VÀI NÉT TỔNG QUAN VỀ ĐỒNG THIẾT KẾ

CODESIGN: AN OVERVIEW

Đinh Đức Anh Vũ
Khoa Công nghệ Thông tin, trường ĐH. Bách Khoa Tp. HCM

TÓM TẮT

Đa số các hệ thống số đều khả lập trình và do đó bao gồm cả thành phần phần cứng và phần mềm. Giá trị của một hệ thống thường được xác định dựa vào một số mục đích đặc thù đối với lĩnh vực ứng dụng của hệ thống đó (ví dụ như hiệu năng, chi phí thiết kế và sản xuất, dễ lập trình) và nó phụ thuộc vào cả thành phần phần cứng và phần mềm. "Đồng thiết kế" phần cứng và phần mềm (HW/SW codesign) là quá trình thiết kế tận dụng khả năng của phần cứng và phần mềm trong cùng một thiết kế để đạt được các mục tiêu ở mức hệ thống. Bài báo này nhằm giới thiệu về codesign và các vấn đề cơ bản liên quan đến qui trình thiết kế này. Các phương pháp luận codesign nổi bật cũng sẽ được đề cập đến. Phần tham khảo sẽ chỉ ra cho các đọc giả quan tâm những vấn đề đặc thù của qui trình codesign để họ có thể tìm đọc thêm chi tiết.

ABSTRACT

The majority of digital systems is programmable, and thus consists of hardware and software components. The value of a system can be measured by some objectives that are specific to its application domain (e.g., performance, design and manufacturing cost, ease of programmability) and it depends on both the hardware and software components. Hardware/Software (HW/SW) codesign means meeting system-level objectives by exploiting the synergism of HW/SW through their concurrent design. This paper provides a general introduction to HW/SW codesign and its fundamental issues. Current state-of-the-art codesign methodologies are addressed. The bibliography should point the interested readers to specific codesign issues to go further into details.

1. GIỚI THIỆU

Codesign thường được dùng để nói đến việc thiết kế hệ thống tích hợp dùng cả hai thành phần: phần mềm (SW) và phần cứng (HW). Hệ thống kết hợp chặt chẽ giữa các mođun phần cứng và phần mềm được ứng dụng để giải quyết một số vấn đề. Những hệ thống này không có gì mới. Tuy nhiên các phương pháp luận hiện đang được ứng dụng và các kỹ thuật thiết kế dung hòa giữa HW và SW gần đây mới trở nên nổi bật ([10][19]). Có nhiều nguyên nhân đưa đến sự quan tâm đối với codesign:

- Sự tiến bộ của công nghệ (môi trường cho phép đặc tả và mô phỏng ở mức hệ thống, kỹ thuật làm mẫu mềm, phương pháp thiết kế và kiểm tra hình thức, tổng hợp cấp cao, và sự trưởng thành của các công cụ trợ giúp thiết kế) đã mở ra một hướng phát triển mới cho codesign.
- Sự gia tăng về sự đa dạng và độ phức tạp của các ứng dụng hệ thống nhúng đòi hỏi những phương pháp tiên tiến cho việc phát triển cả phần cứng và phần mềm.
- Việc tối ưu chi phí và hiệu năng và việc giảm đáng kể thời gian đưa sản phẩm ra thị trường (time-to-market) là những vấn đề quan trọng đối với các công nghệ cao.

Codesign khác với các phương pháp tiếp cận truyền thống ở chỗ qui trình thiết kế phần cứng quan hệ chặt chẽ với qui trình thiết kế phần mềm của nó. Quyết định trong qui trình thiết kế phần cứng cũng sẽ ảnh hưởng quan trọng đến qui trình thiết kế phần mềm. Do đó đối với codesign, các vấn đề cần được xem xét giải quyết một cách toàn diện.

2. CÁC PHƯƠNG PHÁP LUẬN ĐỒNG THIẾT KẾ

Theo truyền thống, việc thiết kế các hệ thống lai (heterogeneous system) dựa trên nhiều bước thiết kế HW/SW liên tục nhau. Trong các bước này, người thiết kế điều chỉnh đặc tả và xây dựng mẫu thử. Dựa vào kinh nghiệm của người thiết kế và nhật ký thiết kế (profile) của hệ thống, các chức năng được chuyển từ SW sang HW hoặc ngược lại trong mỗi chu kỳ. Codesign ngày nay chủ yếu dựa vào những kỹ thuật và phương pháp đã được ứng dụng thành công trước đây. Các đóng góp mới chủ yếu trong lĩnh vực trao đổi thông tin giữa các công cụ, giao tiếp giữa các công cụ, cơ chế phân hoạch HW/SW và các môi trường thiết kế cao cấp.

2.1. Phương pháp tiếp cận truyền thống

Các môi trường thiết kế tổng quát cho phép bao đóng (encapsulate) và tích hợp các công cụ cũng như hỗ trợ quản lý đối với các thiết kế mang tính cộng tác. Hình 1 mô tả sơ đồ các quá trình và các hoạt động cơ bản trong một phương pháp luận codesign.

Hình 1 Phương pháp luận codesign truyền thống

2.1.1. Phân tích ràng buộc và yêu cầu

Trong bước này, các đặc tính của hệ thống cơ bản được định nghĩa dựa trên các đặc tả của khách hàng và người dùng. Các mục tiêu, yêu cầu và ràng buộc của dự án do khách hàng đặt ra thường thiếu tính đầy đủ và chặt chẽ. Do đó quá trình phân tích ràng buộc và yêu cầu này giúp bổ khuyết tính chặt chẽ và xác định những thông tin còn thiếu. Những vấn đề thiết kế được thực hiện ở quá trình phân tích bao gồm tính thị trường dựa trên những nghiên cứu về các yêu cầu của người dùng, yêu cầu hiệu năng thời gian thực, công nghệ hiện thực, khả năng lập trình, công suất tiêu thụ, kích thước sản phẩm, chi phí thiết kế và sản xuất, môi trường sử dụng, độ tin cậy, bảo trì, phát triển của thiết kế và chi phí tái sinh.

2.1.2. Đặc tả hệ thống

Đặc tả hệ thống là kết quả của quá trình phân tích. Đó là một bản đặc tả hình thức, thích hợp cho người thiết kế dựa vào để phát triển các giải thuật mô hình hóa. Các giải thuật này có thể được mô phỏng bằng cách dùng các công cụ làm mẫu nhanh dựa trên sơ đồ trạng thái ([1]) hoặc các mô hình hàng đợi đối với việc mô phỏng theo hiệu năng. Quá trình phát triển thông thường là sự kết hợp của sự thử nghiệm, kinh nghiệm và phán đoán có căn cứ. Bước mô phỏng thực tế là cơ hội đầu tiên của người thiết kế để minh họa ý tưởng cho người quản lý và đội ngũ tiếp thị dự án.

2.1.3. Phân hoạch HW/SW

Vấn đề cốt lõi của phương pháp luận codesign là quá trình phân hoạch HW/SW, trong đó người thiết kế hoặc công cụ thiết kế phải quyết định thành phần nào của hệ thống được hiện thực bằng HW, phần nào sẽ được thực hiện bằng SW. Quá trình thiết kế hệ thống bắt đầu bằng việc mô hình hóa, tức là người thiết kế mô tả hành vi của hệ thống một cách hình thức. Có nhiều phương pháp phân tích và đánh giá hệ thống, tùy thuộc vào mô hình lý thuyết, mức độ trừu tượng và cách thức tích hợp ([2]). Thông thường, người ta dùng các kỹ thuật xác định (hoặc thống kê) cũng như các phương pháp nhật ký (profiling) để tìm ra một phân hoạch (HW/SW) tốt ([3]). Trong số các cách thức phân hoạch, kỹ thuật phân hoạch xác định (deterministic), thống kê (statistical), điểm chuẩn (benchmarking) và nhật ký thường được dùng phổ biến nhất. Kỹ thuật phân hoạch xác định được áp dụng cho

mô hình đặc tả đầy đủ đã loại ra tất cả các phụ thuộc dữ liệu và chi phí của các thành phần đều phải biết trước. Kỹ thuật này thường đạt được một phân hoạch tốt, nhưng sẽ thất bại nếu có một phần tử dữ liệu không biết trước. Trong trường hợp này, người ta phải dùng kỹ thuật phân hoạch theo thống kê. Kỹ thuật này dựa vào việc phân tích các hệ thống và một số tham số thiết kế.

Phương pháp nhật ký thường dựa vào việc khảo sát dòng điều khiển và dòng dữ liệu trong kiến trúc hệ thống để xác định phần nào bị quá tải về khối lượng tính toán và hiện thực nó bằng phần cứng. Phương pháp này có thể cho kết quả tốt ngay cả khi tồn tại những điều kiện thực thi phụ thuộc nhiều vào dữ liệu.

Cơ chế phân hoạch hoàn toàn tự động vẫn chưa có. Các công cụ phân tích và đánh giá cần thiết cho việc phân hoạch hiện mới đang được phát triển. Người dùng có thể dựa vào những kết quả đánh giá tự động để đưa ra các quyết định phân hoạch của mình. Cách này chỉ đáp ứng cho tình hình hiện tại, nhưng lại chưa đủ cho các ứng dụng công nghiệp. Do đó, theo chúng tôi, cần phải tập trung nhiều hơn vào việc phát triển các phương pháp phân hoạch và các công cụ cho nhu cầu công nghiệp.

Quá trình phân hoạch hệ thống có thể được thực hiện ở các mức trừu trượng hoặc ở các giai đoạn khác nhau trong qui trình thiết kế ([4]). Khi được thực hiện ở cấp cao, quá trình phân hoạch giống như quá trình ánh xạ các mođun. Bước ánh xạ này có thể được thực hiện sớm hoặc trễ trong qui trình thiết kế. Ánh xạ sớm thường được sử dụng trong công nghiệp vì nó cho phép lập kế hoạch trước và chọn lựa quyết định thiết kế dễ dàng hơn. Tuy nhiên cách này chỉ cho phép rất ít sự thay đổi từ phía khách hàng. Ngược lại, ánh xạ trễ cho phép có những giải pháp tốt hơn về mặt hiệu năng và vấn đề điều chỉnh mục tiêu theo yêu cầu của khách hàng.

2.1.4. Tổng hợp và cấu hình phần cứng

Trong bước tổng hợp và cấu hình phần cứng, người ta thiết kế một mô hình phần cứng để thực hiện các mã chương trình do quá trình tạo phần mềm sinh ra (ở đây ta giả sử việc tạo ra phần mềm được thực hiện trước). Mô hình này được xây dựng từ những mô tả phần cứng do quá trình phân hoạch tạo ra từ trước. Quá trình tổng hợp bao gồm việc ánh xạ công nghệ, nghĩa là chuyển những mô tả phần cứng (ví dụ như VHDL, Verilog, C, …) thành những khối vật lý được thực hiện bằng phần cứng ([5]). Sự lựa chọn này quyết định bộ sinh mã chương trình trong quá trình tạo ra phần mềm và ảnh hưởng đến tốc độ thực thi của mã chương trình. Nó cũng khiến quá trình cấu hình động phần cứng và các thành phần của thư viện thiết kế được thực hiện dễ dàng. Thông thường, hầu hết phần cứng được tạo ra với các công cụ tổng hợp ([5][6]).

Bên cạnh các thành phần phần cứng, các thiết bị khả lập trình như FPGA và bộ đồng xử lý khả cấu hình còn yêu cầu thêm các mã chương trình. Các phần mềm cấu hình cho các mođun phần cứng như FPGA, CLB động, tham số thiết lập cho ASIC, đơn giản đều có được từ quá trình tổng hợp. Ngoài ra, các bộ đồng xử lý cấu hình động cũng cần các mã chương trình sinh từ các quá trình tạo ra phần mềm. Các thành phần phần mềm phụ thuộc vào các quyết định đưa ra sau bước phân hoạch và do đó quá trình tạo ra phần mềm phụ thuộc vào các quyết định trong bước tổng hợp phần cứng trước đó.

Việc cấu hình phần cứng không chỉ hạn hẹp trong việc sắp xếp các hàm logic ở mức cổng và mức logic. Người ta có thể ứng dụng các kỹ thuật cấu hình dựa trên cơ sở tri thức để xây dựng các mô hình phần cứng ([7]). Theo cách này, các mođun với độ phức tạp khác nhau, từ các khối logic cơ bản đến dãy các bộ xử lý, và từ các lõi xử lý ở mức hành vi đến dãy các bộ xử lý phức tạp, được lưu trữ trong cơ sở tri thức. Dựa vào đặc tả của phần cứng, một chương trình chọn lựa các thành phần thích hợp từ cơ sở dữ liệu này và từ đó xây dựng nên phần cứng.

2.1.5. Tạo và tham số hóa phần mềm

Trong giai đoạn tạo và tham số hóa phần mềm, các mođun phần mềm sẽ được tạo ra cho phần cứng đã được tổng hợp và cấu hình (ở đây ta giả thuyết ngược lại: phần cứng được tạo ra trước phần mềm). Sự tương tác giữa các thành phần phần cứng và phần mềm được đảm bảo bởi một bộ định thời trong phần cứng và phần mềm đó. Do đó, tất cả mã chương trình phải được bổ sung thêm các chương trình con định thời. Bởi vì việc tạo phần mềm phụ thuộc vào mô hình phần cứng và kiến trúc của nó, người ta phải chọn kiến trúc mục tiêu. Việc chọn lựa này cho phép sử dụng các mã chương trình chuẩn và các thư viện các thành phần

phần cứng. Các thành phần từ các thư viện này chỉ cần tham số hóa cho phù hợp với nhu cầu cần thiết.

2.1.6. Tổng hợp giao diện

Quá trình tổng hợp giao diện thiết lập sự đồng bộ hóa phần cứng và phần mềm ([8]).Thông thường, các kỹ thuật được dùng trong quá trình này là sự trao đổi tín hiệu (HW), semaphore (SW), hoặc cơ chế ngắt quãng. Việc hiện thực thay đổi từ logic đặc thù cho tới các thiết bị logic có khả năng cấu hình. Có một số phương pháp tiếp cận kết hợp với một bộ định thời trung tâm bằng phần mềm và bộ định thời này sẽ gởi các tín hiệu để kích hoạt các quá trình phần cứng.

2.1.7. Tích hợp và đồng mô phỏng (cosimulation)

Quá trình tích hợp và đồng mô phỏng HW/SW đối ngẫu với quá trình phân hoạch; và độ phức tạp của các quá trình này là như nhau ([9]). Quá trình này sẽ tạo ra mẫu thử mà có thể xây dựng vật lý hoặc thực hiện trên chương trình mô phỏng hệ thống lai. Chương trình đồng mô phỏng HW/SW phải thực thi các modun được sinh ra trong quá trình tạo phần mềm trên các kiến trúc phần cứng ([10]).

Quá trình thực hiện đồng mô phỏng chiếm rất nhiều thời gian. Do đó người ta phải dùng các kỹ thuật khác nhau để tăng tốc độ mô phỏng. Bài báo [11] đề nghị dùng các mô hình trừu tượng (abstract model) của các bộ xử lý ở mức hành vi thay vì dùng mô hình ở lớp cổng. Trong các phòng thí nghiệm công nghiệp, người ta dùng các máy mô phỏng với tốc độ mô phỏng cao hơn. Cách tiếp cận tổng quát của họ là dùng một vài máy mô phỏng tương tác với nhau để giải quyết vấn đề.

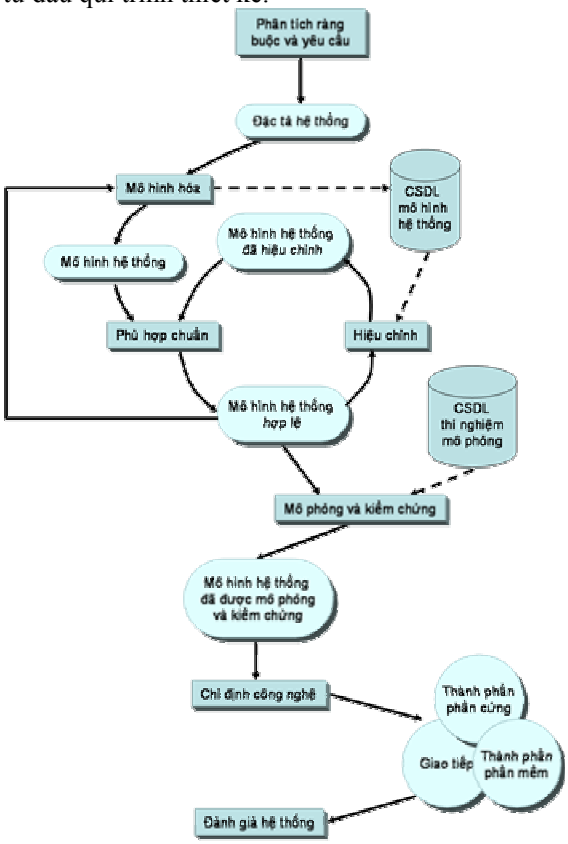
2.1.8. Kiểm chứng thiết kế

Kiểm chứng thiết kế là quá trình để bảo đảm hệ thống tạo ra theo qui trình thiết kế thỏa mãn các đặc tả ban đầu ([12]). Thông thường, người ta kiểm chứng các thiết kế sau khi đã được mô phỏng, nhưng trước khi thực hiện mẫu thử vật lý.

2.2. Phương pháp tiếp cận dựa theo mô hình (model-based)

Trong phần này chúng tôi trình bày sự khác biệt của phương pháp luận dựa theo mô hình ([13]) đối với phương pháp truyền thống. Phương pháp luận này được minh họa trong Hình 2.

Phương pháp luận truyền thống cho phép thực hiện việc phân hoạch trước, nên rất thích hợp trong công nghiệp. Như đã trình bày, việc phân hoạch HW/SW trước sẽ dễ dàng hơn cho việc hoạch định kế hoạch phát triển sản phẩm từ đầu qui trình thiết kế.



Hình 2 Phương pháp luận codesign dựa trên mô hình

Tuy nhiên việc phân hoạch sau (ánh xạ - binding) có thể đưa ra giải pháp tốt hơn về mặt điều chỉnh mục tiêu thiết kế, hiệu năng và chi phí sản phẩm. Trong phương pháp luận này, người ta dùng một bước tinh chỉnh (refinement) để đảm bảo cho việc chỉ định công nghệ sau này đối với mô hình hệ thống hợp lệ. Cách tiếp cận này thúc đẩy việc tái sử dụng các khối thiết kế có sẵn mà không cần quan tâm đến công nghệ hiện thực của nó. Điều này làm giảm chi phí và thời gian thiết kế. Như được minh họa trong Hình 2, vòng lặp tinh chỉnh mô hình hệ thống độc lập với việc chỉ định công nghệ. Đây là một ưu điểm quan trọng khi thiết kế các hệ thống lớn bởi vì việc điều chỉnh công nghệ (hay dịch chuyển từ HW sang SW hoặc ngược lại) thường đòi hỏi sự thay đổi nhiều phần giao tiếp. Sự thay đổi này

ảnh hưởng không chỉ đến hiệu năng mà còn đến hành vi của hệ thống.

2.2.1. Mô hình hóa

Quá trình mô hình hóa hệ thống là quá trình cơ sở của phương pháp luận dựa theo mô hình. Trong hoạt động mô hình hóa, một mô hình là một tập hợp các lệnh để tạo ra dữ liệu hành vi. Dữ liệu tạo ra hợp lệ phải tương đương với (hoặc là tập con của) dữ liệu trong hệ thống thực. Ta không thể xây dựng được các hệ thống tương ứng cho tất cả các hành vi xuất/nhập của hệ thống thực. Chính vì vậy ta xây dựng mô hình cho tập các vấn đề, mục tiêu, và mục đích mà hệ thống đang thiết kế nhắm đến. Mô hình này được xây dựng ở mức trừu tượng tương ứng với những vấn đề này. Bộ mô phỏng khi đó được hiểu là quá trình tính toán để tạo ra dữ liệu khi người ta cho trước các lệnh mô hình.

Việc sử dụng các mô hình thích hợp khiến cho việc chuyển đổi giữa các mức trừu tượng dễ dàng hơn. Trong quá trình mô hình hóa, các thành phần hệ thống, các biến mô tả và tương tác giữa các thành phần được đặc tả ở mức hành vi. Các mô tả hình thức toán học đặc thù (ví dụ đặc tả của các sự kiện rời rạc, máy trạng thái hữu hạn, …) được dùng như những phương tiện cho việc đặc tả mô hình. Bởi vì mô hình là một dạng mã hóa hành vi của hệ thống (thường được gọi là blueprint), người ta không cần thiết phải quyết định cách hiện thực của các thành phần thiết kế ở giai đoạn này.

Trong khi mô hình hóa một hệ thống, người ta tận dụng tối đa khái niệm mođun tính và phân cấp. Cụ thể, các mô hình được xây dựng từ các khối cơ bản theo hình thức phân cấp. Điều này chỉ khả thi nếu mô hình được đặc tả hình thức theo hành vi (nếu không, độ phức tạp hình thức sẽ rất khó điều khiển được).

2.2.2. CSDL mô hình hệ thống

Cơ sở dữ liệu mô hình hệ thống là tập hợp các thành phần hệ thống mà người ta có thể dùng trong quá trình mô hình hóa hệ thống cần thiết kế. Các mô hình trong CSDL được xem như kho tri thức về các hệ thống đã thiết kế. Khi một thiết kế được chỉnh sửa, cải tiến, và tinh chỉnh, các mô hình mới được thêm vào CSDL cho việc sử dụng có thể trong tương lai.

2.2.3. Phê chuẩn (validation)

Người ta gọi một mô hình của một thiết kế là hợp lệ (đối với đặc tả thiết kế) nếu các lệnh mô hình có thể tạo ra các hành vi xuất/nhập phù hợp với những gì mà đặc tả của thiết kế đó qui định.

2.2.4. Tinh chỉnh (refinement)

Trong bước tinh chỉnh, một mô hình hệ thống hợp lệ sẽ được tinh chỉnh thành một mô hình có mức độ chính xác cao hơn. Thông thường trong bước này, người ta phân rã (decompose) mô hình thành các mô hình con cơ bản hơn. Việc kết nối và tương tác giữa các mô hình cũng được định nghĩa ở đây.

2.2.5. Mô phỏng

Như đã trình bày trong phần trước, mô phỏng là quá trình tạo ra các dữ liệu với các lệnh mô hình mã hóa được cho trước. Các bộ mô phỏng nên được kiểm chứng sao cho dữ liệu được tạo ra là đúng. Dữ liệu tạo ra bởi các bộ mô phỏng được dùng để đánh giá các giải pháp thiết kế.

2.2.6. CSDL mô phỏng

Rất dễ nhận thấy rằng bất cứ hệ thống thực nào cũng có thể được phân tích (và thiết kế) từ một triển vọng đa mục tiêu. Các mục tiêu hướng đến quá trình xây dựng mô hình bằng cách giúp phân ranh giới hệ thống và xác định các thành phần mô hình thích hợp. Các mục tiêu cũng có vai trò quan trọng trong quá trình đặc tả các khía cạnh kiểm tra mô phỏng đối với mô hình của hệ thống thực. Khái niệm cơ bản đối với quá trình này là "khung thử nghiệm" (experimental frame), nghĩa là sự đặc tả các tình huống mà chúng ta có thể quan sát và thử nghiệm với một mô hình. Định nghĩa của khung thử nghiệm phản ảnh mục tiêu của việc mô hình hóa bằng cách đưa các dữ liệu vào mô hình, quan sát phản ứng của mô hình thông qua các dữ liệu thu được ở ngõ xuất và điều khiển việc kiểm tra bằng cách đặt các ràng buộc về giá trị đối với các biến trạng thái của mô hình. Dữ liệu thu thập từ việc kiểm tra được dùng để đánh giá giải pháp thiết kế đề nghị so với bản đặc tả.

Chú ý phải phân biệt rõ giữa cái điều khiển mô hình, cái được quan sát ở ngõ xuất và bản thân mô hình. Nếu được vậy, một mô hình có thể được phối với nhiều khung thử nghiệm khác nhau, mỗi cái tương ứng với một đặc tả hiệu năng riêng biệt.

2.2.7. Chỉ định công nghệ (technology assignment)

Do tính độc lập của việc hiện thực hệ thống vẫn tiếp tục tồn tại cho đến giai đoạn này, sự chỉ định công nghệ không chỉ bó hẹp trong cơ chế phân hoạch ranh giới kiến trúc đích như được dùng trong hầu hết các hệ thống ngày nay. Các thành phần của mô hình thiết kế hợp lệ có thể đóng kết với các môđun của các công nghệ khác nhau.

Giao tiếp và tổng hợp giao tiếp là những vấn đề chính của quá trình chỉ định công nghệ. Thực ra, 2 bước phân hoạch và tích hợp bây giờ được thay thế bởi thủ tục tinh chỉnh. Thủ tục này dựa vào mô hình hành vi trừu tượng và quá trình tổng hợp giao tiếp. Đây là vấn đề không dễ dàng. Tuy nhiên công việc thiết kế có thể truy nguyên dễ dàng hơn trong trường hợp các đối tượng hợp lệ nhỏ được gắn kết và được tổng hợp thông qua các giao tiếp biết trước. Những giao tiếp này được tạo ra dựa trên mối tương quan giữa các thành phần trong mô hình đã tinh chỉnh. Tùy thuộc vào công nghệ được lựa chọn, người ta sẽ chọn phương án trao đổi tín hiệu, ngắt quãng hoặc các phương tiện đồng bộ khác. Trong phương pháp luận này, các khả năng thiết kế khác có thể được đánh giá theo nhiều tiêu chuẩn khác nhau, ví dụ, việc gán các mô hình hành vi hoặc chức năng vào các mođun chấp hành (HW, SW, giao tiếp). Bước chỉ định công nghệ này được điều chỉnh bởi kết quả ước lượng hiệu năng có được từ các bước mô phỏng trước đó.

Trong phần kế tiếp, chúng tôi đề nghị về việc codesign nên được gộp vào trong một môi trường lớn hơn. Môi trường này sẽ tích hợp các phương pháp và các khái niệm nâng cao từ phạm trù rộng lớn của CAD.

3. CODESIGN VỚI SỰ TRỢ GIÚP CỦA MÁY TÍNH

Trong vài năm gần đây, chúng ta đã chứng kiến sự gia tăng nhanh chóng của các phần mềm tự động hóa thiết kế (EDA tools) và các hệ thống tích hợp các phần mềm này, thường được gọi là môi trường thiết kế với sự trợ giúp của máy tính (CAD framework) ([15]). CFI (CAD Framework Initiative) xem môi trường này như một tập hợp các chương trình/mođun mở rộng dùng để phát triển một hệ thống CAD đồng nhất ([14]). [16] định nghĩa như sau: "Môi trường thiết kế với sự trợ giúp của máy tính là một cơ sở hạ tầng phần mềm, cung cấp một môi trường hoạt động chung cho các công cụ CAD. Một môi trường phải cho phép người dùng chạy và quản lý các công cụ; tạo, tổ chức và quản lý dữ liệu; quan sát dạng đồ họa toàn bộ quá trình thiết kế; và thực hiện các nhiệm vụ quản lý thiết kế như quản lý cấu hình và phiên bản. Một số thành phần cốt yếu của một môi trường CAD có thể được liệt kê như: giao diện người dùng và đồ họa độc lập với hệ thống, trao đổi thông tin giữa các công cụ, quản lý quá trình và dữ liệu thiết kế, và các dịch vụ CSDL".

Các hệ thống CAD thiết kế vi mạch VLSI đang có và các hệ thống đang phát triển tích hợp nhiều thành phần môi trường, ví dụ như, điều khiển công cụ, quản lý cấu hình và phiên bản, … Tuy nhiên chúng chưa được thay đổi thích hợp để xử lý các đặc tính vốn có của codesign. Các hệ thống thiết kế hiện tại giải quyết vấn đề chỉ trong một lĩnh vực đơn lẻ mà chưa có cái tầm nhìn toàn cục. Có rất ít công cụ được thiết kế cho việc tương tác với môi trường hoặc các công cụ khác để đặc tả chéo HW/SW, phát triển, mô phỏng, tích hợp và kiểm tra.

Chúng tôi cho rằng các công cụ tính toán cao cấp có mức độ hiệu quả giới hạn, không có những quyết định thiết kế tin cậy để hỗ trợ cho phương pháp luận trong việc tạo nên những xử lý có hệ thống nhiều mục tiêu và ràng buộc tác động đến quá trình codesign. Do đó người ta yêu cầu các công cụ và kỹ thuật codesign phải được gộp lại theo một cách tiếp cận gọi là CASHE (Computer-Aided Software/Hardware Engineering).

Môi trường CASHE phải tích hợp các khái niệm cao cấp như quản lý công việc và qui trình thiết kế, các kỹ thuật đồng thời cho việc thiết kế kết hợp, hệ thống hỗ trợ quyết định (DSS) và các kỹ thuật hệ chuyên gia. Trong [17], người ta đã đưa ra khái niệm cho việc thiết kế mẫu thông qua các kỹ thuật mô phỏng dựa trên cơ sở tri thức. Cách tiếp cận này tập trung vào việc phát triển các mô hình mô phỏng với các đặc điểm của hệ thống cần thiết kế. Các mô hình được phát triển ở các mức trừu tượng và granularity khác nhau. Mô hình bao gồm các môđun tái sử dụng trong CSDL các mođun. Việc kiểm tra bổ sung CSDL mô phỏng, dùng để kiểm tra chéo việc tương thích mô hình với đặc tả, ràng buộc và yêu cầu của thiết kế. Môi trường đề nghị cũng cho phép

các kỹ thuật dựa trên hệ chuyên gia để tạo và đánh giá các giải pháp và chiến thuật thiết kế khác nhau.

4. KẾT LUẬN

Một trong những yếu tố mà quá trình thiết kế các hệ thống số phụ thuộc vào là mức độ hỗ trợ của các công cụ CAD. Phương pháp thiết kế đồng thời bằng cả HW và SW mang lại lợi ích cho việc thiết kế các thành phần số của hệ thống. Phương pháp này sẽ tận dụng sự hiệp lực của HW và SW trong việc tìm kiếm các giải pháp sử dụng tối ưu nhất công nghệ chế tạo hiện tại và khả năng của các thành phần HW và chương trình SW.

Hiện nay, sự hỗ trợ của các công cụ CAD cho việc thiết kế codesign vẫn còn yếu. Sự hỗ trợ này đang phát triển với nhịp độ ngày càng cao vì hiệu quả tiềm năng của nó khiến đó là một vùng hấp dẫn cho việc nghiên cứu khoa học cũng như các cơ hội kinh doanh. Trong khi các công cụ mô phỏng đã đạt được mức độ trưởng thành (có nhiều các chương trình đồng mô phỏng đang được thương mại hóa hiện nay), các giải thuật và kỹ thuật tổng hợp mức hệ thống còn vẫn trong giai đoạn nghiên cứu. Tuy nhiên, vẫn có một số sản phẩm thương mại được thiết kế với việc sử dụng quá trình tổng hợp mức hệ thống của các thành phần HW và quá trình biên dịch cho các mục tiêu khác nhau.

Tóm lại, HW/SW codesign là một lĩnh vực thiết kế rộng lớn do nó bao trùm các ứng dụng đa dạng, các kiểu thiết kế và công nghệ hiện thực khác nhau. [18] cho rằng các công cụ CAD ảnh hưởng đối với các thiết kế mức hệ thống sâu rộng hơn đối với các thiết kế vi mạch tích hợp.

TÀI LIỆU THAM KHẢO

[1] D. Harel et al., State-charts: A working environment for the development of complex reactive systems. In IEEE Trans. Software Engineering, 16(4), pp. 403-414, 1990.

[2] C.U. Smith, F.A. Geoffrey, J.L. Cuadrado, An architecture design and assessment system for HW/SW codesign, in Proceedings of DAC, pp. 417-424, June 1985.

[3] F. Vahid, A survey of behavioral level partitioning systems, University of California/Irvine, 1991.

[4] E. Lagnese and D. Thomas, Architectural partitioning for system-level synthesis of intergrated circuits, IEEE Trans. on CAD, 10(7), pp. 847-860, 1991.

[5] D. Gajski, Silicon compilation, Addison-Wesley, MA, 1988.

[6] R. Camposano, in High-level Synthesis, W. Wolf, Ed., Intl. series in Engineering and Computer science, Kluwer Academic Publishers, Boston, 1991.

[7] W. Birmingham, in Automating the design of cumputer systems: the MICON project, D. Siewiorek and A. Gupta, Eds., Jones and Barlett, Newyork, 1992.

[8] E. Barros and W. Rosenstiel, A method for HW/SW partitioning, COMP-EURO, 1992

[9] R.K. Gupta, C.N. Coelho and G.De Micheli, Synthesis and simulation of digital systems containing interacting HW and SW components, in proceedings of DAC, pp. 225-230, June 1992.

[10] D. Becker, R.K. Singh and S.G. Tell, An engineering environment for HW/SW cosimulation, in proceedings of DAC, pp. 129-134, June 1992.

[11] W. Billowitch, Simulation models support HW/SW integration, Computer Des., pp.31, 1988.

[12] L. Philipson, Multilevel design and verification of HW/SW systems, IEEE J. Solid State Circuits, 25(3), pp. 714-719, 1990.

[13] G. Estrin et al., SARA: Modeiling, Analysis and Simulation support for Design of concurrent systems, IEEE Trans. Software Eng., SE-12, pp. 293-311, 1986.

[14] A. Graham, CFI: Towards a Broad, Standard Framework, IEEE Spectrum, 29(11), p. 40, 1992.

[15] D. Harrison et al., Electronic CAD Framework, Proceedings of IEEE, 78(2), pp. 393-417, 1990.

[16] L. Maliniak, CAD Framework Ride a Rough Road to Success, Electronic Des., 40(16), p. 36, 1992.

[17] J. Rozenblit and B. Zeigler, Knowledge-based Simulation Design Methodology: A Flexible Test Architecture Application, Trans. Soc. Computer Simulation, 7(3), pp. 195-228, 1990.

[18] G. De Micheli, Hardware/Software codesign: Application Domains and Design Technologies, in HW/SW Codesign, G. De Micheli and M. Sami Eds., Kluwer Academic Publishers, Boston, 1995.

[19] D. Thomas, H. Schmit and J. Adams, A Model and Methodology for HW/SW Codesign, IEEE Design Test Computers, 10(3), pp. 6-15, 1993.