TẠP CHÍ KHOA HỌC & CÔNG NGHỆ ĐẠI HỌC DUY TÂN
DTU Journal of Science and Technology 04(41) (2020) 3-12
TRƯỜNG ĐẠI HỌC DUY TÂN
DUYTAN UNIVERSITY

Cấu trúc kiểm chứng thiết kế cho bộ cộng toàn phần 4-bit dựa trên phương pháp xác minh phổ quát

UVM-based verification architecture of a 4-bit full adder

Nguyễn Xuân Tiến[a,b*], Tạ Quốc Việt[a,b], Trần Lê Thăng Đồng[a,b]
Xuan Tien Nguyen[a,b*], Quoc Viet Ta[a,b], Le Thang Dong Tran[a,b]

[a]*Phòng Nghiên cứu Điện - Điện tử với Doanh nghiệp, Đại học Duy Tân, Đà Nẵng, Việt Nam*
[b]*Khoa Điện - Điện tử, Đại học Duy Tân, Đà Nẵng, Việt Nam*
[a]*Laboratory for Corporate Electrical - Engineering Research, Duy Tan University, Danang, 550000, Vietnam*
[b]*Faculty of Electrical - Electronics Engineering, Duy Tan University, Da Nang, 550000, Vietnam*

(Ngày nhận bài: 25/3/2020, ngày phản biện xong: 08/4/2020, ngày chấp nhận đăng: 15/8/2020)

Tóm tắt

Việc xác minh chức năng thiết kế là một yêu cầu bắt buộc phải có và chiếm đến gần 70 - 80% thời gian trong chu kỳ của một thiết kế bất kì. Những phương pháp xác minh hiện nay bằng cách kiểm tra trực tiếp các thiết kế thường tốn nhiều thời gian, có độ tin cậy thấp và khá nhàm chán. Bên cạnh đó, nó khó bao quát được hết tất cả các trường hợp cần phải xác minh. Bài báo này trình bày một cấu trúc kiểm chứng thiết kế cho bộ cộng toàn phần 4-bit dựa trên phương pháp xác minh phổ quát (UVM) sử dụng ngôn ngữ System Verilog. Việc kết hợp những ưu điểm của UVM trong cấu trúc được đề xuất này cùng với System Verilog giúp xây dựng môi trường xác minh mà ở đó các biến ngõ vào được thiết lập ngẫu nhiên giúp giảm thời gian xây dựng testbench. Ngoài ra, việc tận dụng ngôn ngữ System Verilog để tạo ra các ma trận nhằm đánh giá độ bao phủ các trường hợp cần xác minh giúp ích rất nhiều trong việc gia tăng độ tin cậy trong thiết kế. Kết quả mô phỏng cho thấy độ bao phủ này lên đến 99.3%. Hơn nữa, cấu trúc được đề xuất này có thể tái sử dụng hoặc mở rộng thêm trong việc kiểm tra các thiết kế SoC khác, rút ngắn được thời gian kiểm chứng.

Từ khóa: UVM; Xác minh thiết kế; SystemVerilog; Bộ cộng toàn phần 4-bit; Độ bao phủ chức năng.

Abstract

Chip functional verification is a requirement and takes almost 70 - 80 percent of the project cycle time for any SoC designs. Traditional and present chip verification methods using directed-testing are time-consuming, low reliability and tedious. Besides, these methods hardly cover almost all operating conditions that need verification. In this work, an efficient UVM-based verification architecture for a 4-bit full adder model using SystemVerilog is presented. The proposed verification architecture used coverage metrics and random stimulus to achieve a result of 99.3 percent functional coverage. Moreover, this method can be reusable and scalable in other SoC verifications, which in turn helps reduce verification time.

Keywords: UVM; design verification; SystemVerilog; full adder 4-bit; functional coverage.

1. Giới thiệu

Xác minh chức năng thiết kế là quá trình xem xét các thiết kế đã đáp ứng các yêu cầu ban đầu được đưa ra hay chưa; và cần rất nhiều các trường hợp kiểm tra được tạo ra để kiểm chứng các thiết kế. Hiện nay, các thiết kế ngày

**Corresponding Author:* Nguyễn Xuân Tiến; Laboratory for Corporate Electrical – Engineering Research, Duy Tan University, Danang, 550000, Vietnam; Faculty of Electrical - Electronics Engineering, Duy Tan University, Da Nang, 550000, Vietnam.
Email: nguyenxuantien7@dtu.edu.vn

càng trở nên phức tạp khiến cho việc sử dụng các phương pháp kiểm tra trực tiếp theo truyền thống từ trước đến nay không còn hữu dụng nữa. Thêm vào đó, quá trình xác minh này chiếm đến 70 - 80% chu kỳ thiết kế [1]. Việc xác minh với các biến ngõ vào ngẫu nhiên thường chỉ bao phủ đến 80% các trường hợp kiểm tra [1].

Xác minh một thiết kế thường được tiến hành bằng nhiều ngôn ngữ và phương thức khác nhau. Cách thuận tiện và phổ biến nhất là viết các trường hợp kiểm tra bằng VHDL hay Verilog. Nhược điểm chính của Verilog và VHDL được trình bày chi tiết ở bài báo [2]. Một cách tóm tắt, cả Verilog và VHDL thiếu những đặc điểm để đáp ứng các loại dữ liệu bậc cao và các phương thức lập trình hướng đối tượng; thiếu việc phân tích độ bao phủ các trường hợp kiểm tra; và thiếu các ràng buộc cho biến ngõ vào. Do đó, System Verilog ra đời hướng tới việc cung cấp các giải pháp cho những hạn chế nêu trên của Verilog và VHDL. System Verilog với hơn 200 từ khóa đủ để tiến hành các tác vụ xác minh thiết kế phức tạp. Nhưng bên cạnh đó, nó còn một số hạn chế về mặt thực tiễn. Ví dụ như code được viết trên công cụ của một nền tảng này thì không thể chạy ở công cụ của nền tảng khác; nên việc sử dụng lại code là một vấn đề lớn; trong khi chu kỳ thời gian yêu cầu cho một thiết kế bất kỳ bị giới hạn.

Phương pháp xác minh phổ quát (Universal Verification Methodology - UVM) - là một trong những cách thức xác minh thiết kế số hiện nay, đã tận dụng những ưu điểm của SystemVerilog về độ đa dạng và cách thức lập trình bậc cao. UVM sử dụng ngôn ngữ SystemVerilog xây dựng lên cách thức xác minh các thiết kế bằng cách cung cấp những thư viện lớp cơ bản cho việc cấu thành và sắp xếp các trường hợp kiểm tra [3] [4] [5] [6]. Phương pháp này phác thảo các quy tắc và quy trình để thực hiện việc xác minh một cách có hệ thống. Những ưu điểm chính của UVM có thể kể đến như là: hỗ trợ thư viện lớp cơ bản trên phạm vi rộng; được phát triển dựa trên tiêu chuẩn IEEE 1800.2 - 2017; có thể thực hiện việc xác minh các thiết kế với các biến ngõ vào ngẫu nhiên có ràng buộc cùng với việc tiến hành kiểm chứng độ bao phủ; được hỗ trợ chạy mô phỏng bởi các phần mềm khác nhau; và được liên tục cập nhật bởi Accellera [7]. Bên cạnh đó, UVM được sử dụng tương thích với tất cả các nhà cung cấp công cụ chính. Điều này đảm bảo cho các kỹ sư có thể chia sẻ chung một cách thức để xác minh các thiết kế của mình, tiết kiệm được thời gian chuyển đổi giữa các ngôn ngữ và công cụ thiết kế. Do đó, có thể nói rằng UVM giúp tạo ra các testbench đủ khả năng xác minh các thiết kế, linh động, có thể tái sử dụng và mở rộng.

Trong các thiết kế số hiện nay, bộ cộng toàn phần là một trong những thành phần cơ bản và quan trọng nhất của CPU. Nó có mối quan hệ chặt chẽ với đơn vị logic số học (ALU), đơn vị dấu phẩy động và đơn vị tạo địa chỉ. Nó thường được sử dụng như một khối xây dựng chính trong các mạch chức năng số học, chẳng hạn như bộ cộng, bộ trừ, bộ nhân và bộ chia. Trong đa số các nghiên cứu hiện nay, sau khi thiết kế xong thì phương pháp để xác minh chức năng của bộ cộng toàn phần thường là sử dụng ngôn ngữ Verilog mô tả lại thiết kế và sử dụng phần mềm Xilink mô phỏng một vài trường hợp [8] hay sử dụng công cụ HSPICE [9] [10] hoặc công cụ QCADesigner [11] và chỉ mô phỏng các trường hợp cần thiết. Bên cạnh đó, một vài công trình nghiên cứu cũng đề xuất phương pháp xác minh bộ cộng bằng cách sử dụng các định lý và công thức toán học [12], nhưng phương pháp này thường chỉ áp dụng cho các thiết kế đơn giản và không mang tính kế thừa cho các thiết kế khác.

Bài báo này phân tích các thành phần của UVM và ứng dụng nó trong việc tạo ra một môi trường xác minh cho bộ cộng toàn phần 4-bit (Full Adder 4-bit hay viết tắt là F.A. 4-bit) sử dụng ngôn ngữ System Verilog. Cấu trúc của

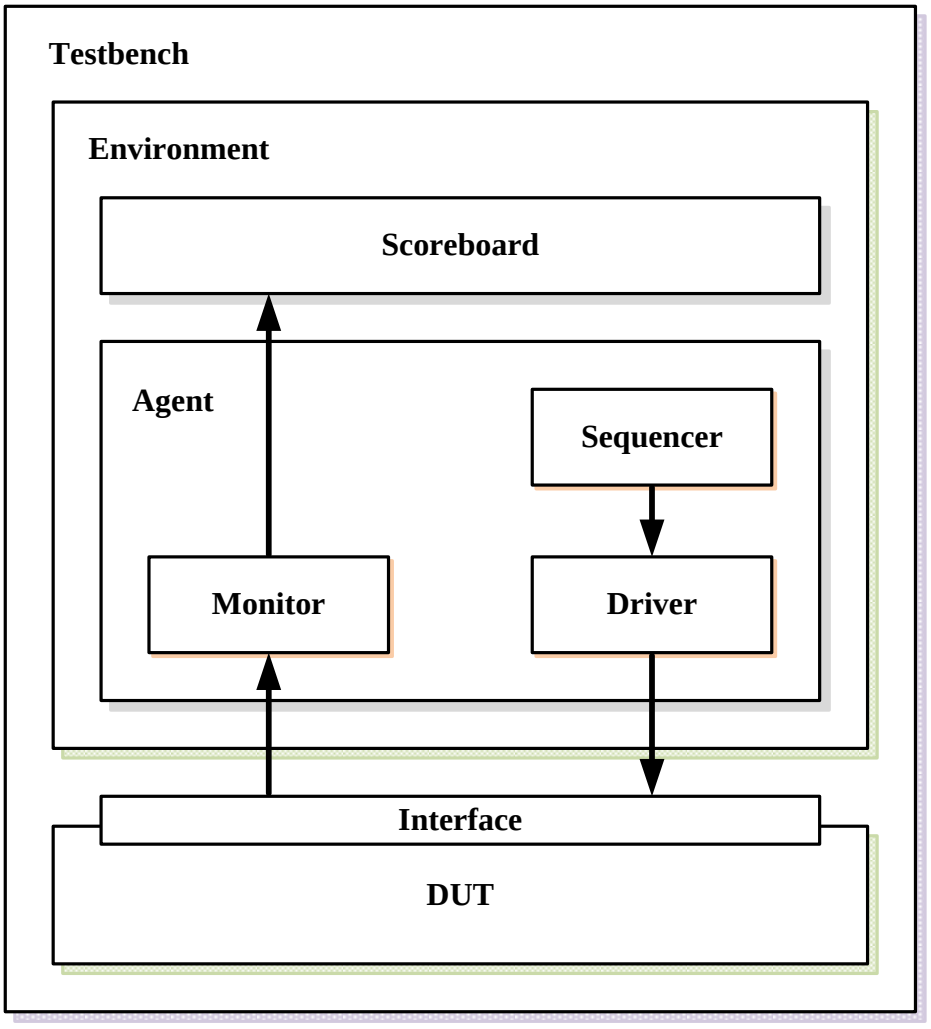
bộ xác minh thiết kế được đề xuất này tận dụng đầy đủ những ưu điểm của UVM, đem lại quá trình xác minh thiết kế chip nhanh chóng và hiệu quả. Cần phải nhấn mạnh rằng, cấu trúc được đề xuất có thể được điều chỉnh để xác minh ở mức khối cho bất cứ thiết kế tín hiệu số nào.

Phần còn lại của bài báo được sắp xếp như sau: phần 2 liệt kê các thành phần cơ bản và các giai đoạn chạy mô phỏng của một cấu trúc UVM điển hình; phần 3 mô tả chi tiết về cấu trúc xác minh được đề xuất cho F.A. 4-bit; phần 4 trình bày kết quả dạng sóng mô phỏng và độ bao phủ của các trường hợp cần kiểm tra; phần 5 đưa ra kết luận và hướng phát triển của đề tài.

2. Phương pháp xác minh phổ quát (UVM)

2.1. Các thành phần cơ bản của UVM

Hình 1 trình bày các thành phần cơ bản của một cấu trúc xác minh thiết kế UVM điển hình. Ở lớp ngoài cùng (top level), mô-đun *Testbench* có nhiệm vụ kết nối khối DUT với các thành phần của môi trường xác minh thông qua lớp giao diện. Sau đây là mô tả lần lượt của các thành phần con.



Hình 1: Cấu trúc testbench UVM

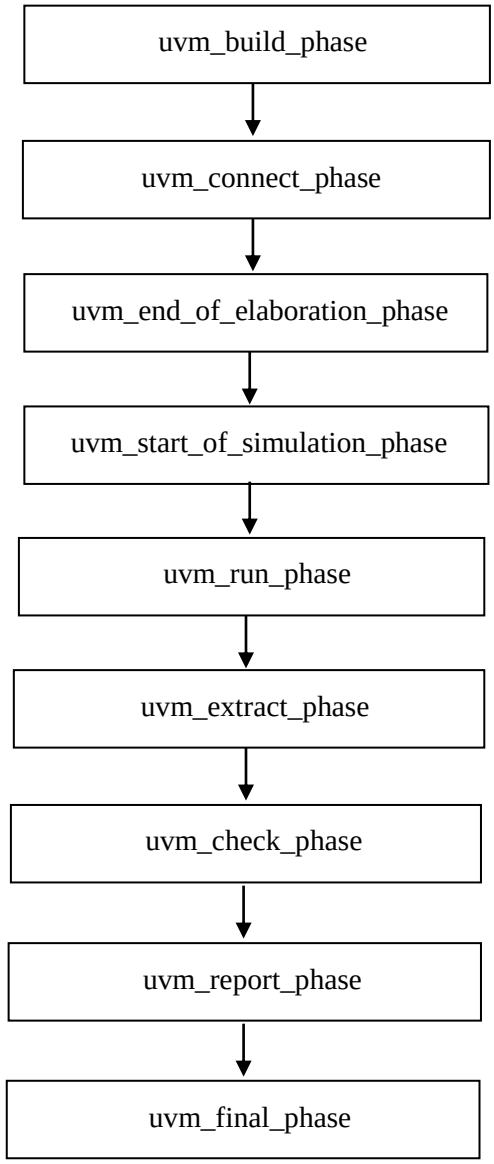
Environment: lớp môi trường.
Scoreboard: bảng kiểm tra kết quả xác minh.
Agent: tác nhân xác minh.
Sequencer: bộ sắp xếp trình tự.
Driver: bộ điều hướng.
Monitor: bảng giám sát.
Interface: lớp giao diện.

- **DUT - Device Under Test**: thiết kế cần xác minh.
- **Environment**: đây là lớp chính của cấu trúc xác minh. Nó kết nối một hay nhiều tác nhân xác minh với bảng kiểm tra kết quả xác minh. Hơn nữa, nó có thể bao gồm các thành phần khác như là khối giám sát và kiểm kê (*Checker*), và khối dự đoán ngõ ra của thiết kế (*Predictor*).
- **Agent**: có nhiệm vụ tạo ra các trường hợp kiểm tra khác nhau cho việc xác minh thiết kế và truyền chúng đến DUT thông qua lớp giao diện. Một tác nhân xác minh thông thường bao gồm bộ sắp xếp trình tự, bộ điều hướng và bảng giám sát, cùng với các thông số để cấu hình nên tác nhân xác minh.
- **Sequencer**: có nhiệm vụ chuyển các chuỗi ngõ vào tới bộ điều hướng bất cứ khi nào bộ điều hướng này cần.
- **Driver**: có nhiệm vụ liên tục nhận các kết quả từ bộ sắp xếp trình tự, và chuyển đổi chúng thành các giá trị tương ứng bậc cao hơn mà DUT có thể hiểu được.
- **Monitor**: nhận lấy tín hiệu ngõ ra của DUT thông qua lớp giao diện và chuyển chúng thành các giá trị tương ứng bậc thấp hơn. Tiếp theo, bảng giám sát gởi các kết quả này tới bảng kiểm tra kết quả xác minh.
- **Scoreboard**: tại đây sẽ kiểm tra các hành vi của DUT như mong muốn hay không, bằng cách so sánh các đáp ứng thực tế của DUT (là các kết quả nhận được từ bảng giám sát) với các giá trị mong đợi (được lấy từ lớp khối dự đoán ngõ ra của thiết kế).

2.2. Các giai đoạn chạy mô phỏng của UVM

Các thành phần nêu trên của UVM sẽ tiến hành hoạt động theo một trật tự nhất định. Trật tự này được quy định bởi các giai đoạn đã được định nghĩa trong UVM. UVM có 9 giai đoạn chủ yếu, hoạt động theo một cơ chế đồng bộ trong suốt quá trình chạy mô phỏng. Nghĩa là các thành phần UVM phải thực hiện xong giai đoạn hiện tại trước khi chuyển đến giai đoạn kế tiếp. Hình 2 trình bày chi tiết các giai đoạn chạy mô phỏng của UVM.

- **build_phase** *(giai đoạn khởi tạo)*: khởi tạo các thành phần và các đối tượng trong UVM.
- **connect_phase** *(giai đoạn kết nối)*: kết nối các thành phần và các đối tượng UVM từ giai đoạn trước đó lại với nhau.
- **end_of_elaboration_phase** *(giai đoạn cấu hình)*: tiến hành cấu hình cho các thành phần UVM sau khi kết nối nếu cần thiết.
- **start_of_simulation_phase** *(giai đoạn tiền mô phỏng)*: kích hoạt các giá trị ban đầu cho các thành phần UVM trước khi chạy mô phỏng hoặc phát đi các thông báo và thông tin về cấu trúc liên kết nếu cần.
- **run_phase** *(giai đoạn thực thi)*: tiến hành mô phỏng bằng việc tạo ra các giao dịch để gởi tới các thiết kế cần xác minh.
- **extract_phase** *(giai đoạn trích xuất)*: thu thập tất cả các thông tin cần thiết cho việc so sánh ở các giai đoạn tiếp theo.
- **check_phase** *(giai đoạn kiểm tra)*: so sánh và kiểm tra các kết quả thực tế nhận được từ thiết kế cần xác minh với các kết quả mong đợi từ khối dự đoán.
- **report_phase** *(giai đoạn báo cáo)*: trả về kết quả PASS / FAIL sau khi so sánh và kiểm tra ở các giai đoạn trên.
- **final_phase** *(giai đoạn hoàn tất)*: thực hiện một số xử lý, thao tác cuối cùng nếu có trước khi kết thúc mô phỏng.

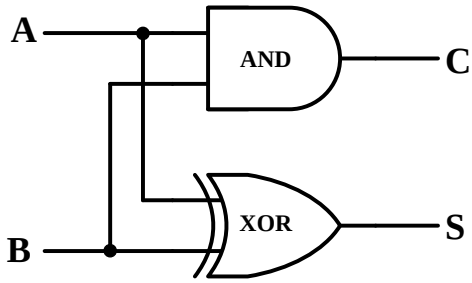


Hình 2: Các giai đoạn chạy mô phỏng của UVM

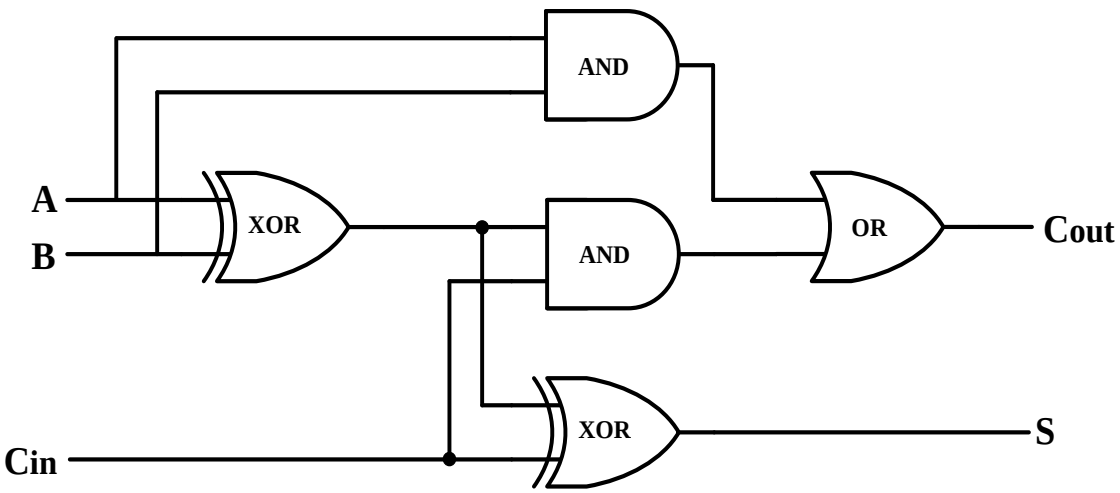
3. Cấu trúc xác minh cho bộ cộng toàn phần 4-bit được đề xuất

3.1. Bộ cộng toàn phần 4-bit (F.A. 4-bit)

Đầu tiên, xét một bộ cộng toàn phần 1-bit cơ bản bao gồm hai bộ cộng bán phần (Hình 3) được nối với nhau bằng một cổng OR. Cấu trúc bộ cộng toàn phần này được cho ở Hình 4. Thiết kế của bộ cộng này sử dụng các cổng logic đơn giản như AND, XOR and OR.



Hình 3: Bộ cộng bán phần



Hình 4: Bộ cộng toàn phần

Bộ cộng toàn phần 1-bit thực hiện việc cộng ba số nhị phân A, B và Cin. Trong đó A và B là hai số cộng 1-bit và Cin là phần dư của phép cộng trước. Mạch cho ra kết quả là tổng S và số dư Cout. Kết quả xác minh chức năng của bộ cộng này phải trùng khớp với bảng sự thật sau:

Bảng 1: Bảng sự thật của một bộ cộng toàn phần

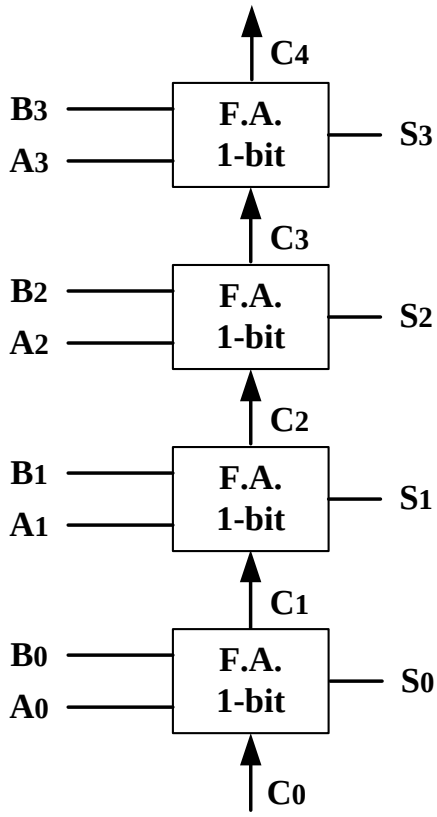
Cin	*A*	*B*	*S*	*Cout*
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Công thức ngõ ra của một bộ cộng toàn phần 1-bit được rút ra từ bảng sự thật như sau:

$$S = (A \oplus B) \oplus Cin \quad (1)$$

$$Cout = A \cdot B + (A \oplus B) \cdot Cin \quad (2)$$

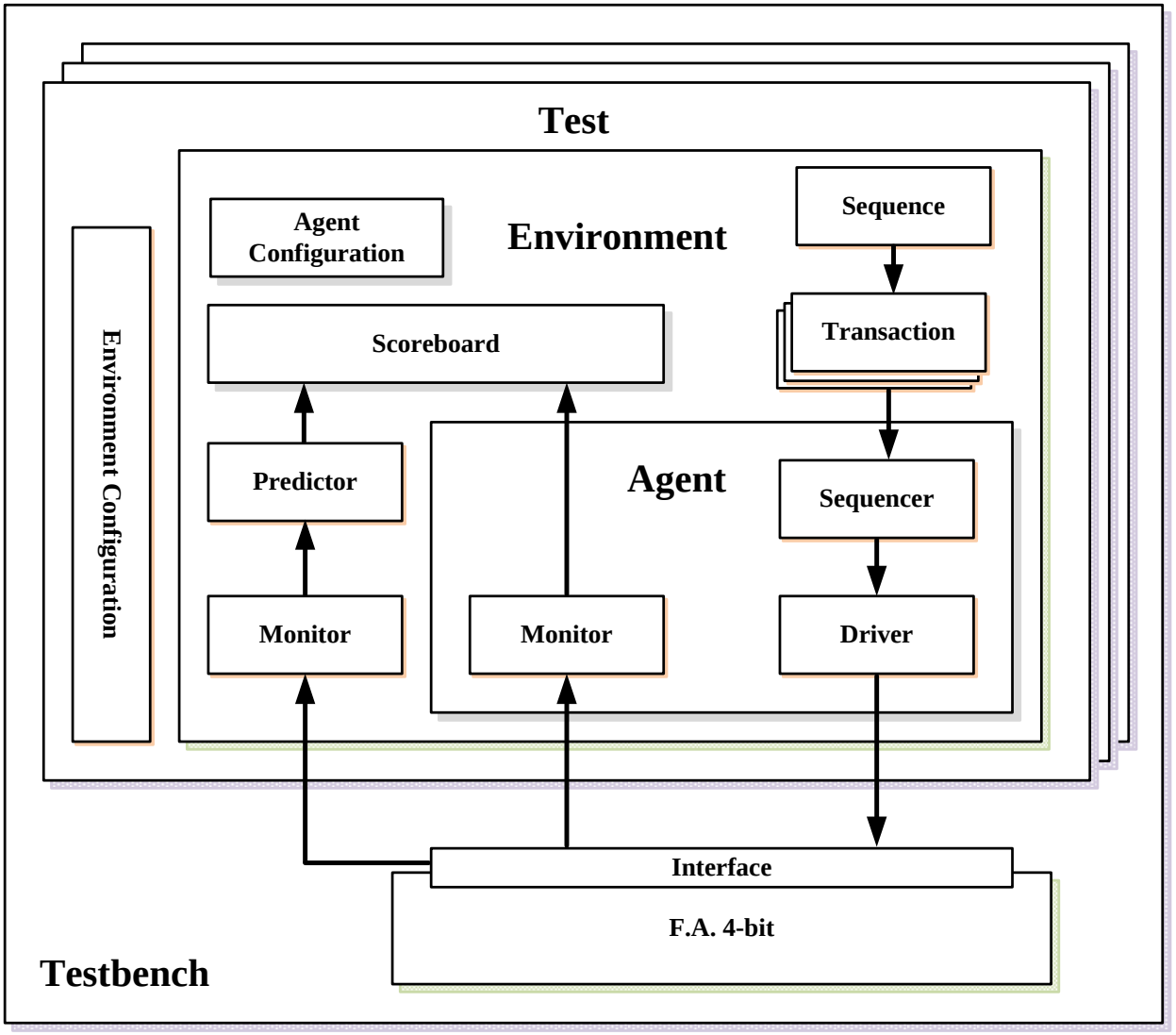
Hình 5 trình bày sơ đồ khối của một F.A. 4-bit; trong đó có bốn bộ cộng toàn phần 1-bit (F.A. 1-bit) được mắc nối tiếp nhau. Mỗi F.A. 1-bit có ba đầu vào tương ứng là A_x, B_x, và C_x, và hai ngõ ra là S_x và C_{x+1} (với x từ 0 tới 3). Như vậy, F.A. 4-bit có các ngõ vào là hai số cộng 4-bit ($A_3A_2A_1A_0$ & $B_3B_2B_1B_0$) và số nhớ từ phép tính trước C_0. Các ngõ ra của F.A. 4-bit lần lượt là số dư C_4 và tổng $S_3S_2S_1S_0$.



Hình 5: Sơ đồ khối của một FA 4-bit

3.2. Cấu trúc xác minh được đề xuất

Hình 6 trình bày cấu trúc xác minh dựa trên UVM được đề xuất cho thiết kế F.A. 4-bit.



Hình 6: Cấu trúc xác minh dựa trên UVM cho FA 4-bit được đề xuất

Test: UVM mức đỉnh.

Environment Configuration: cấu hình môi trường.

Agent Configuration: cấu hình tác nhân xác minh.

Sequence: chuỗi ngõ vào.

Transaction: giao dịch

Predictor: lớp dự đoán kết quả

Mô-đun *Testbench* khai báo đối tượng là F.A. 4-bit, lớp giao diện và lớp UVM mức đỉnh. Mỗi lớp UVM mức đỉnh sẽ tương ứng với mỗi trường hợp kiểm tra khác nhau được tiến hành để xác minh F.A. 4-bit. Lớp UVM mức đỉnh tiến hành khai báo cho lớp môi trường và cấu hình cho quá trình xác minh thông qua các lớp cấu hình môi trường. Lớp môi trường này bao gồm một lớp tác nhân xác minh đơn lẻ, cùng với một lớp bảng kiểm tra kết quả xác minh, một lớp dự đoán kết quả để dự đoán hành vi của F.A. 4-bit dùng trong việc so sánh, một lớp giám sát ngoại vi, và lớp chứa những cấu hình cho những tác nhân xác minh khác nhau. Lớp tác nhân xác minh được đề xuất bao gồm các lớp như đã giới thiệu ở phần 2 (bộ sắp xếp trình tự, bộ điều hướng, và bảng giám sát).

Các chuỗi ngõ vào sẽ cấu thành các giao dịch. Khi bộ điều hướng cần, các giao dịch này sẽ được truyền từ bộ sắp xếp trình tự đến và tiến hành biên dịch sang các giá trị tương ứng

bậc cao hơn mà F.A. 4-bit có thể hiểu, rồi sau đó bắt đầu tiến hành các quá trình tính toán bên trong nó.

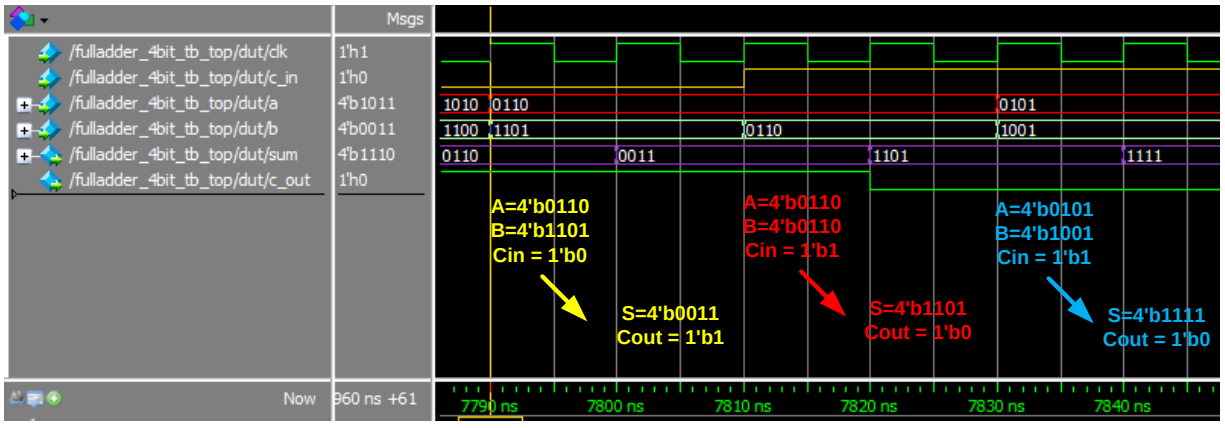
Trong cấu trúc testbench được đề xuất này, bảng kiểm tra kết quả xác minh sẽ kiểm tra hành vi của F.A. 4-bit có đúng với yêu cầu được đưa ra hay không. Bảng kiểm tra kết quả xác minh so sánh hồi đáp của F.A. 4-bit được lấy từ lớp giám sát bên trong lớp tác nhân xác minh với giá trị hoạt động mong đợi của F.A. 4-bit từ lớp dự đoán kết quả. Có nghĩa là trong trường hợp này, dữ liệu của lớp dự đoán kết quả đóng vai trò như một mô hình tham khảo. Nếu kết quả so sánh giống nhau thì bảng kiểm tra kết quả này sẽ đưa ra tín hiệu báo F.A. 4-bit hoạt động như yêu cầu – là PASS và ngược lại – là FAIL. Trong lớp giao diện, các tín hiệu ngõ vào và ngõ ra được khai báo, cho phép việc giao tiếp hiệu quả giữa F.A. 4-bit và các lớp bên trong *Testbench*.

3.3. Độ bao phủ chức năng

Độ bao phủ chức năng là một phần thiết yếu của việc xác minh thiết kế. Nó là một ma trận được định nghĩa bởi người kiểm tra, dùng để xác minh kế hoạch kiểm tra đạt được bao nhiêu phần trăm [13]. Chất lượng của việc xác minh thiết kế phụ thuộc vào chất lượng của kế hoạch kiểm tra. Về mặt bản chất, độ bao phủ sẽ trả lời cho câu hỏi "các trường hợp kiểm thử đưa vào xác minh thiết kế đã đủ ngẫu nhiên hay chưa?" Trong cấu trúc xác minh được đề xuất ở bài báo này, mô hình bao phủ được xây dựng từ tập hợp những điểm cần bao phủ (*coverpoint*). Mỗi điểm cần bao phủ liên quan đến một chức năng hoặc đặc điểm của F.A. 4-bit cần xác minh. Cụ thể trong trường hợp này, mỗi điểm cần bao phủ tương ứng với từng ngõ vào của bộ cộng toàn phần. Thông thường một thiết kế được đánh giá là tốt nếu có kết quả độ bao phủ đạt từ 95% trở lên.

4. Kết quả mô phỏng

Cấu trúc xác minh dựa trên UVM được đề xuất cho F.A. 4-bit được tiến hành và mô phỏng sử dụng phần mềm QuestaSim. SystemVerilog cùng với phương pháp UVM được sử dụng cho việc thiết kế cấu trúc xác minh này. Bên cạnh đó, SystemVerilog cũng được dùng cho việc mô tả F.A. 4-bit. Rất nhiều trường hợp kiểm tra được tạo ra để xác minh chức năng của F.A. 4-bit này một cách đầy đủ nhất. *Testbench* được thiết kế với mục đích xác nhận hoạt động chính xác của mô hình F.A. 4-bit dựa trên dữ liệu được cung cấp ở ba ngõ vào, và kiểm chứng kết quả ở hai ngõ ra.

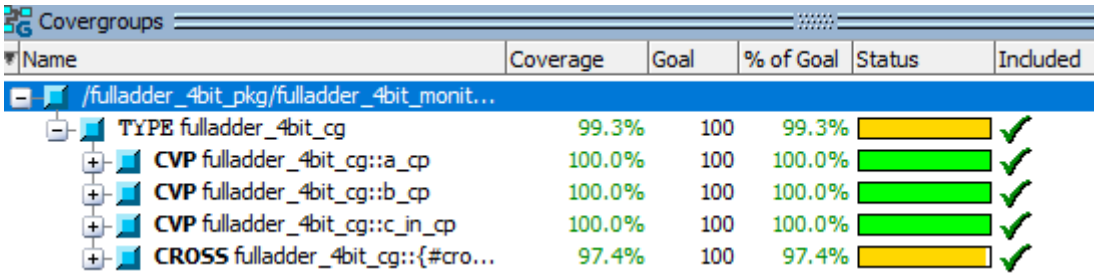


Hình 7: Kết quả dạng sóng mô phỏng

Hình 7 là dạng sóng mô phỏng kết quả của cấu trúc xác minh được đề xuất. Có tất cả 4097 tổ hợp ngõ vào được tạo ra, một vài trong số đó sẽ bị lặp lại; đây là kết quả của quá trình tạo ngõ vào ngẫu nhiên. Ba trong số các kết quả mô phỏng được hiển thị cho thấy cấu trúc xác minh được đề xuất này hoạt động chính xác và đạt được hiệu suất xác minh cao cho F.A. 4-bit.

Đối với phương pháp xác minh truyền thống, kỹ sư thẩm định thiết kế sẽ phải trực tiếp nhập từng giá trị ngõ vào. Ví dụ trong trường hợp này, các ngõ vào A và B có 4 bits nghĩa là có 16 giá trị khác nhau được tạo ra tương ứng cho từng ngõ vào. Ngõ vào Cin có 1-bit, nghĩa là tương ứng với 2 giá trị khác nhau được tạo ra. Tổng cộng, kỹ sư thẩm định thiết kế sẽ phải trực tiếp nhập 512 trường hợp khác nhau để chắc chắn rằng thiết kế hoạt động bình thường. So với việc dùng cấu trúc xác minh dựa trên UVM, thì công việc kiểm tra thủ công này tốn nhiều thời gian hơn và trở nên phức tạp hơn đối với những thiết kế hỗn hợp và có nhiều ngõ vào cần xác minh.

Bên cạnh đó, các phương pháp xác minh truyền thống không có một cơ sở nào để tự động kiểm tra xem các trường hợp đưa vào kiểm tra đã đầy đủ hay chưa. Tuy nhiên, phương pháp xác minh được đề xuất trong bài báo này đã tận dụng chức năng của SystemVerilog để phân tích độ bao phủ cho các trường hợp đưa vào kiểm tra.



Hình 8: Phân tích độ bao phủ cho các trường hợp kiểm tra

Phân tích độ bao phủ ở Hình 8 cho thấy rằng, điểm bao phủ cho tất cả ngõ vào của F.A. 4-bit đạt hoàn toàn 100%. Sự giới hạn của thời gian mô phỏng khiến cho việc kết hợp của cả ba ngõ vào này chưa hoàn toàn bao phủ hết tất cả các trường hợp. Kết quả độ bao phủ đạt 99.3% (tốt hơn rất nhiều so với tiêu chuẩn 95%) đã khẳng định cho hiệu quả xác minh cao của cấu trúc xác minh cho F.A. 4-bit được đề xuất này.

5. Kết luận

Bài báo đã trình bày cấu trúc xác minh chức năng dựa trên UVM cho bộ cộng toàn phần 4-bit. Khi so sánh với các phương pháp xác minh truyền thống, cấu trúc xác minh này đã khai thác được những ưu điểm vốn có của UVM và SystemVerilog để xây dựng nên một phương pháp mà có thể bao quát hầu hết các trường hợp cần kiểm tra, đảm bảo rằng thiết kế hoạt động chính xác nhất nhờ dựa vào tổ hợp các ngõ vào được cung cấp ngẫu nhiên và kiểm tra độ bao phủ. Hơn nữa, trong tương lai, cấu trúc xác minh này có thể được mở rộng dễ dàng để kiểm tra chức năng cho các bộ cộng có số bit ngõ vào cao hơn như 8-bit, 16-bit, 32-bit, 64-bit,..hoặc là các thiết kế số khác mà không cần phải xây dựng bộ xác minh từ đầu.

Tài liệu tham khảo

[1] T. M. Pavithran and R. Bhakthavatchalu, "UVM based testbench architecture for logic sub-system verification," 2017 *International Conference on Technological Advancements in Power and Energy (TAP Energy)*, Kollam, 2017, pp. 1-5.

[2] J.Bergeron, "Writingtestbenchesusingsystemverilog," *www.Verificationguild.com*, 2006, pp. 24.

[3] K. Salah, "A UVM-based smart functional verification platform: Concepts, pros, cons, and opportunities," 2014 *9th International Design and Test Symposium (IDT)*, Algiers, 2014, pp. 94-99.

[4] Ron Vogelsong, Ahmed Hussein Osman, Moustafa Mohamed, "Practical RNM with SystemVerilog", *CDNLive 2015*, 2015.

[5] Walter Hartong and Scott Cranston, "Real Valued Modeling for Mixed Signal Simulation", *Cadence*, 2009.

[6] Sathishkumar Balasubramanian and Pete Hardee, "Solutions for MixedSignal SoC Verification Using Real Number Models", *Cadence Design Systems*, 2013.

[7] Accellera, *Universal Verification Methodology (UVM) 1.1 User's Guide*, 2011.

[8] K. A. K. Maurya, Y. R. Lakshmanna, K. B. Sindhuri and N. U. Kumar, "Design and implementation of 32-bit adders using various full adders," 2017 *Innovations in Power and Advanced Computing Technologies (i-PACT)*, Vellore, 2017, pp. 1-6.

[9] A. K. Yadav, B. P. Shrivatava and A. K. Dadoriya, "Low power high speed 1-bit full adder circuit design at 45nm CMOS technology," 2017 *International Conference on Recent Innovations in Signal processing and Embedded Systems (RISE)*, Bhopal, 2017, pp. 427-432.

[10] M. Yang and E. Oruklu, "Full Adder Circuit Design Using Lateral Gate-All-Around (LGAA) FETs Based on BSIM-CMG Mode," 2018 *IEEE 61st International Midwest Symposium on Circuits and Systems (MWSCAS)*, Windsor, ON, Canada, 2018, pp. 420-423.

[11] B. Ramesh and M. A. Rani, "Implementation of parallel adders using area efficient quantum dot cellular automata full adder," 2016 *10th International Conference on Intelligent Systems and Control (ISCO)*, Coimbatore, 2016, pp. 1-5.

[12] N. Zhang and Z. Duan, "Verification of Hardware Designs: A Case Study," 2011 *First ACIS/JNU International Conference on Computers, Networks, Systems and Industrial Engineering*, Jeju Island, 2011, pp. 198-203.

[13] M.F. S. Oliveira, F. Haedicke, R. Drechsler, C. Kuznik, H.M. Le, W. Ecker, W. Mueller, D. Große, V. Esen "The System Verification Methodology for Advanced TLM Verification " *ISSS*, 2012.