JTE
HCMUTE
ISSN: 1859-1272

JOURNAL OF TECHNICAL EDUCATION SCIENCE
Ho Chi Minh City University of Technology and Education
Website: https://jte.hcmute.edu.vn/index.php/jte/index
Email: jte@hcmute.edu.vn

Design and Verification of an SPI-Wishbone Controller

Vo Thien Tung, Thai Duong Tuan Thanh, Vo Phan Man Dat, Dang Le Ngoc Hoa, Nguyen Van Thuong, Van Ba Huy, Le Thi Quynh Nhi, Nguyen Minh Nhat, Do Duy Tan*
Faculty of Electrical and Electronics Engineering, HCMC University of Technology and Education, Vietnam
* Corresponding author. *Email: tandd@hcmute.edu.vn*

ARTICLE INFO

Received: 19/2/2022
Revised: 18/5/2022
Accepted: 16/8/2022
Published: 30/8/2022

KEYWORDS

SPI standard;
Wishbone bus;
FPGA;
Verilog;
Testbench.

ABSTRACT

SPI (Serial Peripheral Interface) is a synchronous serial communication standard for connecting and transferring data between electronic devices proposed and developed by Motorola Inc. The main advantages of SPI standard are high data transmission speed, synchronization, simple connection, and low logic resources. Furthermore, Wishbone is a popular bus standard with open source codes, widely used in Silicore Corporation's projects. In this paper, we present a detailed design of a controller called SPI-Wishbone, which communicates with peripheral devices based on the SPI standard that can be configured in either Master mode or Slave mode. The designed module can be controlled, transmit, and receive data from a central processing unit via Wishbone bus. Finally, we conduct extensive simulation results and a summary of logic resource usage and power consumption to validate the functionality and effectiveness of the proposed design. We use Verilog Hardware Description Language in the design and simulation processes of the SPI-Wishbone module.

Thiết Kế Và Thi Công Bộ Truyền Nhận Theo Giao Thức SPI-Wishbone

Võ Thiện Tùng, Thái Dương Tuấn Thành, Võ Phan Mẫn Đạt, Đặng Lê Ngọc Hoà, Nguyễn Văn Thương, Văn Bá Huy, Lê Thị Quỳnh Nhi, Nguyễn Minh Nhật, Đỗ Duy Tân*
Khoa Điện-Điện Tử, Trường Đại Học Sư Phạm Kỹ Thuật Thành Phố Hồ Chí Minh, Việt Nam
** Tác giả liên hệ. Email: tandd@hcmute.edu.vn*

THÔNG TIN BÀI BÁO

Ngày nhận bài: 19/2/2022
Ngày hoàn thiện: 18/5/2022
Ngày chấp nhận đăng: 16/8/2022
Ngày đăng: 30/8/2022

TỪ KHÓA

Chuẩn SPI;
Wishbone bus;
FPGA;
Verilog;
Testbench.

TÓM TẮT

SPI (Serial Peripheral Interface) là chuẩn truyền thông nối tiếp đồng bộ dùng để kết nối và truyền dữ liệu giữa các thiết bị điện tử, được phát triển bởi tập đoàn Motorola. Ưu điểm của chuẩn SPI nằm ở tốc độ truyền dữ liệu cao, đồng bộ trong việc giao tiếp, cách kết nối đơn giản và tiết kiệm tài nguyên sử dụng. Trong khi đó, Wishbone là một chuẩn bus thông dụng với mã nguồn mở, được sử dụng nhiều trong các dự án của Silicore Corporation. Trong bài báo này, chúng tôi trình bày một thiết kế chi tiết bộ điều khiển mang tên SPI-Wishbone, giao tiếp với các thiết bị ngoại vi dựa trên chuẩn SPI có thể hoạt động ở chế độ Master hoặc Slave, đồng thời truyền nhận dữ liệu và nhận sự điều khiển từ bộ xử lý trung tâm thông qua chuẩn bus Wishbone. Cuối cùng, chúng tôi thực hiện đánh giá chi tiết thiết kế thông qua môi trường mô phỏng và tổng hợp các thành phần tài nguyên sử dụng, tần số hoạt động tối đa, công suất tiêu thụ nhằm xác thực tính năng của thiết kế được đề xuất. Chúng tôi sử dụng ngôn ngữ mô tả phần cứng Verilog trong quá trình thiết kế và mô phỏng module SPI-Wishbone.

Doi: https://doi.org/10.54644/jte.71B.2022.1142

1. Giới thiệu

Serial Peripheral Interface (SPI) là chuẩn truyền thông nối tiếp được phát triển bởi tập đoàn Motorola. Nhờ tính đơn giản, khả năng truyền tốc độ cao mà trong đó tuỳ thuộc vào tần số xung đồng hồ (clock)

JTE
HCMUTE
ISSN: 1859-1272

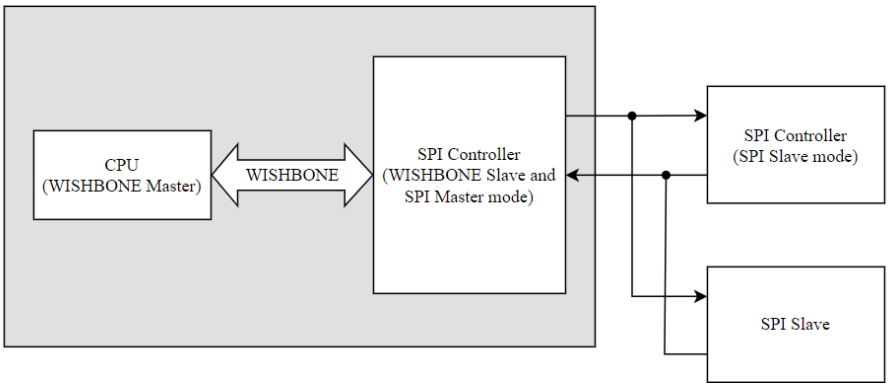
JOURNAL OF TECHNICAL EDUCATION SCIENCE
Ho Chi Minh City University of Technology and Education
Website: https://jte.hcmute.edu.vn/index.php/jte/index
Email: jte@hcmute.edu.vn

SPI sử dụng có thể đạt tốc độ 10Mbps, chuẩn SPI được sử dụng rộng rãi trong các vi mạch tích hợp để giao tiếp với các thiết bị ngoại vi như cảm biến nhiệt độ, cảm biến áp suất, màn hình LCD, các bộ nhớ flash, thanh ghi dịch, SRAM, DRAM…

Trong bài báo [1], tác giả đã trình bày một thiết kế số truyền nhận dữ liệu với chuẩn USB trong đó sử dụng truyền thông nối tiếp để nhận và gửi dữ liệu đến khối xử lý trung tâm. Bài báo [2] các tác giả đã trình bày thiết kế, mô phỏng và tối ưu hoá tần số hoạt động và diện tích thiết kế SPI trên ô tô. Ngoài ra, thiết kế giao thức SPI với ngôn ngữ mô tả phần cứng Verilog [3-4] được đưa ra ở nhiều bài báo [5-8]. Trong đó, bài báo [4], các tác giả thực hiện thiết kế chuẩn SPI trên tiêu chuẩn DO-254 với độ tin cậy cao trong các môi trường thực tế như điện tử hàng không, quân sự. Bài báo [5] trình bày thiết kế bộ chuyển đổi giữa 2 giao thức SPI và I2C với mục đích thực hiện giao tiếp giữa thiết bị gửi có tốc độ cao với thiết bị nhận có tốc độ thấp. Bài báo [6], các tác giả đã cung cấp mô tả thiết kế từ việc xây dựng các thông số kỹ thuật ban đầu đến bước xác minh khối IP SPI Master/Slave trên FPGA trong đó sử dụng OPB (On-Chip Peripheral Bus). Bài báo [7] các tác giả trình bày thiết kế IP sử dụng giao thức APB (Advanced Peripheral Bus) và SPI, hỗ trợ chọn các chế độ hoạt động và tốc độ truyền dữ liệu khác nhau. Bài báo [8], các tác giả đã thiết kế lõi IP SPI tốc độ cao và có thể tái sử dụng dựa trên giao thức Wishbone, tần số hoạt động tối đa tổng hợp trên Virtex-II Pro Xilinx là 115Mhz. Bài báo [9], đối với APB [10], các tác giả đã trình bày và triển khai thiết kế IP SPI-APB, thiết kế này hỗ trợ chế độ Master/Slave, 4 chế độ truyền, dữ liệu có thể thay đổi từ 1-32 bit và chọn tốc độ truyền. SPI đã được các tác giả thiết kế đi kèm với các giao thức bus khác nhau là OPB, APB và Wishbone. Tuy nhiên, chi tiết về thiết kế chưa được trình bày. Việc ứng dụng giao thức SPI vào bên trong các hệ thống lớn đòi hỏi sự tương thích với các bus hệ thống khác, do đó bài báo này trình bày một thiết kế có thể hoạt động ở chế độ Master/Slave, truyền nhận dữ liệu giữa các thiết bị ngoại vi với phần cứng trung tâm sử dụng giao thức SPI và chuẩn bus Wishbone, một chuẩn bus thông dụng trong các dự án mã nguồn mở của Silicore Corporation [11], [12]. Nhờ vào việc ứng dụng chuẩn Wishbone vào trong thiết kế, module có thể được tái sử dụng để giao tiếp giữa hệ thống với các ngoại vi cần tốc độ cao cũng sử dụng chuẩn truyền dữ liệu SPI, từ đó nhận thấy được sự tương thích giữa SPI và Wishbone, đem đến nhiều sự lựa chọn hơn trong việc lựa chọn các chuẩn bus. Thiết kế này được tổng hợp và đánh giá thông qua mô phỏng trên phần mềm Xilinx ISE Design Suite 14.7 và QuestaSim 10.2.

2. Thiết kế chi tiết

Module SPI Controller mô tả ở hình 1 thực hiện công việc truyền nhận dữ liệu dựa trên tín hiệu điều khiển từ CPU thông qua Wishbone khi được cấu hình là SPI Master. Đồng thời SPI Controller Master có thể giao tiếp với các SPI Controller khác khi được cấu hình Slave thông qua giao thức SPI.

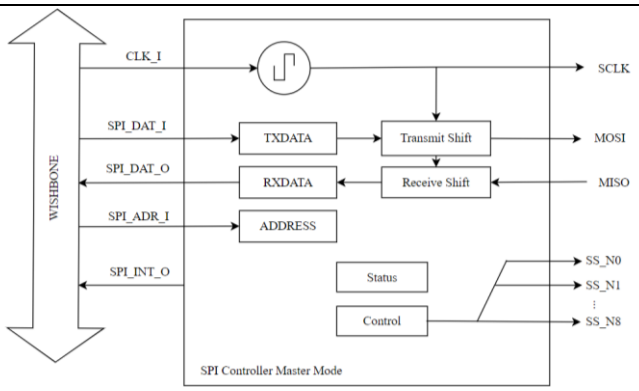


Hình 1. *Mô hình truyền nhận dữ liệu sử dụng Wishbone bus và SPI*

2.1. Cấu tạo của module ở chế độ SPI Master

Hình 2 mô tả sơ đồ khối của SPI Controller khi được cấu hình là SPI Master.

JTE HCMUTE
ISSN: 1859-1272
JOURNAL OF TECHNICAL EDUCATION SCIENCE
Ho Chi Minh City University of Technology and Education
Website: https://jte.hcmute.edu.vn/index.php/jte/index
Email: jte@hcmute.edu.vn



Hình 2. *Sơ đồ khối SPI Controller Master mode*

2.1.1. Bộ chia tần số

Bộ chia tần số hoạt động dựa vào tín hiệu CLK_I (CLOCK INPUT - tần số xung clock ngõ vào của module) được cung cấp từ CPU và hệ số CLOCK_SEL (CLOCK SELECT - Hệ số lựa chọn giá trị chia), ngõ ra là tín hiệu clock SCLK (SERIAL CLOCK – tần số xung clock điều khiển truyền dữ liệu trên MISO và MOSI) với giá trị mong muốn cung cấp cho module SPI Master và SPI Slave. Trong thiết kế này, CLOCK_SEL có giá trị không đổi là 1. Tần số SCLK được tính theo công thức (1).

$$SCLK = \frac{CLK\_I}{2 * (CLOCK\_SEL + 1)} = \frac{CLK\_I}{4} \quad (1)$$

2.1.2. Quá trình truyền

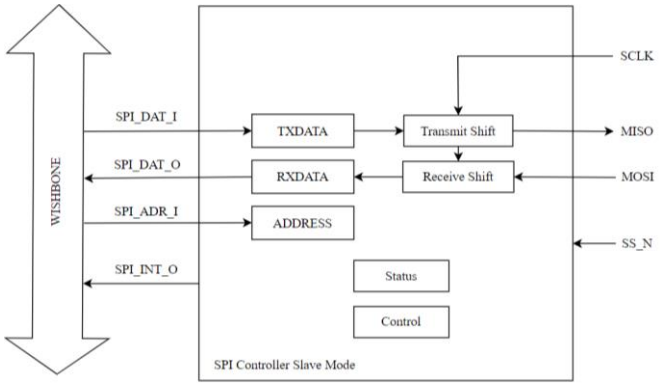
Ban đầu dữ liệu từ đường truyền Wishbone được đưa vào một thanh chốt dữ liệu (latch), dữ liệu được giữ lại ở đây để chờ cấp phép (tín hiệu xung CLK). Khi có điều kiện xác nhận hoạt động truyền dữ liệu và có tín hiệu từ thanh ghi địa chỉ thì dữ liệu sẽ được đưa vào thanh ghi TXDATA của thiết bị Master. Sau đó có tín hiệu cho phép truyền, dữ liệu truyền sang thanh ghi Transmit Shift. Tại đây, thanh ghi có nhiệm vụ đếm và tạo khung dữ liệu 8 bit. Tiếp theo đó, thông báo cho TXDATA đã đủ dữ liệu truyền và bắt đầu dịch dữ liệu chân MOSI theo xung CLK.

2.1.3. Quá trình nhận

Quá trình nhận tín hiệu bắt đầu khi khối Master nhận tín hiệu cho phép truyền và các tín hiệu điều khiển quá trình truyền như xung clock và tín hiệu đồng bộ pha. Sau đó, chân MISO sẽ gửi dữ liệu lần lượt theo xung clock vào thanh ghi Receive Shift. Khi Receive Shift đã nhận được đầy đủ dữ liệu, Master sẽ kéo cờ báo hiệu lên mức cao. Lúc này toàn bộ dữ liệu Receive Shift được chuyển sang RXDATA. Dữ liệu này tiếp tục truyền đến bộ điều khiển trung tâm qua SPI_DAT_O nếu nhận được yêu cầu.

2.2. Cấu tạo của module ở chế độ SPI Slave

Hình 3 mô tả sơ đồ khối của SPI Controller khi được cấu hình là SPI Slave. Trong đó module nhận tín hiệu xung clock SCLK và tín hiệu chọn thiết bị Slave SS_N được cung cấp từ SPI Controller Master.



Hình 3. *Sơ đồ khối SPI Controller Slave mode*

JTE
HCMUTE
ISSN: 1859-1272

JOURNAL OF TECHNICAL EDUCATION SCIENCE
Ho Chi Minh City University of Technology and Education
Website: https://jte.hcmute.edu.vn/index.php/jte/index
Email: jte@hcmute.edu.vn

2.2.1. Quá trình truyền

Quá trình truyền ở khối Slave thực hiện gần tương tự so với thiết bị Master, dữ liệu thu thập từ Slave sẽ được đưa vào một thanh chốt dữ liệu (Latch_data), khi có tín hiệu từ xung CLK và tín hiệu từ thanh ghi địa chỉ gửi tới, dữ liệu sẽ được truyền từ Latch_data tới TXDATA của thiết bị Slave.

Dữ liệu sau khi được truyền qua thanh ghi TXDATA của thiết bị Slave, khi có tín hiệu từ xung CLK và phase gửi tới thì dữ liệu sẽ truyền từ TXDATA qua thanh ghi Transmit_shift, thanh ghi này có chức năng tạo khung dữ liệu 8 bit và khi có tín hiệu đồng bộ từ xung CLK dữ liệu sẽ được dịch từng bit theo CLK ra chân MISO và truyền đến thiết bị Master.

2.2.2. Quá trình nhận

Đối với quá trình nhận tín hiệu của Slave bắt đầu khi Master kéo chân chọn thiết bị SS_Slave mà Master muốn truyền xuống mức thấp. Khi đó, chân MOSI ở Slave sẽ gửi dữ liệu lần lượt theo xung clock đồng bộ vào Receive Shift. Sau khi Receive Shift đã nhận đủ dữ liệu thì Slave kéo cờ báo hiệu cho phép Receive Shift truyền sang RXDATA.

2.3. Mô tả thanh ghi

Các thanh ghi được sử dụng trong thiết kế SPI Controller mô tả trong bảng 1.

Bảng 1. *Thông tin về thanh ghi sử dụng trong SPI Controller*

Thanh ghi	Địa chỉ	Độ rộng (bit)	Quyền truy cập của Wishbone	Mô tả
RXDATA	0x00	8	Chỉ được đọc	Dữ liệu từ SPI
TXDATA	0x04	8	Đọc hoặc ghi	Dữ liệu gửi
STATUS	0x08	8	Chỉ được đọc	Thanh ghi Status
CONTROL	0x0C	8	Đọc hoặc ghi	Thanh ghi Control
SS_MASK	0x10	8	Đọc hoặc ghi	Thanh ghi lựa chọn Slave

Thanh ghi **Status** 8 bit hiển thị trạng thái hoạt động của các thanh ghi truyền nhận khác, chức năng của từng bit trong thanh ghi được định nghĩa tại bảng 2.

Bảng 2. *Định nghĩa bit trong thanh ghi Status*

Tên bit	Bit	Mô tả
Reserved	1:0	Không sử dụng
ROE	2	Quá tải trong quá trình nhận dữ liệu (hiển thị bit ‘1’ khi có lỗi), lỗi cho thấy thanh ghi RXDATA nhận dữ liệu mới trước khi dữ liệu trước đó được đọc, dữ liệu trước sẽ bị mất nếu điều này xảy ra.
TOE	3	Quá tải trong quá trình truyền dữ liệu (hiển thị bit ‘1’ khi có lỗi), lỗi xảy ra khi thanh ghi TXDATA nhận dữ liệu mới trước khi dữ liệu được truyền sang cho thanh ghi Transmit shift.
TMT	4	Thanh ghi Transmit shift đang trống dữ liệu và sẵn sàng nhận dữ liệu từ thanh ghi TXDATA (hiển thị bit ‘1’).
TRDY	5	Thanh ghi TXDATA đang trống và sẵn sàng nhận dữ liệu từ Wishbone bus truyền tới (hiển thị bit ‘1’).
RRDY	6	Thanh ghi RXDATA có dữ liệu và sẵn sàng cho việc đọc dữ liệu (hiển thị bit ‘1’).
E	7	Lỗi bit trong quá trình truyền hoặc nhận dữ liệu.

Thanh ghi **Control** 8 bit điều khiển hoạt động của các khối Master hoặc Slave trong quá trình truyền nhận dữ liệu, các bit trong thanh ghi này được định nghĩa tại bảng 3.

3. Kết quả đánh giá qua mô phỏng

3.1. Tài nguyên sử dụng

Thiết kế được tổng hợp trên phần mềm Quartus Prime Lite với cấu hình board Cyclone V GT FPGA Development Kit để đánh giá tài nguyên logic cần sử dụng tại bảng 4, tần số tối đa tại bảng 5 và ước lượng công suất tiêu thụ của thiết kế khi chạy trên board tại bảng 6. Trong đó, tần số hoạt động tối đa (Fmax) ở bảng 5 đối với thiết kế này lớn hơn 1.39 lần so với [8], đồng thời thiết kế được đề xuất đòi hỏi

JTE
HCMUTE
ISSN: 1859-1272

JOURNAL OF TECHNICAL EDUCATION SCIENCE
Ho Chi Minh City University of Technology and Education
Website: https://jte.hcmute.edu.vn/index.php/jte/index
Email: jte@hcmute.edu.vn

tài nguyên rất nhỏ (<1%) đối với các thành phần logic, tổ hợp và thanh ghi, chỉ chiếm 8% đối với các chân i/o.

Bảng 3. *Định nghĩa bit trong thanh ghi Control*

Tên bit	Bit	Mô tả
IROE	0	Yêu cầu ngắt khi quá trình nhận dữ liệu bị quá tải (hiển thị bit '1').
ITOE	1	Yêu cầu ngắt khi quá trình truyền dữ liệu bị quá tải (hiển thị bit '1').
Reserved	2	Không sử dụng.
ITRDY	3	Ngắt hoạt động các khối khác khi khối truyền trong trạng thái sẵn sàng truyền dữ liệu (hiển thị bit '1').
IRRDY	4	Ngắt hoạt động các khối khác khi khối nhận dữ liệu trong trạng thái sẵn sàng nhận dữ liệu (hiển thị bit '1').
IE	5	Ngắt hoạt động khi quá trình truyền hoặc nhận bị lỗi quá tải (hiển thị bit '1').
Reserved	6	Không sử dụng.
SSO	7	Giữ sự liên kết khi quá trình truyền nhận dữ liệu giữa 2 khối SPI Master và SPI Slave, sau khi trao đổi dữ liệu xong sẽ kết thúc việc kết nối (hiển thị bit '1').

Bảng 4. *Tài nguyên logic được sử dụng*

Tài nguyên được sử dụng			
Tài nguyên logic	Đã dùng	Có sẵn	Sử dụng
Thành phần logic	60	113560	<1%
Tổ hợp chức năng	82	113560	<1%
Thanh ghi	97	113560	<1%
Chân i/o	51	616	8%

Bảng 5. *Tần số hoạt động tối đa của thiết kế*

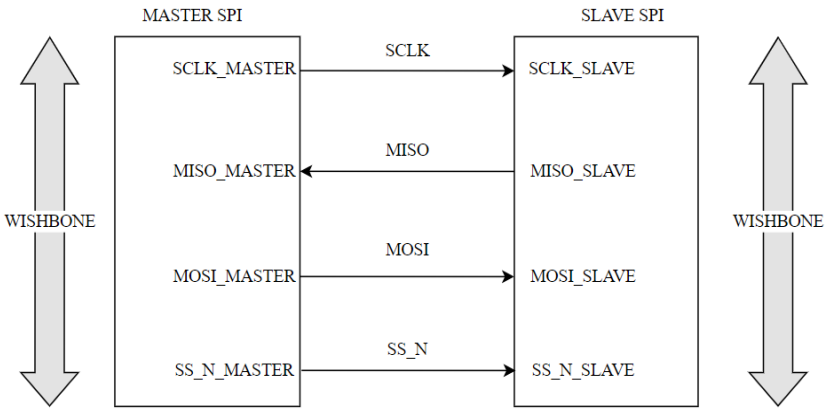
Tóm tắt định thời		
Fmax	**Tần số tối đa hạn chế**	**Tên xung clock**
160.46MHZ	160.46MHZ	CLK_I

Bảng 6. *Công suất tiêu thụ của thiết kế*

Tóm tắt công suất	
Công suất tiêu tán tổng	537.52mW
Công suất tiêu tán động	1.91mW
Công suất tiêu tán tĩnh	518.75mW
Công suất tiêu tán I/0	16.85mW

3.2. Mô hình mô phỏng và kết quả kiểm chứng

Hoạt động của thiết kế được kiểm tra mô phỏng trên phần mềm QuestaSim 10.2 với cấu hình là 1 Master và 1 Slave được thể hiện trong hình 4. Trong mô hình này, module SPI được điều khiển và truyền nhận dữ liệu từ Wishbone bus.



Hình 4: *Mô hình kiểm tra đánh giá hoạt động của module được thiết kế.*

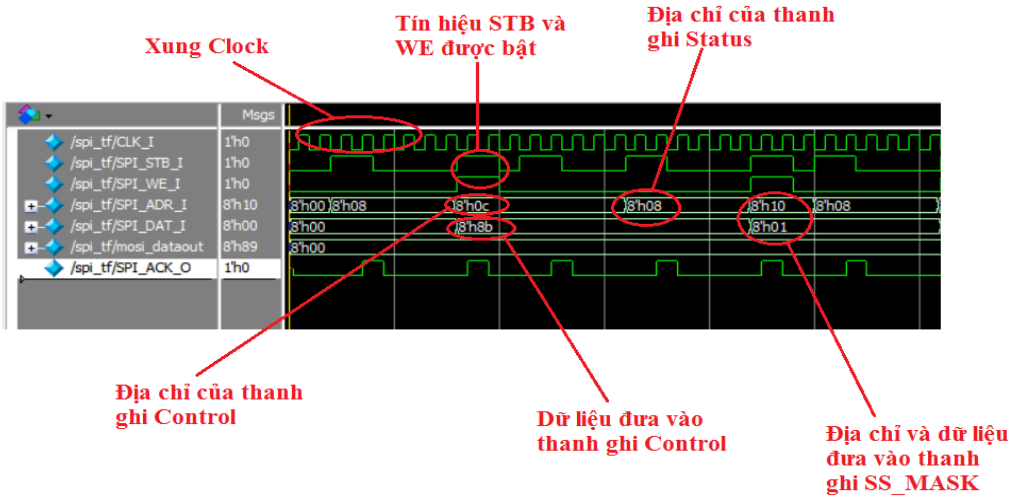
JTE
HCMUTE
ISSN: 1859-1272

JOURNAL OF TECHNICAL EDUCATION SCIENCE
Ho Chi Minh City University of Technology and Education
Website: https://jte.hcmute.edu.vn/index.php/jte/index
Email: jte@hcmute.edu.vn

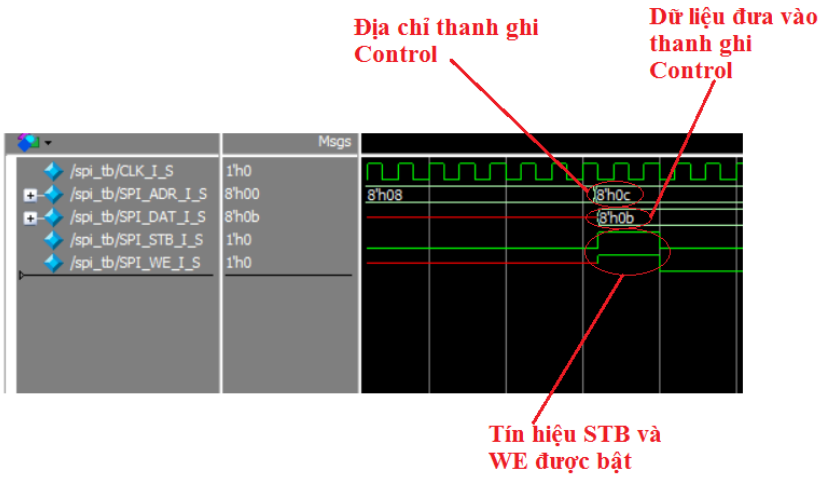
Việc kiểm tra hoạt động của thiết kế trên phần mềm mô phỏng được thực hiện thông qua 2 test case: Master truyền – Slave nhận; Master nhận – Slave truyền được trình bày chi tiết trong bảng 7. Các test case lần lượt được thực hiện. Dữ liệu truyền gồm 2 byte: 8'hda, 8'hf7.

Bảng 7. *Bảng tóm tắt test case*

Test case	Master	Slave
1 (Hình 5,6,7,8) Master truyền – Slave nhận	- Cấu hình thanh ghi Control (0x0c): Master nhận, đưa vào Control 8'h8b. - Cấu hình thanh ghi SS_MASK (0x10): Master nhận, đưa vào SS_MASK 8'h01. - Thanh ghi TXDATA (0x04): Master nhận, đưa vào TXDATA, truyền qua Slave. - Kết thúc: Truy cập SS_MASK, xoá dữ liệu trong thanh ghi.	- Cấu hình thanh ghi Control (0x0c): Master nhận, đưa vào Control 8'h0b. - Thanh ghi RXDATA (0x00): Nhận dữ liệu từ Master, đưa đến Wishbone của Slave.
2 (Hình 5,6,9) Master nhận – Slave truyền	- Cấu hình thanh ghi Control (0x0c): Master nhận, đưa vào Control 8'h8b. - Cấu hình thanh ghi SS_MASK (0x10): Master nhận, đưa vào SS_MASK 8'h01. - Thanh ghi RXDATA (0x00): Nhận dữ liệu từ Slave, đưa đến Wishbone của Master - Kết thúc: Truy cập SS_MASK, xoá dữ liệu trong thanh ghi.	- Cấu hình thanh ghi Control (0x0c): Master nhận, đưa vào Control 8'h0b. - Thanh ghi TXDATA (0x04): Slave nhận dữ liệu, đưa vào TXDATA, truyền đến Master.



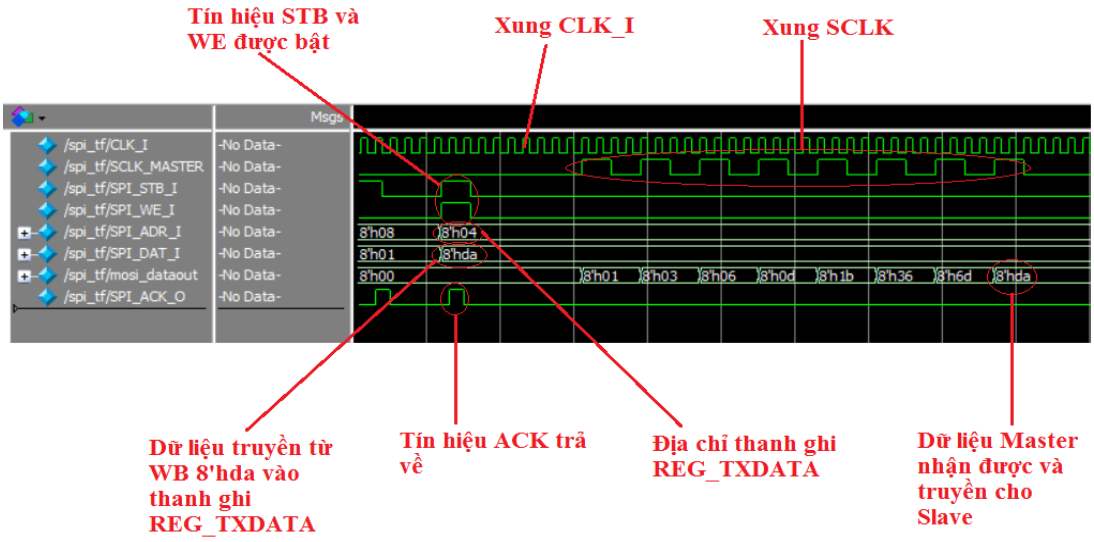
Hình 5. *Dạng sóng mô phỏng quá trình cấu hình trong Master (Test case 1,2)*



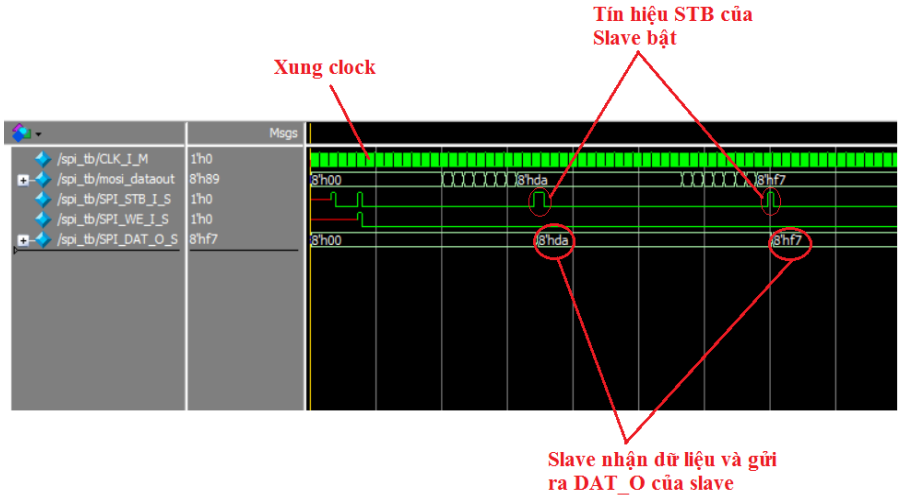
Hình 6. *Dạng sóng mô phỏng quá trình cấu hình trong Slave (Test case 1,2)*

JTE
HCMUTE
ISSN: 1859-1272

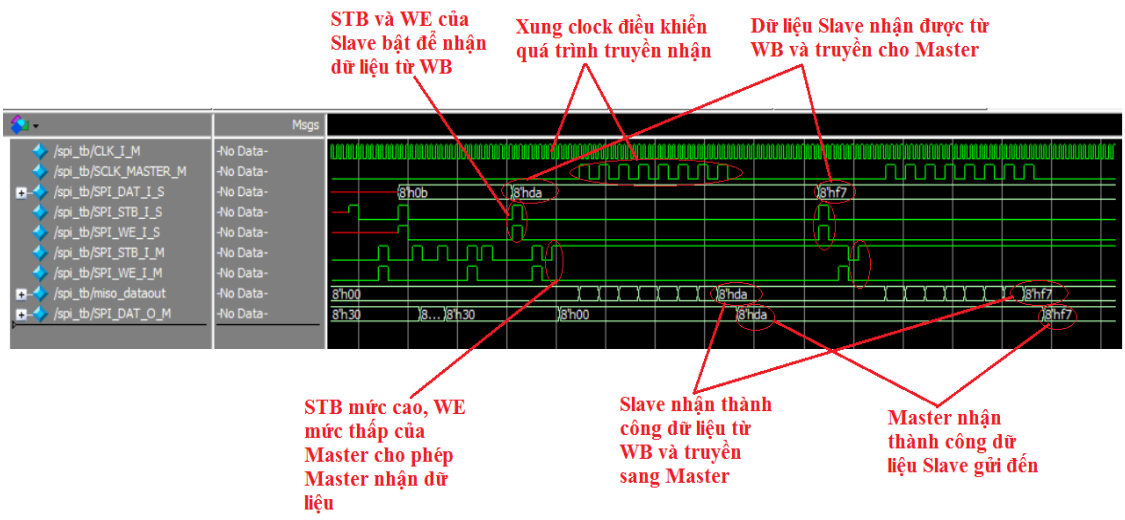
JOURNAL OF TECHNICAL EDUCATION SCIENCE
Ho Chi Minh City University of Technology and Education
Website: https://jte.hcmute.edu.vn/index.php/jte/index
Email: jte@hcmute.edu.vn



Hình 7. *Dạng sóng mô phỏng quá trình truyền data 8'hda của Master (Test case 1)*



Hình 8. *Dạng sóng mô phỏng quá trình nhận dữ liệu của Slave và đưa ra Wishbone (Test case 1)*



Hình 9. *Dạng sóng mô phỏng quá trình Master nhận – Slave truyền (Test case 2)*

3.3. Phân tích kết quả mô phỏng

3.3.1. Test case 1

- Master truyền

Master được cấu hình bằng cách nhận tín hiệu chọn thanh ghi adr_i, stb_i và we_i lên mức cao cho phép Master nhận dữ liệu, sau đó đưa dữ liệu vào thanh ghi bởi tín hiệu dat_i (hình 5). adr_i gửi tín hiệu truy cập vào thanh ghi TXDATA, stb_i và we_i lên mức cao cho phép Master nhận dữ liệu qua dat_i. Khi Master đã nhận dữ liệu thành công, ack_o báo hiệu mức cao. Sau 8 chu kỳ xung SCLK, dữ liệu 8'hda được truyền sang Slave (trong hình 7). Tương tự cho byte dữ liệu 8'hf7.

- Slave nhận

Slave được cấu hình bằng cách nhận tín hiệu chọn thanh ghi adr_i, stb_i và we_i của Slave lên mức cao cho phép Slave nhận dữ liệu thông qua dat_i (trong hình 6). Slave nhận được dữ liệu từ Master, dữ liệu này được lưu trữ trong thanh ghi RXDATA. Tín hiệu stb_i lên mức cao và we_i mức thấp cho phép quá trình nhận dữ liệu, khi đó dữ liệu Slave nhận được từ Master gửi qua dat_o đến Wishbone (hình 8).

3.3.2. Test case 2

- Master nhận

Sau khi Master được cấu hình như trong hình 5. Master bật stb_i lên mức cao đồng thời we_i mức thấp cho phép quá trình nhận dữ liệu từ Slave. Dữ liệu Master nhận được gửi đến Wishbone thông qua dat_o (như thể hiện trong hình 9).

- Slave truyền

Quá trình này Slave được cấu hình tương tự như ở test case 1 (tại hình 6). Slave bật tín hiệu stb_i và we_i cho phép nhận dữ liệu từ adr_i. Sau 8 chu kỳ xung thì dữ liệu này được gửi hoàn thành đến Master (tại hình 9).

3.3.3. Đánh giá chung

Sau quá trình mô phỏng, đánh giá và kiểm tra so với lý thuyết, chúng tôi rút ra được kết quả đánh giá như được trình bày trong bảng 8.

Bảng 8. *Bảng đánh giá kết quả mô phỏng*

Test case	Đánh giá
1	- Cấu hình thành công thanh ghi Control và SS_MASK: Nhận byte địa chỉ - truy cập thanh ghi – đưa data vào thanh ghi. - Master nhận thành công hai byte data từ Wishbone của Master và gửi thành công đến Wishbone của Slave. - Thiết kế hoạt động chính xác khi được cấu hình là Master truyền – Slave nhận.
2	- Cấu hình thành công thanh ghi Control và SS_MASK: Nhận byte địa chỉ - truy cập thanh ghi – đưa data vào thanh ghi. - Slave nhận thành công hai byte data từ Wishbone của Slave và gửi thành công đến Wishbone của Master. - Thiết kế hoạt động chính xác khi được cấu hình là Master nhận – Slave truyền.

4. Kết luận

Trong bài báo này, chúng tôi đã trình bày chi tiết thiết kế bộ truyền nhận sử dụng giao thức SPI để giao tiếp ngoại vi và chuẩn bus Wishbone để điều khiển. Thông qua mô phỏng, chúng tôi đã chỉ ra rằng module SPI-Wishbone được thiết kế hoạt động đúng chức năng khi được so sánh với lý thuyết, với tài nguyên sử dụng ít hơn và tần số hoạt động tối đa lớn hơn so với một số thiết kế trong các bài báo khác. Các thiết kế bộ truyền nhận hiện nay thường được thiết kế xoay quanh chuẩn bus AMBA của ARM, tuy nhiên vẫn có rất nhiều lựa chọn cho các chuẩn bus khác, việc áp dụng chuẩn Wishbone vào thiết kế sẽ đem đến nhiều lựa chọn hơn trong quá trình thiết kế hệ thống và thể hiện được lợi thế của Wishbone so với các chuẩn còn lại nhờ vào sự đơn giản nhưng hiệu suất vẫn ổn định cùng với mã nguồn mở dễ tiếp cận.

Bài báo đã trình bày chi tiết quá trình thiết kế và đánh giá một module số thông dụng (SPI), do đó có thể được xem như tài liệu tham khảo cho các môn học liên quan đến thiết kế vi mạch số. Bên cạnh đó, có thể phát triển thiết kế được đề xuất như một soft IP và tích hợp vào các thiết kế hệ thống trên chip (SoC) lớn hơn sử dụng chuẩn bus Wishbone.

Lời cám ơn

Bài báo này được tài trợ kinh phí nghiên cứu năm 2022 bởi trường Đại Học Sư phạm Kỹ thuật Thành Phố Hồ Chí Minh.

TÀI LIỆU THAM KHẢO

[1] G. Sung, L. Tung, H. Wang and J. Lin, "USB Transceiver With a Serial Interface Engine and FIFO Queue for Efficient FPGA-to-FPGA Communication," in IEEE Access, vol. 8, pp. 69788-69799, 2020.
[2] I. I. b. Jamaludin and H. b. Hassan, "Design and Analysis of Serial Peripheral Interface for Automotive Controller," 2020 IEEE Student Conference on Research and Development (SCOReD), 2020, pp. 498-501.
[3] Anand N, G. Joseph, S. S. Oommen and R. Dhanabal, "Design and implementation of a high speed Serial Peripheral Interface," 2014 International Conference on Advances in Electrical Engineering (ICAEE), 2014, pp. 1-3.
[4] M. Koushik, R. Anushree, B. J. Sowmya and N. Geethanjali, "Design of SPI Protocol with DO-254 Compliance for Low Power Applications," 2017 International Conference on Recent Advances in Electronics and Communication Technology (ICRAECT), 2017, pp. 186-190.
[5] D. Trivedi, A. Khade, K. Jain and R. Jadhav, "SPI to I2C Protocol Conversion Using Verilog," 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA), 2018, pp. 1-4.
[6] A. K. Oudjida, M. L. Berrandjia, A. Liacha, R. Tiar, K. Tahraoui and Y. N. Alhoumays, "Design and test of general-purpose SPI Master/Slave IPs on OPB bus," 2010 7th International Multi- Conference on Systems, Signals and Devices, 2010, pp. 1-6.
[7] M. Hafeez and A. Saparon, "IP Core of Serial Peripheral Interface (SPI) with AMBA APB Interface," 2019 IEEE 9th Symposium on Computer Applications & Industrial Electronics (ISCAIE), 2019, pp. 55-59.
[8] B. Zhou, D. Li and G. Lu, "Design of high-speed and reusable SPI IP core based on Wishbone interface," 2011 International Conference on Electrical and Control Engineering, 2011, pp. 1040-1042.
[9] J. Yang, Y. Xiao, D. Li, Z. Li, Z. Chen and P. Wan, "A Configurable SPI Interface Based on APB Bus," 2020 IEEE 14th International Conference on Anti-counterfeiting, Security, and Identification (ASID), 2020, pp. 73-76.
[10] Ngoc P.T., Bao H.N., Tan D.D., Phuc T.Q., Ca P.V., "A novel multichannel UART design with FPGA-based implementation", International Journal of Computer Applications in Technology, vol. 67, no. 4, pp 358–369, 2022.
[11] Richard Herveille, WISHBONE System-on-Chip (SoC) Interconnection Architecture for Portable IP Cores, rev. version: B4, Open Cores Organization, p.7, 2010. www.opencores.org
[12] C. Dongye, "Design of the on-chip bus based on Wishbone," 2011 International Conference on Electronics, Communications and Control (ICECC), 2011, pp. 3653-3656.

Vo Thien Tung is currently a student at the Ho Chi Minh City University of Technology and Education (HCMUTE), Vietnam. His main research interests include wireless communication networks and FPGA-based designs for DSP applications.

Thai Duong Tuan Thanh is currently a student at the Ho Chi Minh City University of Technology and Education (HCMUTE), Vietnam. His main research interests include wireless communication networks and digital system designs.

Vo Phan Man Dat is currently a student at the Ho Chi Minh City University of Technology and Education (HCMUTE), Vietnam. His main research interests include wireless communication networks and FPGA-based designs for DSP applications.

Dang Le Ngoc Hoa is currently a student at the Ho Chi Minh City University of Technology and Education (HCMUTE), Vietnam. His main research interests include wireless communication networks and digital system designs.

Nguyen Van Thuong is currently a student at the Ho Chi Minh City University of Technology and Education (HCMUTE), Vietnam. His main research interests include wireless communication networks and digital system designs.

Van Ba Huy is currently a student at the Ho Chi Minh City University of Technology and Education (HCMUTE), Vietnam. His main research interests include wireless communication networks and digital system designs.

Le Thi Quynh Nhi is currently a student at the Ho Chi Minh City University of Technology and Education (HCMUTE), Vietnam. His main research interests include wireless communication networks and FPGA-based designs for DSP applications.

Nguyen Minh Nhat is currently a student at the Ho Chi Minh City University of Technology and Education (HCMUTE), Vietnam. His main research interests include wireless communication networks and FPGA-based designs for DSP applications.

Do Duy Tan received his B.S. degree from Ho Chi Minh City University of Technology (HCMUT), Vietnam, and M.S. degree from Kumoh National Institute of Technology, Korea, in 2010 and 2013, respectively. He received his Ph.D. degree from Autonomous University of Barcelona, Spain, in 2019. He is currently with the Department of Computer and Communication Engineering, Ho Chi Minh City University of Technology and Education (HCMUTE) in Vietnam as an Assistant Professor. His main research interests include real-time optimisation for resource allocation in wireless networks and coding applications for wireless communications.