AMBA AXI4 LITE SoC Integration and Simulation with EDA Tool

Mục lục

[1. AXI4-Lite 2](#_Toc168662735)

[2. Thiết kế module AXI-master 2](#_Toc168662736)

[a. Write address channel 4](#_Toc168662737)

[b. Write data channel 4](#_Toc168662738)

[c. Response channel 5](#_Toc168662739)

[3. Thiết kế module AXI-slave 6](#_Toc168662740)

[a. Read address channel 7](#_Toc168662741)

[b. Read data channel 7](#_Toc168662742)

[4. Thiết kế bộ Arithmetic Logic Unit (ALU) 9](#_Toc168662743)

[5. Thiết kế module AXI\_top 10](#_Toc168662744)

[6. Thiết kế testbench và kiểm tra kết quả mô phỏng 12](#_Toc168662745)

[a. Thiết kế testbench 12](#_Toc168662746)

[b. Kiểm tra kết qua mô phỏng 13](#_Toc168662747)

[7. Kết luận 14](#_Toc168662748)

Danh mục hình:

[Hình 1. Kiến trúc Interconnect của AXI4-Lite 2](#_Toc168662754)

[Hình 2. Kiến trúc Write operation của AXI4-Lite 3](#_Toc168662755)

[Hình 3. Write address channel 4](#_Toc168662756)

[Hình 4. Write data channel 5](#_Toc168662757)

[Hình 5. Response channel 5](#_Toc168662758)

[Hình 6. Kiến trúc Read operation của AXI4-Lite 6](#_Toc168662759)

[Hình 7. Read address channel 7](#_Toc168662760)

[Hình 8. Read data channel 8](#_Toc168662761)

[Hình 9. ALU control signal 9](#_Toc168662762)

[Hình 10. Module ALU 10](#_Toc168662763)

[Hình 11. Kiến trúc của AXI4-Lite kết nối với bộ ALU 11](#_Toc168662764)

[Hình 12. Kết nối của master 12](#_Toc168662765)

[Hình 13. Kết nối của Slave 12](#_Toc168662766)

[Hình 14. Kết nối của bộ ALU 12](#_Toc168662767)

[Hình 15. Thiết kế testbench 13](#_Toc168662768)

[Hình 16. Kết quả mô phỏng 13](#_Toc168662769)

1. AXI4-Lite

AXI4-Lite là một phiên bản của giao diện truyền thông AXI (Advanced eXtensible Interface) được thiết kế để đơn giản hóa việc truy cập bộ nhớ và giao tiếp với các thiết bị nội vi trong các hệ thống SoC. Giao diện này thường được sử dụng cho các thiết bị nhỏ hơn hoặc các vi mạch IP đơn giản hóa, nơi yêu cầu truy cập nhanh chóng và đơn giản hơn so với phiên bản đầy đủ của AXI. Gồm có 5 kênh:

* Write Address Channel: Kênh này được sử dụng để gửi địa chỉ của dữ liệu sẽ được ghi vào hoặc đọc từ thiết bị.
* Write Data Channel: Kênh này chứa dữ liệu mà master muốn ghi vào bộ nhớ hoặc thiết bị nội vi.
* Write Response Channel: Kênh này dùng để trả lời lại master về kết quả của phép ghi ở 2 kênh phía trên.
* Read Address Channel: Kênh này chứa địa chỉ của bộ nhớ hoặc thiết bị nội vi mà master muốn đọc dữ liệu từ slave.
* Read Data Channel: Kênh này chứa dữ liệu mà master muốn đọc ra từ bộ nhớ hoặc thiết bị nội vi.

A diagram of a computer

Description automatically generated

Hình 1. Kiến trúc Interconnect của AXI4-Lite

* Bài lab 5 này sẽ thực hiện tích hợp và mô phỏng SoC AXI4-Lite được kết nối với bộ ALU gồm các thao tác write Op1, Op2, read alu\_result thông qua AXI interface.

1. Thiết kế module AXI-master

AXI4-Lite Master tạo ra các lệnh trên bus AXI4-Lite. Các chức năng chính của master bao gồm:

* Trong quá trình ghi, ban đầu Master sẽ gửi tín hiệu AWVALID với logic cao để chỉ ra rằng có transaction ghi sẵn sàng. Nó gửi địa chỉ cụ thể bằng cách sử dụng tín hiệu AWADDR đến một bộ nhớ tùy chỉnh (16 - 32 bit) nơi nó muốn ghi dữ liệu, địa chỉ từ Master sẽ duy trì ổn định cho đến khi AWREADY tích cực mức cao.
* Bộ nhớ Slave xác nhận với Master bằng cách đưa ra tín hiệu AWREADY chỉ ra rằng nó chấp nhận địa chỉ.
* Master gửi dữ liệu đến một địa chỉ cụ thể trong Bộ nhớ sử dụng một bus dữ liệu thông qua kênh ghi dữ liệu.
* Khi dữ liệu hợp lệ có mặt trên bus dữ liệu thì tín hiệu WVALID tích cực.
* Tín hiệu WREADY tích cực chỉ ra rằng bộ nhớ chấp nhận dữ liệu.
* Trong response channel, sẽ có hai tín hiệu BVALID và BREADY. Tín hiệu BVALID sẽ cao như một token cho việc hoàn thành thành công của dữ liệu được nhận trong quá trình ghi. Tín hiệu BREADY sẽ cao khi Master muốn chấp nhận tín hiệu hợp lệ chỉ ra sự chấp nhận từ Master đến Slave.

A diagram of a system

Description automatically generated

Hình 2. Kiến trúc Write operation của AXI4-Lite

1. Write address channel

A screenshot of a computer program

Description automatically generated

Hình 3. Write address channel

Đoạn trên triển khai chức năng của Write Address Channel của một AXI4-Lite Master bao gồm các chức năng:

* Định nghĩa các trạng thái của máy trạng thái cho kênh Write Address.
* Sử dụng một câu lệnh case để xác định trạng thái tiếp theo của Master dựa trên trạng thái hiện tại.
* Xác định tín hiệu AWVALID dựa trên trạng thái tiếp theo để chỉ ra Master sẵn sàng cho phép ghi.

Đoạn này điều khiển việc gửi địa chỉ từ Master đến Slave trên kênh Write Address của giao diện AXI4-Lite.

1. Write data channel

A screenshot of a computer program

Description automatically generated

Hình 4. Write data channel

Đoạn trên triển khai chức năng giao diện Write Data Channel của một AXI4-Lite Master dùng để điều khiển quá trình gửi dữ liệu từ Master đến Slave trên kênh Write Data.

1. Response channel

A screenshot of a computer program

Description automatically generated

Hình 5. Response channel

Response Channel trong giao diện AXI4-Lite giúp đảm bảo rằng Master và Slave có thể giao tiếp với nhau một cách hiệu quả và đồng bộ trong các giao dịch ghi trên giao diện AXI4-Lite.

1. Thiết kế module AXI-slave

AXI4-Lite Slave thực hiện hoạt động đọc. Các chức năng chính của Slave gồm:

* Trong quá trình đọc, Master sẽ gửi một địa chỉ cụ thể từ nơi nó muốn đọc dữ liệu trong tệp/bộ nhớ ngoại vi thông qua tín hiệu ARADDR trong kênh địa chỉ đọc, đồng thời kiểm tra xem đây có phải là một địa chỉ hợp lệ hay không.
* Nếu có một địa chỉ hợp lệ, tín hiệu ARVALID sẽ tích cực mức cao. Địa chỉ từ Master sẽ duy trì ổn định cho đến khi tín hiệu RREADY tích cực.
* Slave đưa ra sự chấp nhận đối với Master bằng cách khẳng định ARREADY, chỉ ra rằng nó chấp nhận địa chỉ, tiếp theo là đặt dữ liệu trên một bus dữ liệu.

A diagram of a system

Description automatically generated

Hình 6. Kiến trúc Read operation của AXI4-Lite

1. Read address channel

A screenshot of a computer program

Description automatically generated

Hình 7. Read address channel

Read Address Channel được sử dụng để truyền địa chỉ của dữ liệu mà AXI-Master muốn đọc từ AXI-Slave.

* AXI-Master gửi một yêu cầu đọc bằng cách gửi tín hiệu ARADDR với địa chỉ của dữ liệu mà nó muốn đọc.
* AXI-Slave nhận tín hiệu ARADDR và xử lý yêu cầu đọc tương ứng với các tín hiệu ARREADY và ARVALID ứng với địa chỉ đó
* Sau khi hoàn thành việc xử lý, AXI-Slave sẽ gửi dữ liệu đọc tương ứng qua kênh dữ liệu đọc (Read Data Channel) của giao diện AXI.

1. Read data channel

A screenshot of a computer program

Description automatically generated

Hình 8. Read data channel

Read Data Channel được sử dụng để truyền dữ liệu từ AXI-Slave đến AXI-Master sau khi một yêu cầu đọc đã được gửi.

* Khi AXI-Slave gửi dữ liệu đến AXI-Master, nó đặt tín hiệu RDATA vào giá trị tương ứng với dữ liệu được gửi.
* AXI-Slave đặt tín hiệu RVALID ở mức cao để chỉ ra rằng dữ liệu trong RDATA là hợp lệ.
* AXI-Master kiểm tra tín hiệu RVALID để xác định xem dữ liệu nào là hợp lệ để đọc.
* Nếu AXI-Master đã sẵn sàng để nhận dữ liệu và tín hiệu RVALID cũng là mức cao, nó sẽ đọc dữ liệu từ tín hiệu RDATA.

1. Thiết kế bộ Arithmetic Logic Unit (ALU)

Module ALU được sử dụng để thực hiện các phép tính toán như cộng, trừ và các phép toán logic như AND, OR.

Trong bài lab này, ta có bảng địa chỉ các thanh ghi trong ALU như sau:

|  |  |
| --- | --- |
| Register | Address |
| Op1 | 0x300 |
| Op2 | 0x310 |
| ALU\_result | 0x320 |

Bảng 1. Địa chỉ các thanh ghi

Do đó, trong AXI slave, ta tạo các tín hiệu điều khiển we1, we2, read\_result với các điều kiện địa chỉ như trên để điều khiển lần lượt thanh ghi Op1, Op2 và ALU\_result.

A white background with black text

Description automatically generated

Hình 9. ALU control signal

Module ALU được thiết kế như hình bên dưới. Ở đây dùng thêm 3 thanh ghi reg1, reg2, reg3 để chốt các giá trị đưa vào Op1, Op2 và giá trị đưa ra ALU\_result.

Bảng tính hiệu điều khiển ALU\_op của bộ ALU:

|  |  |
| --- | --- |
| ALU\_op | Operation |
| 00 | Add |
| 01 | Sub |
| 10 | And |
| 11 | Or |

Bảng 2. Tín hiệu điều khiển ALU\_op

A computer code on a white background

Description automatically generated

Hình 10. Module ALU

1. Thiết kế module AXI\_top

Kiến trúc tổng quát của module gồm 3 phần chính:

* Master tạo và gửi các yêu cầu truy cập đến Slave thông qua AXI interface. Các yêu cầu bao gồm việc đọc hoặc ghi dữ liệu.
* Slave thực hiện các yêu cầu của Master và gửi các tín hiệu điều khiển đến ALU.
* Bộ ALU được kết nối với Slave thông qua các kênh truyền dữ liệu, cho phép nó thực hiện các phép tính trên dữ liệu từ master hoặc gửi kết quả của các phép tính.

A diagram of a computer program

Description automatically generated

Hình 11. Kiến trúc của AXI4-Lite kết nối với bộ ALU

A screenshot of a computer program

Description automatically generated

Hình 12. Kết nối của master

A screenshot of a computer program

Description automatically generated

Hình 13. Kết nối của Slave

A screenshot of a computer code

Description automatically generated

Hình 14. Kết nối của bộ ALU

1. Thiết kế testbench và kiểm tra kết quả mô phỏng
2. Thiết kế testbench

Lần lượt ghi vào các địa chỉ 0x300 của thanh ghi Op1, 0x310 của thanh ghi Op2 và đọc ra địa chỉ 0x320 của thanh ghi ALU\_result. Sau đó điều khiển tín hiệu ALU\_op lần lượt thực hiện các chức năng của bộ ALU: add, sub, and, or.

A screenshot of a computer code

Description automatically generated

Hình 15. Thiết kế testbench

1. Kiểm tra kết qua mô phỏng

A screenshot of a computer

Description automatically generated

Hình 16. Kết quả mô phỏng

Bảng giá trị đầu vào và đầu ra của hệ thống:

|  |  |  |  |
| --- | --- | --- | --- |
| Op1 | Op2 | ALU\_op | ALU\_result |
| 15  1111 | 8  1000 | add | 23 |
| sub | 7 |
| and | 1000 |
| or | 1111 |

|  |  |  |  |
| --- | --- | --- | --- |
| Op1 | Op2 | ALU\_op | ALU\_result |
| 12  1100 | 8  1000 | add | 20 |
| sub | 4 |
| and | 1000 |
| or | 1100 |

1. Kết luận

* Tích hợp được các thành phần để tạo thành một hệ thống SoC đơn giản bằng giao thức AXI4-Lite kết nối với bộ ALU.
* Mô phỏng thiết kế SoC bằng công cụ Vivado.