Системы автоматизированного проектирования

УДК 681.324

О.П. Кургаєв, д-р техн. наук, І.В. Савченко

Проектування вбудованої системи в САПР на основі поведінкової моделі

Незважаючи на широкий спектр засобів проектування вбудованих систем (ВС), процес розробки специфікації системи автоматизований неповністю. Метою роботи є розробка підходу до проектування специфікації ВС на основі поведінкової моделі роботи системи, який додатково розв'язує задачу вибору оптимальної кількості процесів системи та описує перехід від специфікації системи до її апаратно-програмної реалізації. Загальними методами дослідження є: теорія проектування обчислювальних систем; методи оптимізації апаратних витрат; методи проектування ВС із застосуванням мови VHDL. Запропонований підхід може бути застосований при розробці складних ВС.

In spite of that wide spectrum of planning tools of the embedded systems (ES) in CAD has been developed, particular emphasis should be placed on specification development. The purpose objective is to develop the formalist approach of specification planning, which is based on behavioural model and additionally decides the problem of optimal choice of number of process and describes a conversion from the system specification to its final realization. Investigations were made by using such methods: theory of computer systems planning, methods of hardware resources optimization; methods of planning with use of VHDL language. Offered approach can be applied in ES developing.

Ключові слова: *вбудовані системи, система автоматизованого проектування, специфікація системи, етапи проектування, поведінкова модель, розділення специфікації, апаратно-програмне розбиття.*

Вступ

Проектування вбудованих систем (embedded systems) є досить складним процесом, який сьогодні відбувається із застосуванням системи автоматизованого проектування (САПР). Сучасні технології розробки вбудованих систем (ВС) вимагають одночасної розробки інструментальними засобами САПР як програмного, так і апаратного забезпечення. Успішне розподілення функцій між програмним та апаратним забезпеченням здатне забезпечити невисоку ціну, оптимальну продуктивність, високу надійність, можливість подальшої модифікації ВС.

Більшість етапів проектування в САПР автоматизовано, однак початковий процес – процес розробки специфікації системи у зв'язку із його складністю не автоматизований. В роботах [1-4] наведено формальні підходи до розробки специфікації системи у середовищі САПР, однак вони потребують уточнення, зокрема, відносно розв'язку задачі вибору оптимальної кількості процесів системи та особливостей переходу від специфікації системи до її реалізації.

У зв'язку з цим метою роботи є розробка формального підходу до проектування специфікації ВС на основі поведінкової моделі роботи системи, який додатково розв'язує задачу вибору оптимальної кількості процесів системи та описує перехід від специфікації системи до її апаратно-програмної реалізації.

1. Етапи проектування ВС

На рис. 1 представлено основні етапи проектування ВС: розробка специфікації системи, розділення специфікації системи, апаратно-програмне розбиття, апаратно-програмна реалізація. На першому етапі відбувається розробка специфікації системи, яка представляється у вигляду напрямленого графу, вузлами якого є процеси, кожний із яких виконує певну послідовність дій. Для забезпечення обміну даними між процесами встановлюються зв'язки у вигляді дуг графу. Граф процесів може бути побудований вручну або у вигляді файлу опису поведінки системи на мові високого рівня, який в подальшому за допомогою інструментальних засобів САПР може бути синтезований у модель системи на рівні регістрових передач (RTL-рівень).

2. Мови специфікації системи

Класичними методами опису специфікації системи є опис поведінкової моделі системи за до-

помогою скінчених автоматів, мереж Петрі або графів. Сучасні САПР пропонують широкий спектр інструментальних засобів для опису апаратури та програмного забезпечення. Для опису структури, поведінкової моделі, а також логічного

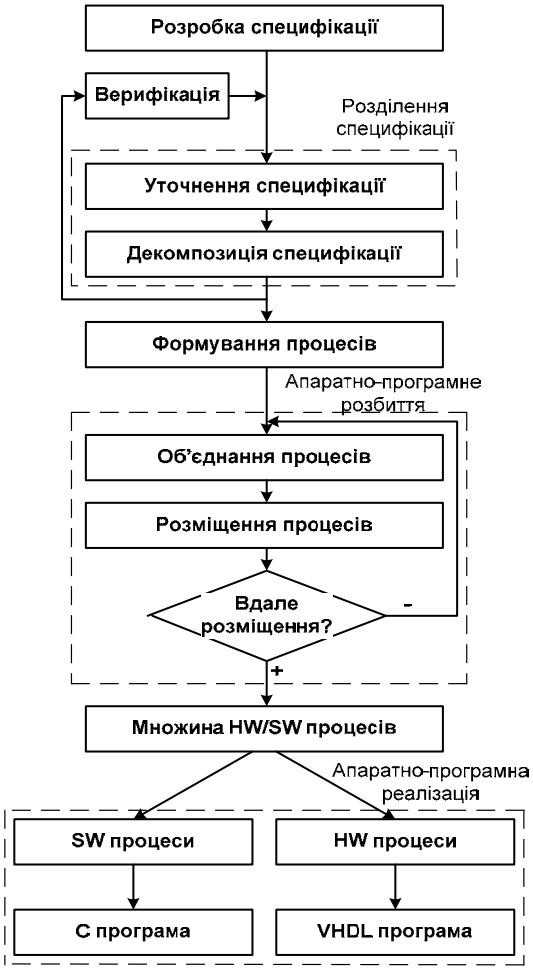

Рис. 1. Етапи проектування ВС

рівня застосовуються такі мови програмування апаратури [5]: VHDL, Verilog®-HDL, SPICE, Verilog®-A, VHDL-AMS, Verilog®-AMS, SystemVerilog, SystemC, аналогічно для опису програмного забезпечення – C, C++, Visual Basic, JAVA, SystemC. Головними особливостями мов програмування апаратури є забезпечення широких можливостей опису моделі системи [5]:

- Високий ступінь абстракції. Дозволяє зосередитись тільки на розробці абстрактної моделі пристрою, що не залежить від реалізації. Зокрема, на початковому етапі можна отримати моделі процесів, які можуть бути реалізованими або апаратно, або програмно.
- Паралелізм процесів. Дана властивість забезпечує можливість організувати моделювання одночасної роботи декількох процесів системи.
- Часові константи. Дозволяють встановлювати обмеження на часові затримки при виконанні процесів системи та обміну даними між ними.
- Апаратні константи. Дозволяють встановлювати обмеження до витрат апаратних ресурсів, специфіку розміщення ресурсів ПЛІС.
- Бібліотеки компонентів. Дозволяють скоротити час розробки специфікації системи за рахунок використання вже готових, оптимізованих та верифікованих IP-модулів і стандартних елементів.
- Можливість компіляції та аналізу. Дозволяє перевірити на коректність представлення специфікації системи у середовищі САПР, провести оптимізацію, верифікацію, при необхідності згенерувати із текстового опису графічне представлення поведінкової моделі системи у вигляді графу.
- Можливість комбінованого опису системи. Завдяки цьому модель системи може бути представлена із застосуванням поведінкового (внутрішня структура пристрою, що реалізує процес, не специфікується, задається тільки функціональний опис), структурного (описується лише внутрішня структура пристрою, що реалізує процес) або змішаного опису (комбінація першого та другого).

3. Розділення специфікації

3.1. Уточнення специфікації

Процес розробки специфікації ВС розпочинається із її опису на системному рівні, на наступному кроці відбувається процес її ітеративного розділення на ще менші частини. Процес розділення відбувається до тих пір, поки не буде досягнуто рівня бібліотечних функцій та компонентів САПР. На рис. 2 представлений процес розділення специфікації системи.

На нульовому рівні специфікація представляється на самому найвищому абстрактному рівні (на рівні поведінкової моделі). На наступних рівнях відбувається розділення функціональних блоків специфікації на більш елементарні блоки, при цьому поведінкова модель системи не повинна змінюватись. Після кожного рівня розділення специфікації відбувається процес її верифікації, який виконується методом моделювання поведінки системи при заданих вхідних сигналах.

Головною метою процесу розділення специфікації системи є отримання нової (більш деталізованої або уточненої) специфікації системи, яка може бути успішно впроваджена за допомогою інструментальних засобів САПР. Даний процес включає також визначення множини внутрішніх сигналів, які будуть використані для об-

міну даними та командами поміж функціональними модулями.

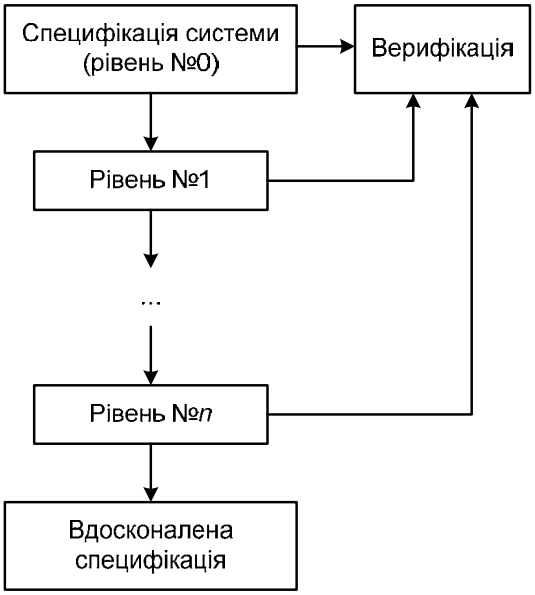

Рис. 2. Процес розділення специфікації

3.2. Декомпозиція специфікації

Найголовнішим завданням на даному етапі є визначення основних функціональних блоків (модулів) системи. Дані функціональні модулі будуть використовуватися на наступних етапах проектування для розв'язку задачі апаратно-програмного розбиття. Етап декомпозиції системи передбачає розділення специфікації на процеси (task), які в подальшому можна буде синтезувати у програмне або апаратне забезпечення.

Досить суттєвою задачею, на цьому етапі, є розділення специфікації системи на множину таких функціональних модулів, які забезпечать найліпшу продуктивність роботи ВС, вартість розробки та можливість її оптимізації. Однак на даному етапі розробки системи не слід визначати те, яким саме чином буде реалізовано конкретний процес (програмно чи апаратно), оскільки це може зменшити вірогідність отримання системи, параметри якої близькі до оптимальних. Крім того це також заважатиме реорганізовувати систему в подальшому. Тому для досягнення найбільшої незалежності від завершальної реалізації системи (на логічному рівні), на етапі декомпозиції системи не слід прив'язуватися до конкретної архітектури, а намагатися при цьому використовувати поведінкові моделі для опису функціональних модулів.

Процес декомпозиції розпочинається із перетворення основної специфікації системи на множину паралельно взаємодіючих між собою процесів, якщо між ними не присутня залежність по даним. Якщо така залежність присутня, то вона встановлюється завдяки послідовному виконанню таких процесів. В кінцевому випадку специфікація системи, представляється у вигляді одного або декількох файлів. На рис. 3 представлено структуру файлу специфікації системи, головними частинами якого є інтерфейсна та поведінкова частина [6].

<Множина бібліотек>

<Інтерфейсна частина>

<Множина портів вводу-виводу>

<Поведінкова модель системи>

<Множина внутрішніх сигналів>

<Процес 1>

<Процес 2>

...

<Процес n>

Рис. 3. Структура файлу специфікації системи

Інтерфейсна частина містить бібліотеки, множину портів та сигналів, поведінкова частина містить опис процесів системи та їх взаємодію між собою. Кожний процес описується у вигляді послідовності окремих дій.

Крім того в процесі декомпозиції системи можна також додатково застосовувати спеціальні директиви компілятора інструментального засобу синтезу САПР, які забезпечують задані критерії декомпозиції системи. Тому для досягнення оптимальних результатів при декомпозиції системи, бажано, щоб початкова специфікація системи (на 0-му рівні) була представлена у такому стилі, який буде сприяти зручному виявленню в ній множини процесів.

4. Апаратно-програмне розбиття

Після розділення специфікації згідно заданим критеріям системи, розпочинається етап програмно-апаратного розбиття системи. Головною задачею даного етапу є розподілення процесів на дві категорії: 1) процеси, які буде реалізовано програмно (SW-процеси); 2) процеси, які буде реалізовано апаратно (HW-процеси). Результатом такого розбиття є отримання структури системи, яка враховує задані параметри: продуктивність, часові характеристики, вартість, енергоспоживання. Етап апаратно-програмного розбиття може бути розділений на два етапи: об'єднання та розміщення процесів.

4.1. Об'єднання процесів системи

Головною метою об'єднання процесів є мінімізація їх кількості, що буде сприяти спрощен-

ню наступних етапів розв'язку задачі апаратно-програмного розбиття. Якщо кількість процесів системи є занадто великою, то розв'язок задачі програмно-апаратного розбиття значно ускладнюється. Натомість, при надто малій кількості процесів можна отримати систему, параметри якої суттєво відрізняються від оптимальних.

Для визначення оптимальної кількості процесів системи застосовується алгоритм об'єднання процесів, який мінімізує кількість транзакцій між ними. Для цього спочатку необхідно провести аналіз кількості транзакцій (операцій зв'язку між процесами) між усіма процесами системи. Далі будується граф, в якому об'єднуються в один процеси, між якими присутня найбільша кількість транзакцій. Отриманий граф повторно аналізується на кількість транзакцій між процесами та при необхідності відбувається повторне об'єднання процесів.

4.2. Розміщення процесів

На етапі розміщення виконується розбиття всієї множини процесів на два типи, ті, що будуть реалізовані програмно, та ці, що буде реалізовано апаратно. Метою такого розбиття є отримання системи із найбільшою продуктивністю роботи чи найменшою вартістю виробництва. Для досягнення цього потрібно отримати найоптимальнішу реалізацію кожного із процесів. Далі будується функція вартості розробки системи та мінімізується її значення ітеративним перебором множини можливих варіантів реалізації системи. Система, у якій функція вартості буде мінімальною і є оптимальною.

5. Реалізація системи

5.1. Написання драйверу системи

На даному етапі множина SW-процесів є основою для реалізації драйверу системи. Одним із основних завдань на даному етапі є визначення інтерфейсів та алгоритму взаємодії між модулями системи. Необхідність цієї розробки викликана тим, що різні програмні модулі системи для обміну даними і командами між ними використовують спільно такі апаратні ресурси, як універсальний процесор, сопроцесор, пам'ять, шини.

При написанні драйверу необхідно дотримуватись коректної послідовності виконання процесів, якщо між ними присутній зв'язок по даних, в той же час забезпечувати коректну синхронізацію апаратних модулів. Драйвер системи повинен враховувати часові затримки в апаратних модулях та в їх каналах обміну даних, встановлювати пріоритетність використання апаратних ресурсів. Це дасть змогу оптимально використовувати апаратні ресурси системи.

5.2. Особливості реалізації процесів на мові апаратури

Множина HW-процесів є основою специфікації на реалізацію апаратури. Для представлення специфікації системи у САПР застосовується спеціальний набір інструментальних засобів, які можна умовно розділити на графічні та текстові. Текстові інструментальні засоби здатні працювати як із мовами апаратури, так із об'єктно-орієнтованими мовами програмування, а також здатні трансформувати графічне представлення системи в текстовий вигляд. В кінцевому вигляді специфікація системи повинна бути представлена у формі множини текстових файлів опису структури та алгоритму роботи апаратури системи на мові високого рівня. Розглянемо структуру файлу опису апаратури.

Будь-який файл опису апаратури складається із декларативної та архітектурної частини. В декларативній частині описуються специфікація інтерфейсу пристрою (вхідні та вихідні порти, їх характеристики). В архітектурній частині можуть бути визначені функції та алгоритм роботи кожного із функціональних модулів, а також особливості їх сполучення між собою.

Для опису архітектури системи використовують поведінковий, структурний та структурно-поведінковий опис:

- *Поведінковий опис*. Внутрішня структура системи не специфікується, задається функціональний опис поведінки системи відносно до його вхідних сигналів та станів. Описуються особливості формування вихідних сигналів на основі вхідних сигналів та поточного стану системи.
- *Структурний опис*. Описується внутрішня структура системи, як множина функціональних модулів та зв'язків між ними. Поведінка системи, особливості формування вихідних сигналів при надходженні вхідних сигналів визначаються складом та зв'язками між модулями, що входять до складу системи.
- *Структурно-поведінковий опис*. Застосовується комбінація першого та другого методів.

На основі специфікації, яка представляється на мові апаратури, САПР здатна автоматично згенерувати файл прошивки кристалу ПЛІС, інтеграція якого в кристал забезпечить реалізацію заданої апаратної частини системи.

Висновки

Сьогодні САПР не забезпечують автоматизацію проектування ВС повністю. Найбільші

труднощі в даному процесі пов'язані із розробкою специфікації системи, в якій буде оптимально розподілено функції між програмним та апаратним забезпеченням. В даній роботі запропоновано формальний підхід до проектування ВС, на основі опису поведінкової моделі системи, який додатково розв'язує задачу вибору оптимальної кількості процесів системи та описує перехід від специфікації системи до її апаратно-програмної реалізації. Запропонований підхід передбачає розв'язання наступних задач: розділення специфікації, апаратно-програмне розбиття та апаратно-програмна реалізація системи. Даний підхід може застосовуватися при розробці складних ВС у середовищі САПР ПЛІС.

Література

1. Salim Ouadjaout, Dominique Houzet. Generation of Embedded Hardware/Software from SystemC // EURASIP Journal on Embedded Systems. – 2006. – Vol. 2006. – P. 12.
2. *Haubelt C.* , Falk J. , Keinert J. , Schlichter T. , Streubuhr M., Deyhle A., Hadert A., Teich. J. A SystemC-Based Design Methodology for Digital Signal Processing Systems // EURASIP Journal on Embedded Systems. – 2007. – Vol. 2007. – P. 23.
3. *Knerr B.,* Holzer M., Rupp M. RRES: A Novel Approach to the Partitioning Problem for a Typical Subset of System Graphs // EURASIP Journal on Embedded Systems. – 2008. – Vol. 2008. – P. 14.
4. *Domer R.,* Gerstlauer A., Peng J., Dongwan Shin, Cai L., Yu H., Abdi S. , Gajski D.D.. System-on-Chip Environment: A SpecC-Based Framework for Heterogeneous MPSoC Design // EURASIP Journal on Embedded Systems. – 2008. – Vol. 2008. – P. 14.
5. Grout I. Digital systems design with FPGAs and CPLDs. Copyright © 2008, Elsevier Ltd. – 2008. – P. 724.
6. *Суворова Е.А.,* Шейнин Ю.Е. Проектирование цифровых систем на VHDL. – СПб.: БХВ-Петербург, 2003. – 576 с.

Інститут кібернетики ім. В.М. Глушкова НАН України