ПГА в межах від мінімального значення висоти ПГА, що дорівнює кількості ярусів ПГА до максимального, що дорівнює кількості ФО ПГА. Це показало, що при змінюванні ступеня розпаралелювання алгоритмів складність їх апаратної реалізації не змінюється, а змінюється тільки час виконання алгоритмів (табл. 2).

Таблиця 2

Обчислювальні характеристики алгоритмів сортування залежно від кількості вхідних значень та ширини паралельної форми алгоритму

Алгоритм сортування	Кількість операцій, *w*	Кількість ярусів, *t*
модифікований “бульбашки”	$N(N-1)/2$	$2N-3 \le t \le N(N-1)/2$
“парно-непарної” перестановки	$N(N-1)/2$	$N \le t \le N(N-1)/2$
за методом Бетчера	$0{,}48N\ln^2 N$	$\frac{1}{2}\lceil\log_2 N\rceil(\lceil\log_2 N\rceil+1) \le t \le 0{,}48N\ln^2 N$

Для 16 вхідних значень залежність висоти ПГА алгоритмів сортування від ширини розпаралелювання алгоритмів зображена на рис. 2. При розпаралелюванні алгоритмів висота ПГА, а значить і час виконання алгоритмів, гіперболічно зменшуються, досягаючи свого мінімального часу виконання в точці максимального розпаралелювання. Сортування за методом Бетчера при послідовному виконанні операцій для 16 вхідних значень виконується в два рази швидше, ніж іншими описаними алгоритмами, а при розпаралелюванні швидкість виконання порівняно із модифікованим методом “бульбашки” час виконання алгоритму зменшується майже в три рази, а порівняно із методом “парно-непарної” перестановки в 1,6 раза.

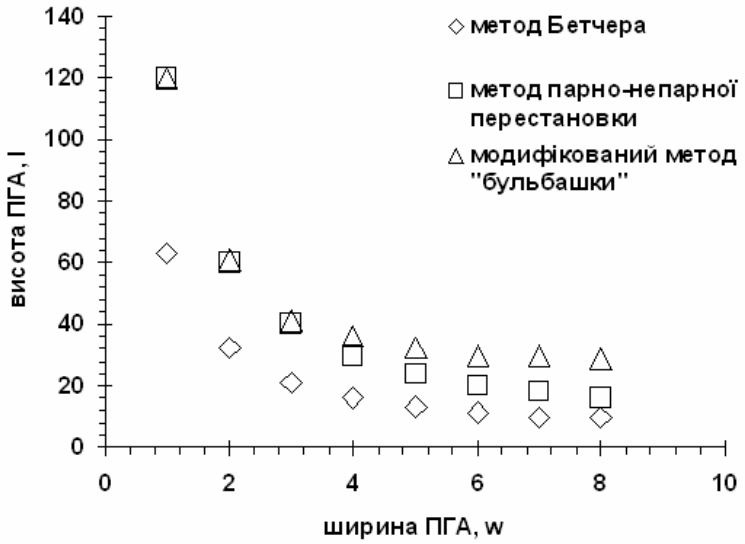

Рис. 2. Залежність висоти ПГА алгоритмів сортування від ширини розпаралелювання алгоритмів для N=16

Порівняно із послідовним сортуванням паралельне сортування 16 вхідних значень виконується швидше: для модифікованого методу “бульбашки” в 4,1 раза, для методу “парно-непарної” перестановки в 7,5 раза і для методу Бетчера в 6,3 і з збільшенням кількості вхідних значень гіперболічно зростає.

Отже, автоматична зміна ширини дозволяє використовувати всі можливості розпаралелювання алгоритму і змінювати динамічно ступінь розпаралелювання від максимальної до повної відсутності розпаралелювання.

4. Спрощення ПГА в залежності від правила виведення. Відомі також задачі, для яких необхідно отримати не весь вектор вихідних даних, який визначається заданим алгоритмом, а лише діапазон вектора результату. Оскільки, як результат, необхідний тільки певний діапазон вихідного

вектора даних, то очевидно, що не всі ФО ПГА беруть участь в його формуванні і необхідно виконувати тільки ті ФО, що безпосередньо відповідають за отримання вибраного діапазону вихідних значень, і від яких він залежить.

Кількість варіантів спрощення ПГА залежно від правила виведення [8] дорівнює 2^N, де N – величина вектора результату. Тут розглянутий варіант для N=16, і діапазон виведення результату зменшується на 1 від максимального до мінімального значення. Ця залежність кількості ФО ПГА від зменшення величини вектора результату зображена на рис. 3, а.

Спрощений алгоритм модифікованої “бульбашки” в деяких точках (15 по осі абсцис рис. 3, а) показав кращі результати за алгоритмом Бетчера: за необхідності отримання тільки останніх двох значень спрощеним модифікованим методом „бульбашки” необхідно виконати лише 29 операцій, у той час, як за спрощеним алгоритмом Бетчера – 43, а для пошуку останнього елемента обидва ПГА мають тільки 15 вершин. Але важливим є не тільки кількість операцій, а й кількість ярусів ПГА, на яких розміщені ФО.

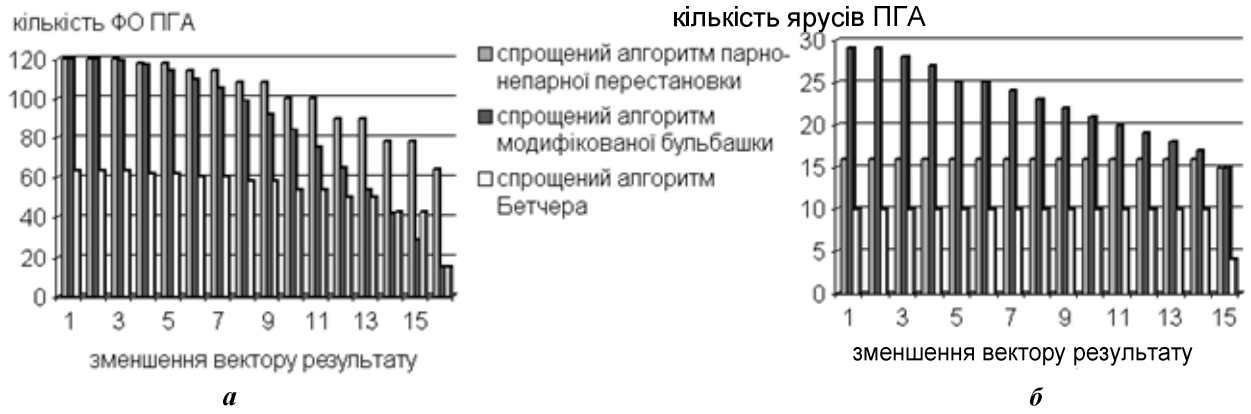

Рис. 3. Залежність кількості ФО ПГА(а) та кількості ярусів (б) спрощених алгоритмів сортування за правилом виведення

Для 16 вхідних значень залежність кількості ярусів ПГА від зменшення величини вектора результату зображена на рис. 3, б.

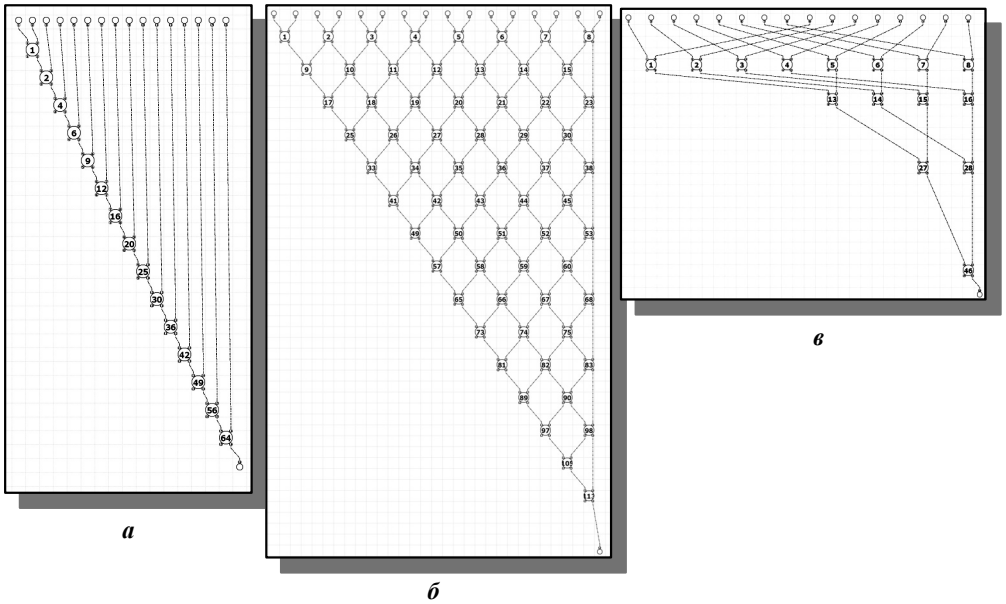

Рис.4. Графи спрощених алгоритмів за правилом виведення знаходження максимального елемента із 16 значень: а – ПГА сортування модифікованим методом “бульбашки”; б – ПГА сортування “парно-непарної” перестановки; в – ПГА сортування Бетчера

З рисунка очевидно, що для пошуку двох останніх значень вектора результату спрощеним модифікованим методом „бульбашки” необхідно виконати лише 29 операцій за 17 ярусів, а спрощеним алгоритмом Бетчера 43 ФО за 10. А для пошуку останнього значення результуючого вектора перший алгоритм виконує 15 операцій на 15 ярусах, а другий (алгоритм Бетчера) таку ж кількість на 4 ярусах (рис. 4).

Отже, визначити кращий із алгоритмів можна, аналізуючи тільки увесь комплекс їх параметрів.

5. Перетворення графічного подання алгоритму в його апаратну модель. Пристрій сортування впорядковує N вхідних елементів невпорядкованого набору значень $\overline{X} = \{x_1, x_2, ..., x_N\}$ в порядку монотонного зростання $\overline{X} \approx \overline{Y} = \{(y_1, y_2, ..., y_N) : y_1 \le y_2 \le ... \le y_N\}$. Структура пристрою точно відображає обраний один із трьох ПГА сортування за методом “парно-непарної” перестановки, модифікованим методом “бульбашки” або за методом Бетчера із накладеними на нього обмеженнями (спрощення за правилом виведення, зміна ширини паралельної форми). Крім алгоритму сортування та кількості вхідних значень N, для перетворення графічного подання обраного алгоритму в його апаратну модель, проектувальник може обрати тип пристрою (синхронний (табл. 3) або асинхронний (табл. 4)), формат подання даних натуральних чисел і для нього вибрати ширину шини даних (width) або формат IEEE-754.

Таблиця 3

Опис інтерфейсу синхронних апаратних моделей алгоритмів сортування

Вивід (натуральні числа/формат IEEE-754)	**Опис**
CLK	Clock
RST	Asynchronous Reset
X_N[width – 0] / X_N[32 – 0]	Input
Y_N[width – 0] / Y_N [32 – 0]	Output

Таблиця 4

Опис інтерфейсу асинхронних апаратних моделей алгоритмів сортування

Вивід (натуральні числа/формат IEEE-754)	**Опис**
F	Flag ready input
RST	Reset
X_N[width – 0] /X_N [32 – 0]	Input
Y_N[width – 0] /Y_N [32 – 0]	Output
R	Output flag ready result

Для перетворення алгоритму сортування в його апаратну модель вибраний алгоритмом Бетчера синхронного сортування 16 цілих чисел (рис. 5).

Для верифікації апаратної моделі виконано контрольні розрахунки. Часова діаграма роботи обчислювального пристрою сортування за методом Бетчера засобами Waveform Editor наведена на рис. 5, в, а процес сортування зображений на рис. 5, а: 63 операції “порівняти й переставити” для сортування 16 вхідних значень виконуються за десять кроків.

Висновки

1. При розпаралелюванні алгоритмів сортування висота ПГА, а значить і час виконання алгоритмів, гіперболічно зменшуються, досягаючи свого мінімального часу виконання в точці максимального розпаралелювання.

2. При фіксованій кількості вхідних значень, із зменшенням ширини ПГА сортування збільшується висота ПГА в межах від мінімального значення висоти ПГА, що дорівнює кількості ярусів ПГА до максимального, і дорівнює кількості ФО ПГА.

3. Змінюючи ступінь розпаралелювання алгоритмів, складність їх апаратної реалізації не змінюється, а змінюється тільки час виконання алгоритмів.

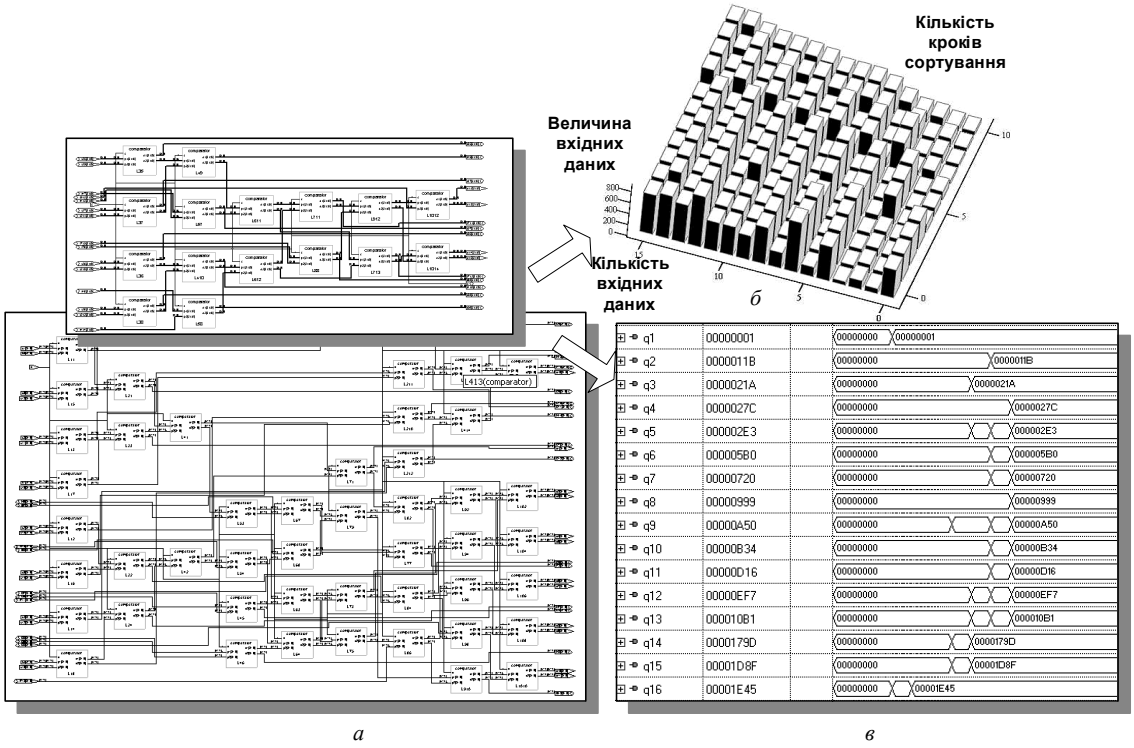

Рис. 5. Сортування 16 значень за методом Бетчера: а – VHDL-модель;
б – процес сортування процес сортування; в –часова діаграма

4. Автоматична зміна ширини ПГА сортування дозволяє використовувати всі можливості розпаралелювання алгоритму і змінювати динамічно ступінь розпаралелювання від максимальної до повної відсутності розпаралелювання.

5. Спрощення ПГА залежно від правила виведення надає можливість зменшувати апаратну складність пристроїв сортування.

6. З порівнюваних алгоритмів апаратна реалізація алгоритму сортування за методом Бетчера має найкращі часові властивості при найменших апаратних затратах.

7. Програмний модуль реалізований в середовищі Delphi 7.0, VHDL 6.3 та Synplify і є частиною проекту САПР спеціалізованих алгоритмічних операційних пристроїв.

1. Мельник А.О. Спеціалізовані комп'ютерні системи реального часу. – Львів: Нац. ун-т "Львівська політехніка", 2002. – 60 с. 2. Мельник А.О. Архітектура комп'ютера. – Луцьк: Волинська обл. друк, 2008. – 470 с. 3. Воеводин В.В. Вычислительная математика и структура алгоритмов. – М.: Изд-во МГУ, 2006. – 112 с. 4. Анісімов А.В., Кулябко П.П., Терещенко В.М. Паралельні алгоритми в дослідженнях неперервних систем. – К.: РВЦ Київський університет, 1999. – 55 с. 5. Кнут Д. Искусство программирования для ЭВМ. Т.3: Пер. с англ. – М.: Мир, 1978. – 841 с. 6. Т. Кормен, Ч. Лейзерсон, Р. Ривест. Алгоритмы: построение и анализ: Пер. с англ. / Под ред. А. Шеня. – М.: МЦНМО: БИНОМ. Лаборатория знаний, 2004. – 960 с. 7. Бройнль Т. Паралельне програмування: Пер. з нім. – К.:Вища шк., 1997. – 358 с. 8. A. Melnyk, V. Melnyk. Parameters of Algorithm // Proceedings of Conference "Modern Problems in Electrical Engineering and Informatics", Politechnika Świętokrzyska, Ameliówka, Poland, 17–18.06.2005. – P. 115–121.