выпрямителях с помощью коммутационных функций // Электричество. – 1973. –№ 4. – С. 21–26. 6. Новский В.А., Жарский Б.К., Козлов А.В., Бойко П.С. Анализ режимов работы и энергетических показателей однофазных сетевых выпрямителей // Техн. електродинаміка. Тематичний випуск. Силова електроніка та енергоефективність. – 2009. Ч. 4.– С. 15 – 18. 7. Плахтына Е.Г. Математическое моделирование электромашинно-вентильных систем. — Львов: Вища школа, 1986. — 164 с.8. Самотий В.В. Математичне моделювання стаціонарних процесів електромагнетних пристроїв систем керування. – Львів: Фенікс, 1997. – 170 с. 9. Чабан В.И., Самотый В.В. Применение экстраполяционного метода в задачах ускоренного поиска стационарных процессов электромагнитных устройств // Изв. вузов. Электромеханика. – 1987.– № 8. – С. 13 – 17. 10. Эйприлл Т., Трик Т. Анализ стационарного режима нелинейных цепей с периодическими входными сигналами //В кн.: Автоматизация в проектировании.– М.: Мир, 1972 .– С. 148 – 155.

УДК 681.325.5-181.4

В. Глухов, Р. Еліас
Національний університет "Львівська політехніка",
кафедра електронно-обчислювальних машин

ПЕРЕТИННА ДЕКОМПОЗИЦІЯ ЦИФРОВИХ АВТОМАТІВ

Визначено найкращий спосіб перетинної декомпозиції цифрових автоматів.

In this article the best way of finite state machines overlapping decomposition is described.

Вступ

Аналогія між використанням підпрограм та багаторівневих структур дає змогу адаптувати відомі програмні рішення для проектування багаторівневих спеціалізованих процесорів.

Пропонується процедурне абстрагування та декомпозицію застосувати для проектування багаторівневих спеціалізованих обчислювачів, а саме, їхніх керуючих цифрових автоматів.

У роботі визначається найкращий спосіб перетинної декомпозиції цифрових автоматів. Для цього автомат представляється як мікропрограмний. Співвідношення об'ємів пам'яті мікрокоманд використовується як критерій перетинної декомпозиції.

Постановка проблеми

Для апаратної реалізації гарантоздатних систем використовується апаратна аналогія процедурної абстракції – багаторівневі системи. Для керування багаторівневими ієрархічними структурами потрібно мати, відповідно, ієрархічну систему керуючих цифрових автоматів. Актуальною є задача декомпозиції одного автомата, який реалізує загальний алгоритм розв'язання задачі на декілька ієрархічно зв'язаних автоматів, кожний з яких керує окремим рівнем багаторівневої системи. У статті розглядається розв'язок задачі знаходження найкращого варіанта перетинної декомпозиції цифрового автомата на два ієрархічно зв'язані між собою автомати.

Аналіз основних досліджень та публікацій

Аналогія між використанням підпрограм та багаторівневих структур дає змогу адаптувати відомі програмні рішення для проектування багаторівневих спеціалізованих процесорів. Для проектувальників апаратних систем дуже актуальною є проблема "Hardware-Software Codesign" (одночасне розроблення апаратного і програмного забезпечення) [1, 2], для вирішення якої можна використати методи об'єктно-орієнтованого програмування (ООП) [3].

Перевагами процедурної абстракції у програмуванні є модифікованість і локальність, відомі методи їх забезпечення [6]. Одночасно, для побудови гарантоздатних систем [4, 5] і модифікованість, і локальність є одними з визначальних рис. Досвід програмування можна використати для проектування гарантоздатних систем.

Процедури (відповідно, і багаторівневі структури), як і інші види абстракцій необхідно мінімізовувати. Одним з методів мінімізації є метод функціональної декомпозиції, який також широко використовується при мінімізації функцій алгебри логіки [7]. Метод функціональної декомпозиції належить до загальних методів мінімізації і не гарантує знаходження найкращого рішення. Більше того, при неграмотному використанні декомпозиція може принести масу неприємностей. Декомпозицію поділяють на просту розділову та перетинну [8, 9].

Аналогом програмних процедур можуть бути пристрої – цифрові автомати. Якщо підходити до цифрового автомату як до мікропрограмного, тобто розглядати його комбінаційну частину як ПЗП, то за показник якості мінімізації (декомпозиції) можна обрати зміну загального об'єму ПЗП усіх автоматів [10]. Результати застосування простої розділової декомпозиції для декомпозиції цифрових автоматів оцінено у [11]].

Відомі абстрактні і структурні автомати. Абстрактному алгоритму відповідає абстрактний автомат, логічно структурному автомату поставити у відповідність структурний алгоритм [12], який реалізується на цьому автоматі і враховує особливості структури цього автомату.

Теорія програмування рекомендує використовувати процедури і функції за такими загальними принципами [13, 14]:

процедура (функція) не повинна містити більше ніж 60 рядків тексту;

процедура (функція) повинна викликатися не менше двох разів (більше одного разу);

функція повинна робити тільки одну справу;

мати дуже багато рівнів абстракції або інкапсуляції так само погано, як і дуже мало;

код, що використовується більше одного разу, має бути поміщений у функцію;

функція повинна мати лише одну точку виходу.

Відповідно, ці самі рекомендації можна застосувати і для цифрових автоматів, які реалізують функції нижчих рівнів у складі багаторівневих гарантоздатних систем. Тоді текстом можна вважати опис мовою VHDL (наприклад).

Одним із способів представлення алгоритмів роботи автоматів є граф – множина V вершин та набір E невпорядкованих і впорядкованих пар вершин [15]. Сучасними засобами автоматизованого проектування можна автоматично генерувати описи мовою VHDL з описів у вигляді графів.

До задач, які вимагають для свого розв'язання використання багаторівневих апаратних ієрархічних структур [16], можна віднести задачу утворення і перевірки цифрового підпису у гарантоздатних бортових системах [17]. Оцінка апаратних витрат на реалізацію таких структур наведена у [18].

Цілі статті

Метою роботи є визначення умов найкращої перетинної декомпозиції одного цифрового автомату на два ієрархічно зв'язані автомати.

Оцінювання структурних алгоритмів і структурних автоматів

Рекомендації з використання процедур і функцій можна застосувати і для цифрових автоматів, які реалізують функції нижчих рівнів у складі багаторівневих системах.

Одному абстрактному алгоритму можуть відповідати декілька структурних алгоритмів, і, відповідно, декілька структурних автоматів. Першочергова задача проектування спеціалізованих комп'ютерів полягає в синтезі з відомого абстрактного алгоритму структурних алгоритмів і тісно пов'язаних з ними структурних автоматів. Саме звідси починається проблема "Hardware-Software Codesign". Вирішити цю проблему можна, застосовуючи принципи об'єктно-орієнтованого програмування – спочатку описати об'єкт (структуру), а потім написати програму його роботи (структурний алгоритм). Ідеалом є автоматична зміна програми за будь-якої зміни опису об'єкта.

Другою важливою задачею є вибір з множини структурних автоматів одного, найкращого за обраними критеріями.

Перехід від абстрактного алгоритму до структурного, а потім до структурного автомату дає змогу оцінити алгоритми у структурних одиницях (кількості вентилів, кількості зв'язків, об'ємі пам'яті, кількості тактів, потрібних для виконання якоїсь операції тощо), які є найвагомішими для практичної реалізації автомату і які практично використовуються як критерії для визначення найкращого варіанта реалізації.

Декомпозиція графів і автоматів

Одним із способів представлення алгоритмів є граф – множина *V* вершин та набір *E* невпорядкованих і впорядкованих пар вершин [15]. Результат декомпозиції одного автомата на два, представлених за допомогою графів, ілюструє рис. 1. Взаємодія здійснюється за принципом «запит–відповідь». Алгоритм *A* починає працювати після появи сигналу Старт і запускає на виконання алгоритм *B*, для чого формує сигнал Старт *B*. Після цього алгоритм *A* очікує, доки алгоритм *B* не сформує сигнал Готов *B*, після чого продовжує свою роботу.

Алгоритм, представлений графами рис. 1, можна реалізувати як за допомогою програми і підпрограми, так і за допомогою двох цифрових автоматів *A* та *B* відповідно, перший з яких запускає роботу другого.

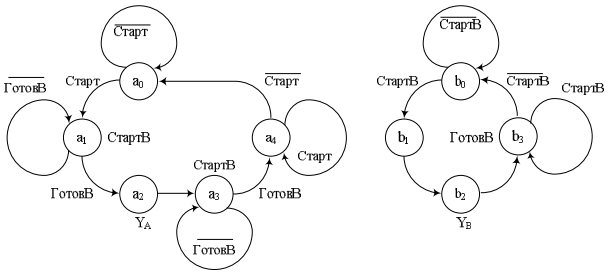

Рис. 1. Взаємодія автоматів

Найкращий варіант перетинної декомпозиції автоматів

Ефект від декомпозиції цифрових автоматів можна побачити, порівнюючи цифрові пристрої: автомата, який не підлягав декомпозиції (рис. 2, з кількістю вході *m*, кількістю виходів *n*, кількістю зворотних зв'язків *s*), і результату його перетинної декомпозиції на два автомати *A* та *B* (рис. 3, з кількістю входів $m_A+m_C+1=m_{AC}+1$ та $m_B+m_C+1==m_{BC}+1$, кількістю виходів n_A та n_B, кількістю зворотних зв'язків s_A та s_B, відповідно). При цьому для методу перетинної декомпозиції m_A – кількість входів, що належать тільки автомату *A*, m_B – кількість входів, що належать тільки автомату *B*, m_C – кількість входів, що належать обидвох автоматам *A* та *B*, $m=m_A+m_B+m_C$, $n=n_A+n_B$, $s\text{-}1=s_A=s_B$ (для великої кількості станів кількість станів кожного з автоматів приблизно вдвічі менша порівняно з початковим автоматом, відповідно, кількість зворотних зв'язків менша на 1).

На рис. 2 та рис. 3 позначено:

$X = \{x_i\}$ - $i \in \overline{1,m}$ – множина вхідних сигналів автомата (вхідний алфавіт автомата);

$S = \{s_j\}$ - $j \in \overline{1,s}$ – множина станів автомата (алфавіт станів автомата);

$Y = \{y_k\}$ - $k \in \overline{1,n}$ – множина вихідних сигналів автомата (вихідний алфавіт автомата);

φ – функція переходів автомата, що задає відображення $(X \times S) \to S$;

λ – функція виходів автомата, що задає відображення $(X \times S) \to Y$,

КСх – комбінаційна схема;

ПА – пам'ять автомата.

З результатів аналізу складності таких ієрархічно зв'язаних автоматів, як процесор і сопроцесор [18], випливає, що декомпозицію треба виконувати на дві приблизно однакові за апаратними витратами частини, тобто, можна запропонувати, що повинно виконуватися $m_A \approx m_B$, $n_A \approx n_B$, $s_A \approx s_B$ (в ідеалі у

наведених рівняннях повинен стояти знак («дорівнює»). Якщо прийняти $m_A=m_B$, тоді $m_A=m_B=(m-m_C)/2$ і $m_{AC}=m_A+m_C=(m-m_C)/2+m_C=(m+m_C)/2$. Аналогічно $m_{BC}=(m+m_C)/2$.

Якщо підходити до цифрового автомата як до мікропрограмного, тобто розглядати його комбінаційну частину як ПЗП, то показником складності можна вважати об'єм цього ПЗП [10, 11], який дорівнює $V=2^{m+s}(n+s)$, де m – кількість входів, n – кількість виходів, s – кількість зворотних зв'язків.

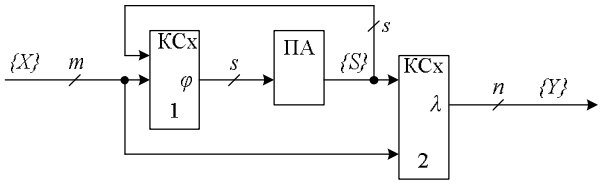

Рис. 2. Абстрактний цифровий автомат

Об'єм ПЗП автомата (рис. 2) $V_1=2^{s+m}(s+n)$.

Об'єм ПЗП кожного з автоматів (рис. 1) $V_2=2^{s-1+(m+m_C)/2+z}(s-1+\frac{n}{2}+z)$, z – кількість додаткових сигналів, необхідних для взаємодії автоматів (сигнали типу Старт – Готов). Для найпростішого варіанта z=1, тобто, кожний автомат має один додатковий вхід (Готов) і один додатковий вихід (Старт).

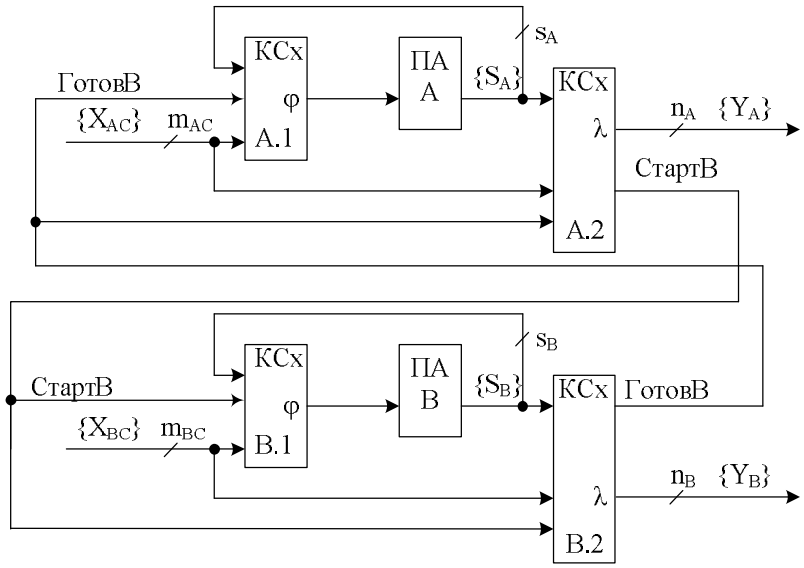

Рис. 1. Результат декомпозиції автомата

Відношення об'ємів ПЗП

$$d=\frac{V_1}{2\cdot V_2}=\frac{2^{s+m}(s+n)}{2\cdot 2^{s-1+(m+m_C)/2+z}(s-1+\frac{n}{2}+z)}=$$

$$\frac{2^{s+m}(s+n)}{2^{s+(m+m_C)/2+z}(s-1+\frac{n}{2}+z)}=2^{(m-m_C)/2-z}\cdot\frac{s+n}{s-1+\frac{n}{2}+z}=K_I\cdot K_O$$

Для отримання позитивного ефекту від декомпозиції потрібно забезпечити умову $\delta>1$, тобто, $K_IK_O>1$. Одним з можливих рішень є варіант, коли одночасно $K_I>1$ і $K_O>1$.

$K_I=2^{(m-n_C)/2-z}$ – коефіцієнт зменшення об'єму ПЗП за рахунок входів, для $K_I>1$ необхідно, щоб $(m-m_C)/2-z>0$, $m-m_C>2z$. Якщо z=1, необхідно, щоб $m-m_C=m_A+m>2$, що практично завжди виконується.

$K_O = \dfrac{s+n}{s-1+\dfrac{n}{2}+z}$ – коефіцієнт зменшення об'єму ПЗП за рахунок виходів, для $K_O > 1$ необхідно, щоб $\dfrac{n}{2}+z-1<n$, $n>2(z-1)$, що також практично завжди виконується (наприклад, для z=1, потрібно n>0).

Висновок

Необхідність проектування структурних алгоритмів та їхніх структурних автоматів породжує проблему "Hardware-Software Codesign". Вирішення цієї проблеми ґрунтується на методах об'єктно-орієнтованого програмування.

Розділова декомпозиція цифрового автомата на два приблизно однакові за кількістю входів, виходів і зворотних зв'язків автомати практично завжди дає виграш у складності, якщо складність вимірювати як об'єм ПЗП мікропрограмного автомату.

1. Voros N. Hardware/Software Co-Design of Complex Embedded Systems // Design Automation for Embedded Systems. 2003. № 8. P . 5 –49. 2. De Micheli G., Gupta R.K. Hardware/software co-design // Proc. of the IEEE. 1997. Vol. 86. № 3. 3. Терехов А.Н., Романовский К.Ю, Кознов Дм. В., Долгов П.С., Иванов А.Н. Объектно-ориентированная методология разработки информационных систем и систем реального времени // Объектно-ориентированное визуальное моделирование / Под ред. А.Н. Терехова. – СПб: Издательство СПб. университета, 1999. – С. 4–20. 4. Avizienis A., Laprie J.-C., Randell B., and Landwehr C. Basic Concepts and Taxonomy of Dependable and Secure Computing, IEEE Transactions on Dependable and Secure Computing, vol. 1, pp. 11-33, 2004. 5. Бондарук А.Б., Глухов В.С., Євтушенко К.С., Оліярник Б.О. Гарантоздатна інтегрована система навігації рухомих наземних об'єктів // Вісник Нац. ун-ту «Львівська політехніка» «Комп`ютерні системи та мережі». – 2008. – № 630. – С.24–30. 6. Лисков Б., Гатэг Дж. Использование абстракций и спецификаций при разработке программ. – М.: Мир, 1989. – 424 с. 7. Шоломов Л.А. Основы теории дискретных логических и вычислительных устройств. М.: Наука. Главная редакция физико-математической литературы, 1980. – 400 с. 8. Выхованец В. С. Синтез эффективных математических моделей дискретной обработки данных на основе алгебраической и понятийной декомпозиции предметной области: Дис. … д-ра техн. наук. На правах рукописи. Российская Академия Наук, Институт проблем управления им. В.А. Трапезникова. – М., 2007. http://rykov-ft.narod.ru/vyxz.pdf. 9. Шоломов Л.А. Разделительная декомпозиция отношений в задачах многокритериального выбора // Дискретный анализ и исследование операции – Апрель—июнь 2001. Серия 1. Т. 8. – № 2. – 63–89. 10. Глухов В.С., Заїченко Н.В. Зменшення апаратних витрат при реалізації мікропрограмних пристроїв // Вісник Держ. ун-ту "Львівська політехніка" "Комп`ютерні системи та мережі". – 1998. – № 383. – С.22–29. 11. Глухов В., Еліас Р. Вибір варіанту декомпозиції цифрових автоматів. Матеріали 4ої міжнародної науково-технічної конференції CSIT'2009 «Комп'ютерні науки та інформаційні технології 2009», 15 – 17 жовтня 2009 р. Україна, Львів. – С. 202–205. 12. Майоров С.А., Крутовских С.А., Смирнов А.А. Электронные вычислительные машины: Справочник по конструированию / Под ред. С.А. Майорова. — М.: Сов. радио, 1975. 13. Anderson Paul. Trends in safety-critical coding and testing practices. In Boards & Solutions (the European embedded computing magazine), June 2009, pp. 32-33. 14. Ален И. Голуб. Правила программирования на С и С++: – М.: БИНОМ, 1996. 15. Математический энциклопедический словарь / Гл. ред. Ю.В. Прохоров; Ред. кол.: С.И. Адян, Н.С. Бахвалов, В.И. Битюцков, А.П. Ершов, Л.Д. Кудрявцев, А.Л. Онищик, А.П. Юшкевич. – М.: Сов. энциклопедия, 1988. – 847 с. 16. Глухов В.С. Обчислювальний пристрій для операцій над еліптичними кривими // Вісник Нац. ун-ту "Львівська політехніка" "Комп`ютерні системи та мережі". – 2006. – № 573. – С.54–61. 17. ДСТУ 4145-2002. Інформаційні технології. Криптографічний захист інформації. Цифровий підпис, що ґрунтується на еліптичних кривих. Формування та перевіряння. – К.: Держ. комітет України з питань технічного регулювання та споживчої політики, 2003. 18. Глухов В.С. Оцінка апаратних витрат на реалізацію багаторівневої комп'ютерної системи // Вісник Нац. ун-ту «Львівська політехніка» «Комп'ютерні науки та інформаційні технології». – 2008. – № 629. – С.13–20.