

X2P IP core

X2P IP core

VLSI Technology

X2P IP core

2019.10.23

1

TỔNG QUAN

Nội dung chương này sẽ trình bày tổng quan về lõi IP X2P

- Đặc điểm
- Cấu hình X2P
- Giao tiếp
- Sơ đồ khối
- Hoạt động

1.1 Đặc điểm

Lõi IP X2P là cầu chuyển đổi từ giao thức AXI4 sang APB4 và có các đặc điểm sau:

- Hỗ trợ 1 master AXI có thể kết nối tới nhiều APB slave. Số lượng slave APB lên tới 32.
- Mỗi slave APB có độ rộng vùng nhớ là 4 KB.
- Cả 2 giao thức AXI và APB đều sử dụng chung 1 nguồn xung clock.
- Hỗ trợ truyền các transfer có kích thước 32 bit.
- Giao thức AXI và APB trong bản thiết kế này cùng sử dụng một nguồn clock, không hỗ trợ việc sử dụng 2 nguồn clock khác nhau.
- Không hỗ trợ truyền các gói dữ liệu có độ rộng: 8, 16, 64, 128 bit.

1.2 Hướng dẫn sử dụng, cấu hình lõi IP X2P

Trước khi hướng dẫn cách sử dụng, cũng như cấu hình lõi IP X2P thì nhóm tác giả sẽ giải thích 1 số tham số được sử dụng trong thiết kế để người đọc có thể hiểu nhanh hơn.

1.2.1 Tham số SLAVE_NUM

Lõi IP X2P hỗ trợ việc kết nối 1 master AXI có thể kết nối tới nhiều slave APB. Số lượng slave APB là tham số có thể cấu hình được tùy vào mục đích sử dụng của người dùng. Do đó, nhóm tác giả sẽ khai báo tham số này bằng 1 **PARAMETER** với tên là **SLAVE_NUM**. Tham số này được khai báo trong file **x2p_parameter.h**

1.2.2 Địa chỉ của các slave APB

Địa chỉ của các slave APB cũng sẽ được khai báo bằng các tham số **PARAMETER**. Ví dụ, địa chỉ của slave APB_0 sẽ có địa chỉ bắt đầu và địa chỉ kết thúc. Địa chỉ bắt đầu của slave APB_0 sẽ được khai báo bằng tham số **A_START_SLAVE0** còn địa chỉ kết thúc của slave APB_0 sẽ được khai báo bằng tham số **A_END_SLAVE0**. Tương tự, những slave khác cũng sẽ được khai báo địa chỉ bắt đầu và địa chỉ kết thúc giống như slave APB_0. Các tham số này cũng sẽ được khai báo trong file **x2p_parameter.h**

Hình 1.1: Bảng địa chỉ của các SLAVE APB

SLAVE	Địa chỉ bắt đầu	Địa chỉ kết thúc
SLAVE0	32'h0000_1000	32'h0000_1FFF
SLAVE1	32'h0000_2000	32'h0000_2FFF
SLAVE2	32'h0000_3000	32'h0000_3FFF
SLAVE3	32'h0000_4000	32'h0000_4FFF

SLAVE4	32'h0000_5000	32'h0000_5FFF
SLAVE5	32'h0000_6000	32'h0000_6FFF
SLAVE6	32'h0000_7000	32'h0000_7FFF
SLAVE7	32'h0000_8000	32'h0000_8FFF
SLAVE8	32'h0000_9000	32'h0000_9FFF
SLAVE9	32'h0000_A000	32'h0000_AFFF
SLAVE10	32'h0000_B000	32'h0000_BFFF
SLAVE11	32'h0000_C000	32'h0000_CFFF
SLAVE12	32'h0000_D000	32'h0000_DFFF
SLAVE13	32'h0000_E000	32'h0000_EFFF
SLAVE14	32'h0000_F000	32'h0000_FFFF
SLAVE15	32'h0001_0000	32'h0001_0FFF
SLAVE16	32'h0001_1000	32'h0001_1FFF
SLAVE17	32'h0001_2000	32'h0001_2FFF
SLAVE18	32'h0001_3000	32'h0001_3FFF
SLAVE19	32'h0001_4000	32'h0001_4FFF
SLAVE20	32'h0001_5000	32'h0001_5FFF
SLAVE21	32'h0001_6000	32'h0001_6FFF
SLAVE22	32'h0001_7000	32'h0001_7FFF
SLAVE23	32'h0001_8000	32'h0001_8FFF
SLAVE24	32'h0001_9000	32'h0001_9FFF
SLAVE25	32'h0001_A000	32'h0001_AFFF
SLAVE26	32'h0001_B000	32'h0001_BFFF
SLAVE27	32'h0001_C000	32'h0001_CFFF
SLAVE28	32'h0001_D000	32'h0001_DFFF
SLAVE29	32'h0001_E000	32'h0001_EFFF
SLAVE30	32'h0001_F000	32'h0001_FFFF
SLAVE31	32'h0002_0000	32'h0002_0FFF

1.2.3 Địa chỉ của khối Register

Địa chỉ của khối Register cũng sẽ được khai báo bằng **PARAMETER**. Địa chỉ bắt đầu của khối Register sẽ được khai báo bằng tham số **A_START_REG = 32'h0000_0000**, địa chỉ kết thúc của khối Register được khai báo bằng tham số **A_END_REG = 32'h0000_0FFF**.

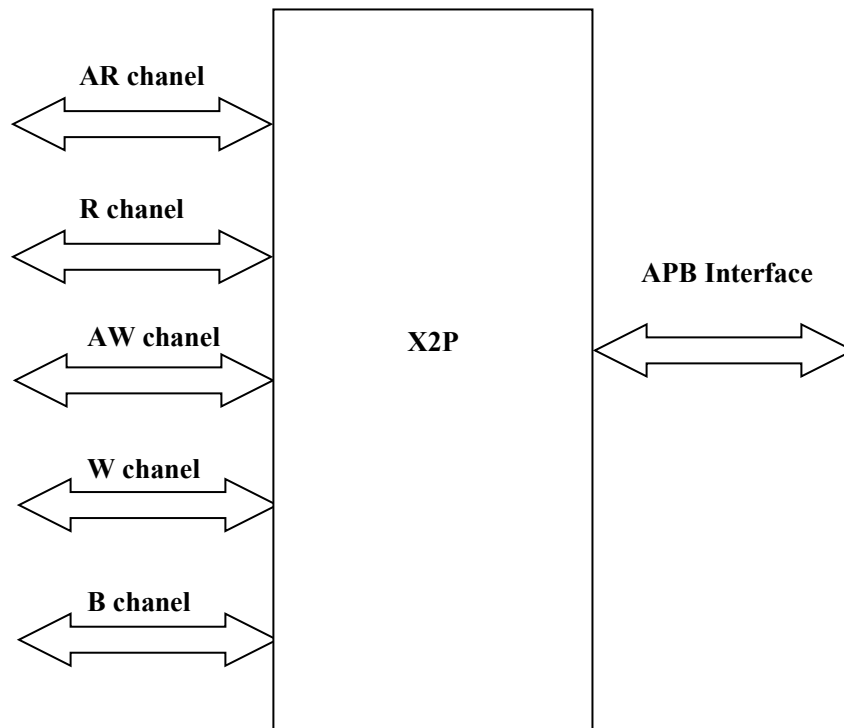
1.2.4 Cấu hình sử dụng lõi IP X2P

Sau khi hiểu về ý nghĩa của các tham số trên, nhóm tác giả sẽ đi giải thích cách cấu hình lõi **IP X2P** để người dùng có thể sử dụng.

Ví dụ, có 1 người dùng lõi IP X2P để kết nối với 3 slave APB là slave0, slave1, slave2. Khi đó người sử dụng lõi **IP X2P** sẽ thực hiện theo các bước sau:

- Bước 1: Mở file **x2p_parameter.h** lên.
- Bước 2: Định nghĩa tham số **SLAVE_NUM = 3**.

1.3 Giao tiếp



Hình 1.1: Sơ đồ giao tiếp tín hiệu của lõi IP X2P

AR channel: Address read channel, gọi tắt là kênh AR, là kênh dùng để truyền địa chỉ đọc và các thông tin điều khiển từ master đến slave.

R channel: Read data channel, gọi tắt là kênh R là kênh dùng để truyền dữ liệu đọc và thông tin phản hồi từ slave đến master.

AW channel: Address write channel, gọi tắt là kênh AW, là kênh dùng để truyền địa chỉ ghi và các thông tin điều khiển từ master đến slave.

W channel: Write data channel, gọi tắt là kênh W là kênh dùng để truyền dữ liệu ghi từ master đến slave.

B channel: Write respond channel, gọi tắt là kênh B là kênh dùng để truyền thông tin phản hồi của một transaction ghi từ slave đến master.

Bảng 1.1: Tín hiệu giao tiếp của lõi IP X2P

TT		Tín hiệu	Số bit	Chiều	Mô tả
1		aclk	1	Ngõ vào	Tín hiệu clock của giao thức AXI
2		aresetn	1	Ngõ vào	Tín hiệu reset của giao thức AXI Tích cực mức thấp
3	Kênh địa chỉ ghi	awvalid	1	Ngõ vào	Tín hiệu valid của kênh AW Khi tín hiệu này tích cực thì master sẽ thông báo cho slave biết các thông tin về địa chỉ ghi và thông tin điều khiển đã sẵn sàng.
4		awready	1	Ngõ ra	Tín hiệu ready của kênh AW Khi tín hiệu này tích cực thì slave sẽ thông báo cho master biết slave đã sẵn sàng nhận các thông tin về địa chỉ và thông tin điều khiển.
5		awaddr	32	Ngõ vào	Tín hiệu địa chỉ của kênh AW. Tín hiệu này chứa thông tin về địa chỉ của transfer đầu tiên trong 1 transaction
6		awsize	3	Ngõ vào	Tín hiệu điều khiển của kênh AW Dựa vào tín hiệu này ta xác định được kích thước của 1 transfer có bao nhiêu byte.

TT		Tín hiệu	Số bit	Chiều	Mô tả
7		awlen	8	Ngõ vào	Tín hiệu điều khiển của kênh AW Dựa vào tín hiệu này ta xác định được kích thước của 1 transaction ghi có bao nhiêu transfer.
8		awburst	2	Ngõ vào	Tín hiệu điều khiển của kênh AW Dựa vào tín hiệu này sẽ xác định được loại burst, từ đó tính được địa chỉ của các transfer dữ liệu tiếp theo trong transaction ghi.
9		awid	8	Ngõ vào	Tín hiệu này xác định ID của 1 transaction ghi. Mỗi transaction ghi sẽ có 1 ID để phân biệt với các transaction ghi khác.
10		awprot	3	Ngõ vào	Tín hiệu này định nghĩa quyền truy nhập của transaction ghi.
11	Kênh địa chỉ đọc	arvalid	1	Ngõ vào	Tín hiệu valid của kênh AR. Khi tín hiệu này tích cực thì master muốn thông báo cho slave biết thông tin địa chỉ đọc và thông tin điều khiển đã sẵn sàng.
12		arready	1	Ngõ ra	Tín hiệu ready của kênh AR Khi tín hiệu này tích cực thì slave muốn thông báo cho master biết slave đã sẵn sàng nhận thông tin địa chỉ đọc và thông tin điều khiển.
13	Kênh địa chỉ đọc	araddr	32	Ngõ vào	Tín hiệu địa chỉ của kênh AR Tín hiệu này chứa thông tin về địa chỉ của transfer đầu tiên trong 1 transaction.

TT		Tín hiệu	Số bit	Chiều	Mô tả
14		arsize	3	Ngõ vào	Tín hiệu điều khiển của kênh AR Dựa vào tín hiệu này ta xác định được 1 transfer có bao nhiêu byte
15		arlen	8	Ngõ vào	Tín hiệu điều khiển của kênh AR Dựa vào tín hiệu này ta xác định được 1 transaction có bao nhiêu transfer
16		arburst	2	Ngõ vào	Tín hiệu điều khiển của kênh AR Dựa vào tín hiệu này sẽ xác định được loại burst, từ đó tính được địa chỉ của các transfer dữ liệu trong transaction đọc.
17		arid	8	Ngõ vào	Tín hiệu này xác định ID của transaction đọc. Mỗi transaction sẽ có 1 ID.
18		arprot	3	Ngõ vào	Tín hiệu này định nghĩa quyền truy nhập của transaction.
19	Kênh dữ liệu ghi	wvalid	1	Ngõ vào	Tín hiệu valid của kênh W Khi tín hiệu này tích cực thì thông báo cho slave biết là dữ liệu ghi đã sẵn sàng để truyền
20		wready	1	Ngõ ra	Tín hiệu ready của kênh W Khi tín hiệu này tích cực thì slave muốn thông báo cho master biết đã sẵn sàng nhận dữ liệu từ master.
21		wdata	32	Ngõ vào	Tín hiệu data của kênh W
22		wstrb	4	Ngõ vào	Tín hiệu điều khiển byte trong 1 transfer

TT		Tín hiệu	Số bit	Chiều	Mô tả
23		wlast	1	Ngõ vào	Khi transfer cuối cùng được truyền cho slave thì master sẽ tích cực tín hiệu wlast lên mức 1 để báo cho slave biết đó là transfer cuối cùng trong 1 transaction.
24	Kênh dữ liệu đọc	rvalid	1	Ngõ ra	Tín hiệu valid của kênh R. Khi tín hiệu này tích cực thì slave sẽ thông báo cho master biết dữ liệu mà master muốn đọc đã sẵn sàng.
25		rready	1	Ngõ vào	Tín hiệu ready của kênh R. Khi tín hiệu này tích cực thì master sẽ thông báo cho slave biết đã sẵn sàng nhận dữ liệu đọc từ phía slave.
26		rlast	1	Ngõ ra	Khi truyền transfer cuối cùng trong 1 transaction đọc thì slave sẽ tích cực tín hiệu này lên để báo cho master biết đó là transfer cuối cùng trong truy nhập đọc.
27		rresp	2	Ngõ ra	Slave sẽ thông báo cho master biết về trạng thái của các tranfer đọc. Dựa vào tín hiệu này ta xác định được dữ liệu đọc có bị lỗi hay không.
28		rid	8	Ngõ ra	Khi master thực hiện 1 truy nhập đọc thì truy nhập đọc đó sẽ có 1 ID để phân biệt với các truy nhập khác. Khi dữ liệu đọc được trả về cho master thì các dữ liệu đó đều có thông tin ID để giúp master phân biệt được dữ liệu đọc đó thuộc truy nhập nào.
29		rdata	32	Ngõ ra	Dữ liệu đọc.

TT		Tín hiệu	Số bit	Chiều	Mô tả
30	Kênh đáp ứng ghi	bvalid	1	Ngõ ra	Khi tín hiệu này tích cực thì slave sẽ thông báo cho master biết slave đã sẵn sàng để truyền đi thông tin phản hồi về transaction ghi.
31		bready	1	Ngõ vào	Khi tín hiệu này tích cực thì master sẽ báo cho slave biết master đã sẵn sàng tiếp nhận thông tin phản hồi.
32		bresp	2	Ngõ ra	Dữ liệu trong tín hiệu này sẽ thông báo cho master biết trạng thái của transaction ghi.
33		bid	8	Ngõ ra	Khi slave gửi thông tin phản hồi về transaction ghi cho master thì master phải biết được thông tin phản hồi đó là của transaction ghi nào thì thông tin nằm trong tín hiệu bid sẽ giúp master phân biệt được điều đó. Vì mỗi transaction ghi đều có 1 ID để phân biệt
34	Giao thức APB	pclk	1	Ngõ vào	Tín hiệu xung clock của giao thức APB
35		preset_n	1	Ngõ vào	Tín hiệu reset của giao thức APB. Tích cực mức thấp
36		paddr	32	Ngõ ra	Tín hiệu địa chỉ Tín hiệu này chứa thông tin địa chỉ đọc hoặc ghi
37		pwwdata	32	Ngõ ra	Tín hiệu dữ liệu Tín hiệu này chứa dữ liệu cần truyền.
38		psel	SLAVE_NUM	Ngõ ra	Mỗi tín hiệu psel sẽ chỉ tới 1 slave APB. Tín hiệu này sẽ cho phép slave APB đó hoạt động hay không hoạt động. ✓ Trong thiết kế X2P này, psel[0] là tín

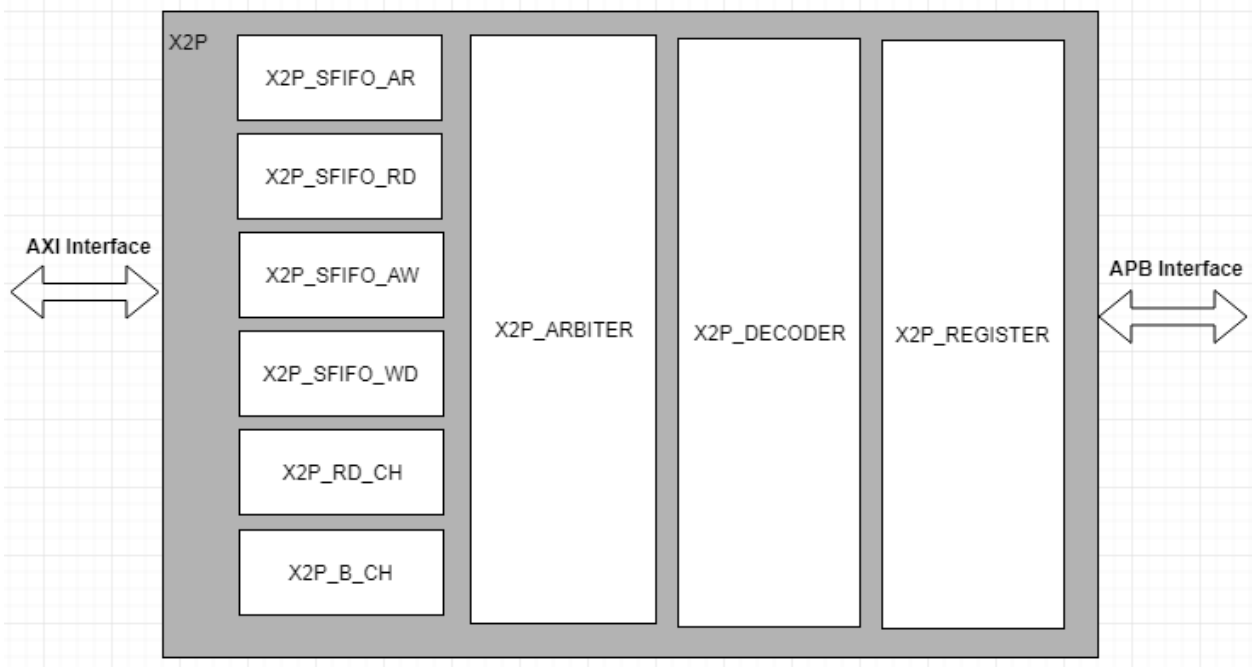
X2P IP core

					<p>hiệu điều khiển việc chọn khối X2P_REGISTER.</p> <p>✓ psel[1] đến psel[SLAVE_NUM] là tín hiệu điều khiển việc chọn các slave tương ứng.</p>
39		penable	1	Ngõ ra	Tín hiệu cho phép
40		pprot	3	Ngõ ra	Tín hiệu này thể hiện quyền truy cập của transfer.
41		pready	SLAVE_NUM	Ngõ vào	<p>Khi tín hiệu này tích cực thì slave sẽ thông báo cho master biết slave đã sẵn sàng nhận dữ liệu.</p> <p>✓ pready[0] là tín hiệu thông báo cho master biết khối X2P_REGISTER đã sẵn sàng nhận yêu cầu từ master hay chưa.</p> <p>✓ pready[1] đến pready[SLAVE_NUM] là tín hiệu thông báo trạng thái sẵn sàng nhận dữ liệu từ master của các slave tương ứng.</p>
42		pstrb	4	Ngõ ra	Tín hiệu này có chức năng điều khiển các byte trong 1 transfer.
43		pwrite	1	Ngõ ra	<p>Tín hiệu này có chức năng phân biệt truy nhập.</p> <p>pwrite = 1 là truy nhập ghi</p> <p>pwrite = 0 là truy nhập đọc</p>
44		prdata	32	Ngõ vào	Dữ liệu đọc được gửi từ slave APB
45		pslverr	SLAVE_NUM	Ngõ vào	<p>Tín hiệu này có chức năng thông báo cho APB master biết trạng thái của transfer có lỗi hay không lỗi.</p> <p>✓ pslverr[0] là tín hiệu</p>

X2P IP core

					<p>thông báo lỗi của khối X2P_REGISTER.</p> <p>✓ pslverr[1] đến pslverr[SLAVE_NUM] là tín hiệu thông báo trạng thái lỗi của các slave APB tương ứng.</p>
--	--	--	--	--	--

1.4 Sơ đồ khối



Hình 1.2: Sơ đồ khối của lõi IP X2P

Lõi IP X2P gồm các khối chức năng sau:

- X2P_SFIFO_AR: Là 1 FIFO dùng để lưu các địa chỉ đọc của transfer đầu tiên và các thông tin điều khiển như: ARSIZE, ARLEN, ARBURST, ARPROT, ARID.
- X2P_SFIFO_RD: Là 1 FIFO dùng để lưu dữ liệu đọc PRDATA, thông tin phản hồi Respond, tín hiệu Rlast, và ID của transaction đọc. Trong giao thức AXI, khi thực hiện 1 truy nhập đọc thì sẽ gồm 2 bước sau:
 - ✓ Phía Master sẽ gửi thông tin địa chỉ và thông tin điều khiển trên kênh AR để khởi động 1 truy nhập đọc.
 - ✓ Phía Slave sẽ gửi dữ liệu kèm theo thông tin phản hồi và ID của transaction cho phía Master. Chính vì lý do: Dữ liệu, thông tin phản hồi, và ID của transaction sẽ được gửi đồng thời nên dữ liệu đọc, thông tin phản hồi, và ID sẽ được lưu chung vào 1 FIFO.
- X2P_SFIFO_AW: Là 1 FIFO dùng để lưu địa chỉ ghi của transfer đầu tiên và các thông tin điều khiển như: AWSIZE, AWLEN, AWBURST, AWPROT, AWID.
- X2P_SFIFO_WD: Là 1 FIFO dùng để lưu dữ liệu ghi WDATA và tín hiệuWSTRB.

- X2P_RD_CH: Khối này dùng để xác định các tín hiệu RLAST và RRESP của truy nhập đọc.
- X2P_B_CH: Khối này dùng để xác định các tín hiệu BRESP và BID của truy nhập ghi.
- X2P_ARBITER: Khối này có chức năng phân xử trong trường hợp có 2 truy nhập đọc và ghi cùng thực hiện 1 lúc. Khi đó bộ phân xử ARBITER sẽ phân xử xem truy nhập nào sẽ được thực hiện.
- X2P_DECODER: Khối này có chức năng sẽ giải mã xem Master yêu cầu truy nhập đến slave nào.
- X2P_REGISTER: Khối này chứa các thanh ghi read_only, các thanh ghi này sẽ chứa địa chỉ của các slave. Khi Master muốn biết thông tin về địa chỉ của các slave thì Master sẽ phát đi 1 truy nhập đọc có địa chỉ nằm trong tầm vực của khối X2P_REGISTER.

1.5 Hoạt động của X2P

- ✓ Khi reset thì X2P sẽ xử lý như sau:
 - Dữ liệu lưu trong các FIFO sẽ bị reset toàn bộ.
 - Sau khi tín hiệu reset thôi tích cực, nếu có 2 transaction đọc và ghi cùng tới 1 lúc, thì X2P sẽ ưu tiên transaction đọc.
- ✓ Khi 1 transaction truy nhập vào vùng reserved:
 - Nếu là transaction ghi thì X2P sẽ trả lại tín hiệu phản hồi trên kênh B là DECERR. Nếu là transaction đọc thì X2P sẽ trả lại tín hiệu phản hồi trên kênh R là DECERR và RDATA = 0.
- ✓ Khi X2P hoạt động bình thường, tức là không bị reset thì X2P sẽ đối xử với truy nhập đọc và ghi là như nhau. Các truy nhập đọc sẽ có cơ hội ngang nhau. Còn trong trường hợp reset thì X2P sẽ đối xử như trên.

2

PHÂN TÍCH THIẾT KẾ

Nội dung chương này trình bày chi tiết về cách thiết kế lõi IP X2P

- Phương pháp
- Phân tích tổng quan
- Phân tích chi tiết

2.1 Phương pháp thiết kế

Phương pháp thiết kế được áp dụng là top_down, phân tích từ tổng quan đến chi tiết. Thiết kế sẽ được phân tích từ tổng quát, từ top module, đến mức chi tiết hóa bởi những phần tử cơ bản. Những phần tử cơ bản là những phần tử là người mô tả RTL có thể hiểu và mô hình hóa được bằng ngôn ngữ mô tả phần cứng. Trong thiết kế số, các phần tử cơ bản là các cổng logic, Flip Flop, latch, FSM, ...

2.2 Phân tích tổng quan lõi IP X2P

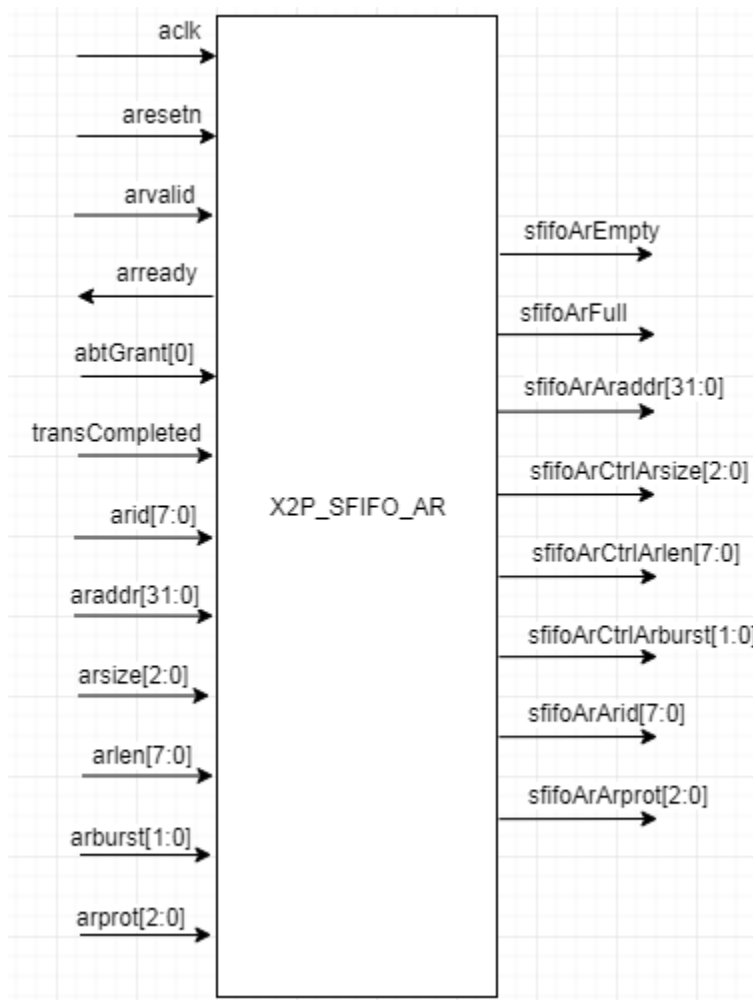
- Các khối FIFO sử dụng trong lõi IP X2P sẽ được sử dụng lại. Các khối FIFO này đã thiết kế sẵn, việc còn lại là cấu hình và kết nối các tín hiệu để sử dụng.
- Khối Arbiter sử dụng trong thiết kế này được thiết kế với đặc điểm như sau:
 - Tính linh động cao, có thể sử dụng lại. Trong thiết kế lõi IP X2P trong phiên bản này bộ phân xử Arbiter chỉ phân xử 2 hoạt động đọc và ghi đến từ 1 Master. Tuy nhiên trong trường hợp khi thiết kế các lõi IP khác, nếu bộ phân xử Arbiter phải phân xử nhiều hoạt động đọc ghi đến từ nhiều Master khác nhau thì bộ phân xử trong thiết kế này vẫn có thể được sử dụng lại.
- Khối Decoder sẽ dựa vào tín hiệu địa chỉ gửi từ Master AXI sẽ giải mã để lựa chọn ra slave APB mà Master AXI muốn truy nhập.
- Khối Register sẽ chứa các thanh ghi read_only, các thanh ghi này sẽ chứa thông tin địa chỉ của các slave. Khối này được thiết kế với mục đích để khi Master muốn biết thông tin của các slave thì sẽ truy cập vào khối này để lấy thông tin địa chỉ.

2.3 Phân tích kết nối khối X2P_SFIFO_AR

Khối X2P_SFIFO_AR dùng để lưu các thông tin sau:

- Thông tin về địa chỉ của transfer đầu tiên trong transaction đọc, thông tin đó chứa trong tín hiệu **araddr**[31:0]
- Các thông tin điều khiển sau:

- **Arsize[2:0]**: Thông tin này dùng để xác định kích thước của 1 transfer. Giả sử **Arsize[2:0] = 2** thì kích thước của transfer sẽ là $2^{\text{Arsize}} = 4$ byte.
- **Arlen[7:0]**: Thông tin này dùng để xác định kích thước của transaction có bao nhiêu transfer. Giả sử **Arlen = 7**, thì kích thước của transaction sẽ là $1 + \text{Arlen} = 8$ (transfers).
- **Arburst[1:0]**: Thông tin này dùng để xác định loại burst mà AXI hỗ trợ.
- **Arid[7:0]**: Tín hiệu này chứa thông tin ID của transaction.
- **Arprot[2:0]**: Tín hiệu này chứa thông tin về quyền truy nhập của transaction



Hình 2.3: Sơ đồ kết nối của khối X2P_SFIFO_AR

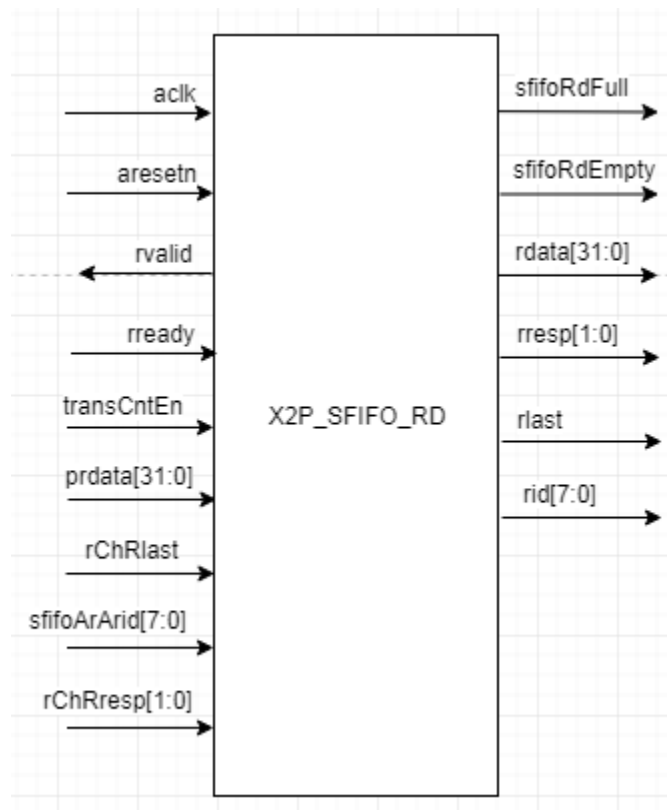
Chức năng của các tín hiệu khác trong khối **X2P_SFIFO_AR**:

- **transCompleted**: Tín hiệu này được sinh ra từ khối Decoder. Tín hiệu này sẽ tích cực ở 1 chu kỳ báo hiệu rằng 1 truy nhập đã được hoàn thành. Tín hiệu này dùng để xác định khi nào đọc FIFO.
- **abtGrant[0]**: Tín hiệu này sẽ được sinh ra từ khối Arbiter. Tín hiệu này tích cực thì sẽ báo hiệu bộ phân xử đã phân xử cho truy nhập đọc. Tín hiệu này dùng để xác định khi nào đọc FIFO.
- **sfifoArEmpty**: Tín hiệu này báo trạng thái rỗng của FIFO.
- **sfifoArFull**: Tín hiệu này báo trạng thái đầy của FIFO.
- **sfifoArAraddr[31:0]**: Tín hiệu này chứa thông tin về địa chỉ của transfer đầu tiên của transaction đọc.
- **sfifoArCtrlArsize[2:0]**: Tín hiệu này chứa thông tin về kích thước của 1 transfer.
- **sfifoArArlen[7:0]**: Tín hiệu này chứa thông tin về kích thước của 1 transaction đọc.
- **sfifoArArburst[1:0]**: Tín hiệu này chứa thông tin về loại burst sử dụng.
- **sfifoArArid[7:0]**: Tín hiệu này chứa thông tin về ID của transaction đọc.

2.4 Phân tích kết nối khối X2P_SFIFO_RD

Nhiệm vụ của khối **X2P_SFIFO_RD** dùng để lưu các thông tin sau:

- Dữ liệu đọc từ phía slave APB trả về.
- Thông tin phản hồi của các transfer dữ liệu đọc.
- Thông tin về ID của transaction đọc.
- Ngoài ra còn để lưu thông tin về tín hiệu **RLAST** báo transfer cuối cùng của transaction.



Hình 2.4: Sơ đồ kết nối của khối X2P_SFIFO_RD

Chức năng của các tín hiệu:

- **transCntEn**: Tín hiệu này được sinh ra từ khối Decoder. Tín hiệu này dùng để xác định thời điểm ghi vào FIFO.
- **prdata[31:0]**: Dữ liệu đọc từ slave APB.
- **sfifoArArid[7:0]**: Tín hiệu này được sinh ra từ khối X2P_SFIFO_AR. Tín hiệu này chứa thông tin ID của transaction đọc.
- **rChRresp**: Tín hiệu này được sinh ra từ khối X2P_R_CH. Tín hiệu này chứa thông tin phản hồi của từng transfer.
- **rChRlast**: Tín hiệu này được sinh ra từ khối X2P_R_CH. Tín hiệu này được dùng để xác định tín hiệu **RLAST**.
- **sfifoRdFull**: Tín hiệu này báo trạng thái đầy của FIFO.
- **sfifoRdEmpty**: Tín hiệu này báo trạng thái rỗng của FIFO.
- **rdata[31:0]**: Dữ liệu đọc.
- **rresp[1:0]**: Tín hiệu phản hồi về trạng thái của transfer.
- **rlast**: Tín hiệu báo transfer cuối cùng trong 1 transaction đọc.
- **rid[7:0]**: Tín hiệu này chứa ID của transaction đọc.

2.5 Phân tích kết nối của khối X2P_SFIFO_AW

Chức năng của khối này cũng tương tự như khối X2P_SFIFO_AR. Dùng để lưu thông tin về địa chỉ của transfer đầu tiên của transaction ghi và các thông tin điều khiển. Về ý nghĩa của các tín hiệu thì nó cũng tương tự như khối X2P_SFIFO_AR nên ta sẽ không phân tích chi tiết thêm nữa.

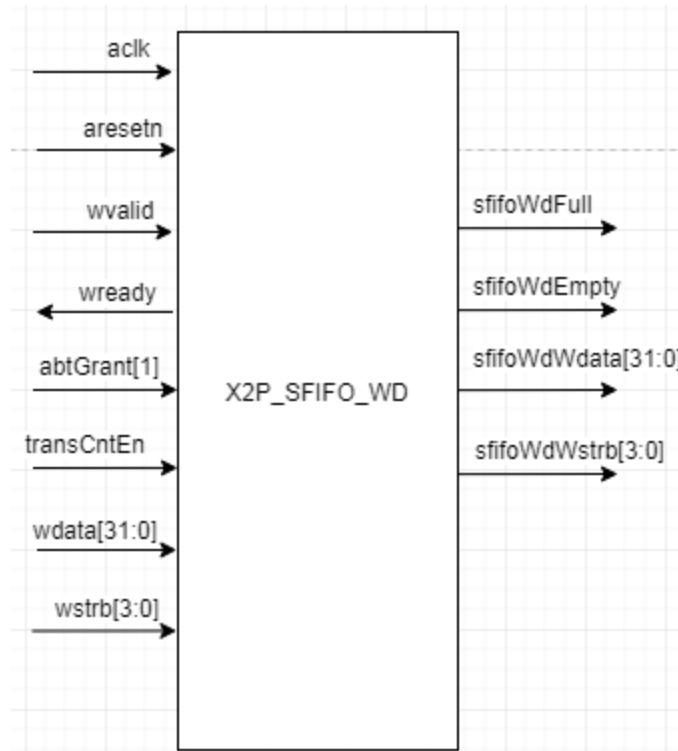


Hình 2.5: Sơ đồ kết nối của khối X2P_SFIFO_AW

2.6 Phân tích kết nối khối X2P_SFIFO_WD

Khối X2P_SFIFO_WD dùng để lưu các thông tin sau:

- Lưu dữ liệu ghi **wdata[31:0]** trên kênh W
- Lưu tín hiệu **wstrb[3:0]**



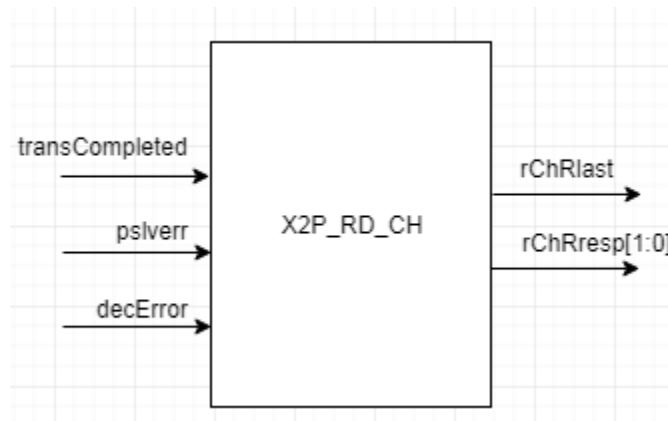
Hình 2.6: Sơ đồ kết nối khối X2P_SFIFO_WD

Ý nghĩa của các tín hiệu:

- **abtGrant[1]**: Tín hiệu này được sinh ra từ khối Arbiter. Tín hiệu này dùng để xác định thời điểm đọc dữ liệu từ FIFO ra.
- **transCntEn**: Tín hiệu này được sinh ra từ khối Decoder. Tín hiệu này dùng để xác định thời điểm đọc dữ liệu từ FIFO ra.
- **sfifoWdFull**: Tín hiệu này báo trạng thái đầy của FIFO.
- **sfifoWdEmpty**: Tín hiệu này báo trạng thái rỗng của FIFO.
- **sfifoWdWdata[31:0]**: Tín hiệu này chứa dữ liệu ghi.
- **sfifoWdWstrb[3:0]**: Tín hiệu này chứa thông tin về số byte hợp lệ trong 1 transfer.
- Những tín hiệu còn lại đã được giải thích ở phần trên.

2.7 Phân tích kết nối khối X2P_RD_CH

Khối **X2P_RD_CH** dùng để xác định 2 tín hiệu **rChRlast** và tín hiệu **rChRresp[1:0]**. Hai tín hiệu này sau đó sẽ được lưu vào khối **X2P_SFIFO_RD**.



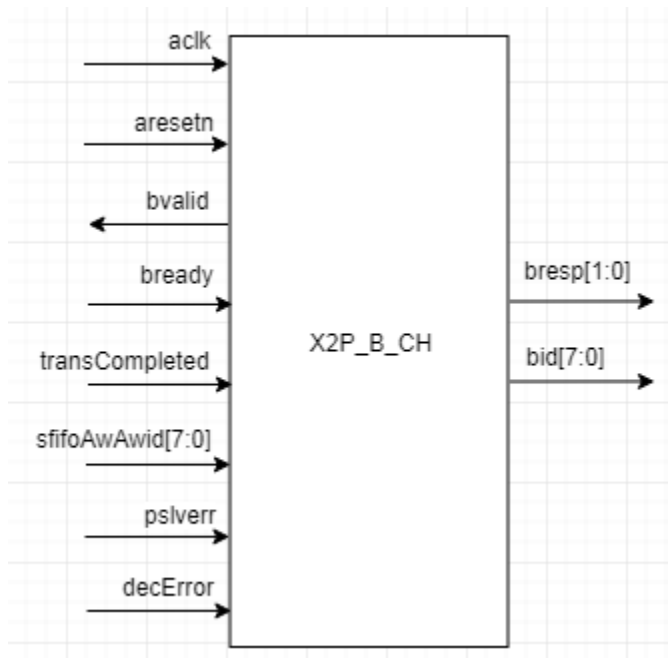
Hình 2.7: Sơ đồ kết nối của khối X2P_RD_CH

Ý nghĩa của các tín hiệu:

- **transCompleted:** Tín hiệu này báo hiệu 1 transaction hoàn thành. Tín hiệu này dùng để xác định tín hiệu **rChRlast**.
- **decError:** Tín hiệu này báo hiệu việc giải mã địa chỉ bị lỗi. Địa chỉ mà Master gửi tới không thuộc tầm địa chỉ của các Slave và khối Register.

2.8 Phân tích kết nối của khối X2P_B_CH

Khối này có chức năng là xác định tín hiệu **bresp[1:0]**, **bid[7:0]**, **bvalid**



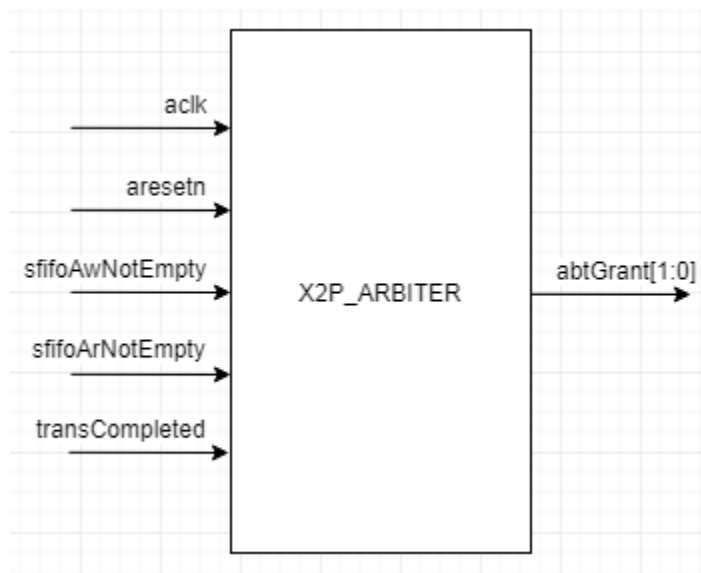
Hình 2.8: Sơ đồ kết nối của khối X2P_B_CH

X2P IP core

Ý nghĩa của những tín hiệu:

- **transCompleted**: Khác với truy nhập đọc các tín hiệu ID và respond sẽ được gửi cùng transfer trên kênh R. Trong truy nhập ghi, sau khi 1 truy nhập hoàn thành, tức là khi có tín hiệu **transCompleted** tích cực trong 1 chu kỳ thì các tín hiệu ID và respond mới được phát trên kênh B.
- **bvalid**: Tín hiệu **bvalid** này tích cực thì sẽ báo hiệu cho Master biết thông tin phản hồi về truy nhập ghi đang sẵn sàng để truyền. Do đó, **bvalid** sẽ tích cực trong 1 chu kỳ sau khi có tín hiệu **transCompleted** tích cực.
- Các tín hiệu còn lại đều đã được giải thích ở các phần trên.

2.9 Phân tích sơ đồ kết nối của khối X2P_ARBITER



Hình 2.9: Sơ đồ kết nối của khối X2P_ARBITER

Bộ Arbiter trong thiết kế này sẽ có chức năng chính là phân xử truy nhập đọc và ghi. Tuy nhiên như đã nói ở phần trước, bộ Arbiter sẽ được thiết kế sao cho có độ linh động cao, tức là có khả năng tái sử dụng lại. Trong thiết kế lõi X2P này chỉ có 2 yêu cầu đọc và ghi phát ra từ 1 Master, tuy nhiên trong những thiết kế lõi IP khác có thể sẽ có nhiều yêu cầu truy nhập được phát ra từ nhiều Master khác nhau thì khi đó bộ Arbiter được sử dụng trong thiết kế này vẫn đáp ứng được. Chi tiết về cách thiết kế thì ta sẽ bàn kỹ ở các phần sau.

Ý nghĩa của các tín hiệu:

- **sfifoAwNotEmpty, sfifoArNotEmpty:** Hai tín hiệu này là 2 tín hiệu đầu vào của bộ Arbiter. Chỉ cần 2 khối FIFO là **X2P_SFIFO_AR** và **X2P_SFIFO_AW** có dữ liệu, tức là không rỗng thì sẽ đưa ra yêu cầu phân xử cho bộ Arbiter.
- Tín hiệu **transCompleted** cũng là đầu vào của khối Arbiter với ý nghĩa: Chỉ khi nào 1 transaction kết thúc, tức là khi **transCompleted** tích cực thì bộ Arbiter sẽ cập nhật lại quyền truy nhập của các nguồn.
- **abtGrant[1]** tương ứng với truy nhập ghi. **abtGrant[1]** tích cực thì có nghĩa là truy nhập ghi được phân quyền.
- **abtGrant[0]** tương ứng với truy nhập đọc. **abtGrant[0]** tích cực thì có nghĩa là truy nhập đọc được phân quyền.
- Các tín hiệu còn lại đã được giải thích ở các phần trên.

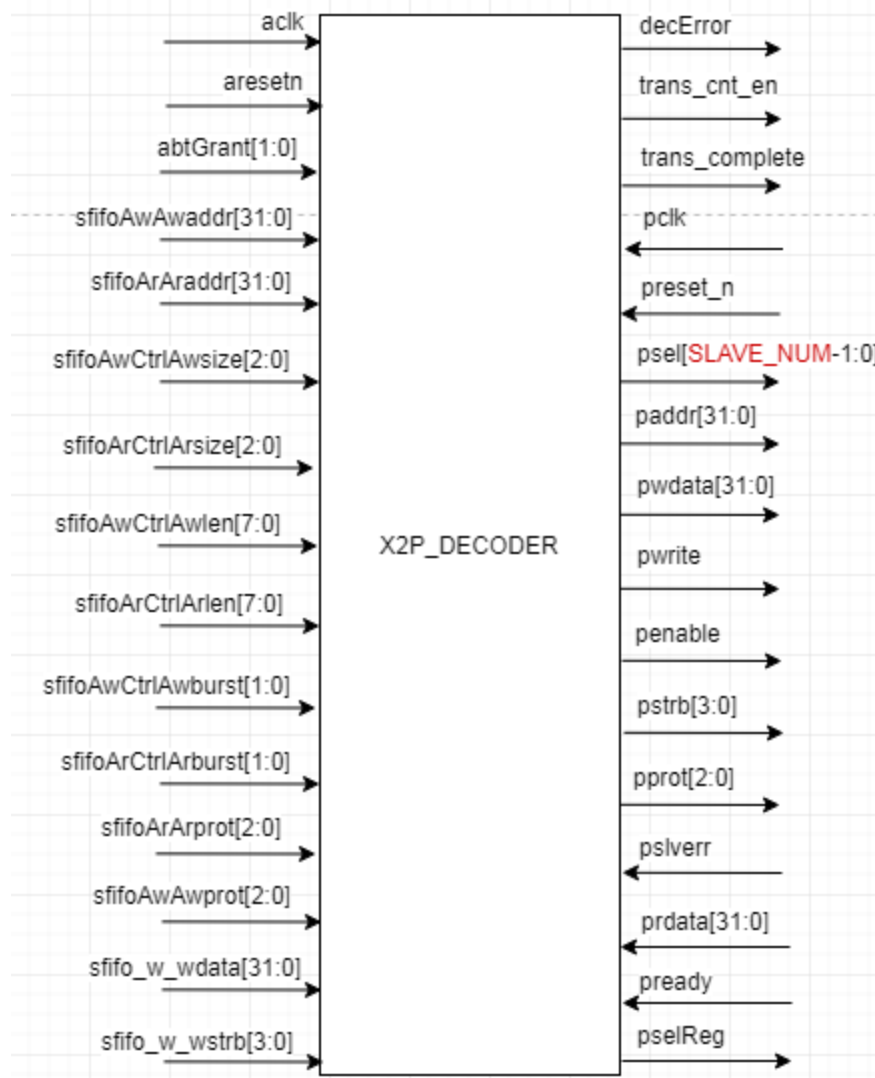
2.10 Phân tích sơ đồ kết nối của khối X2P_DECODER

Khối Decoder trong thiết kế này thực hiện các chức năng sau:

- Giải mã địa chỉ để chọn ra Slave APB mà Master muốn truy nhập tới.
- Lái các tín hiệu của giao thức APB.

Ý nghĩa của các tín hiệu:

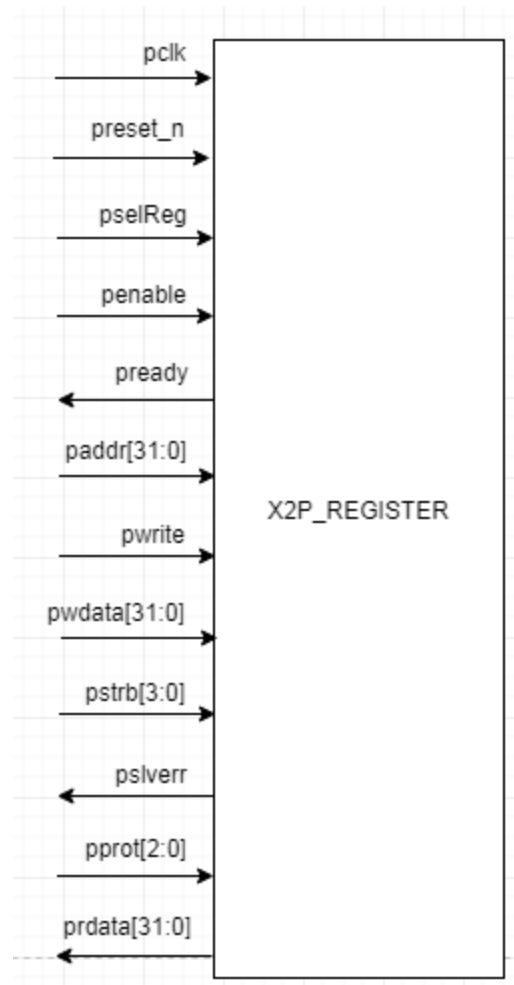
- **psel[SLAVE_NUM -1]:** Số lượng tín hiệu **psel** sẽ phụ thuộc vào số Slave APB. Giả sử có 10 Slave APB thì sẽ có 10 tín hiệu **psel**. Mỗi tín hiệu **psel** sẽ tương ứng với 1 Slave APB, nếu tín hiệu **psel** của Slave APB nào tích cực thì Slave APB đó sẽ được lựa chọn để truy nhập.
- **pselReg:** Tín hiệu này là tín hiệu điều khiển khối **REGISTER**. Khi tín hiệu **pselReg** tích cực thì khối **REGISTER** sẽ được lựa chọn để truy nhập.
- Các tín hiệu còn lại đã được giải thích ở trên.



Hình 2.10: Sơ đồ kết nối của khối Decoder

2.11 Phân tích sơ đồ kết nối của khối X2P_REGISTER

Khối **X2P_REGISTER** trong thiết kế lõi IP X2P bao gồm các thanh ghi read_only. Các thanh ghi này chứa địa chỉ của các Slave APB. Khi Master muốn biết về thông tin địa chỉ của các Slave APB thì sẽ truy nhập vào khối **X2P_REGISTER** để lấy thông tin. Địa chỉ của khối **X2P_REGISTER** và địa chỉ của các thanh ghi sẽ được phân tích chi tiết ở mục sau.



Hình 2.11: Sơ đồ kết nối của khối X2P_REGISTER

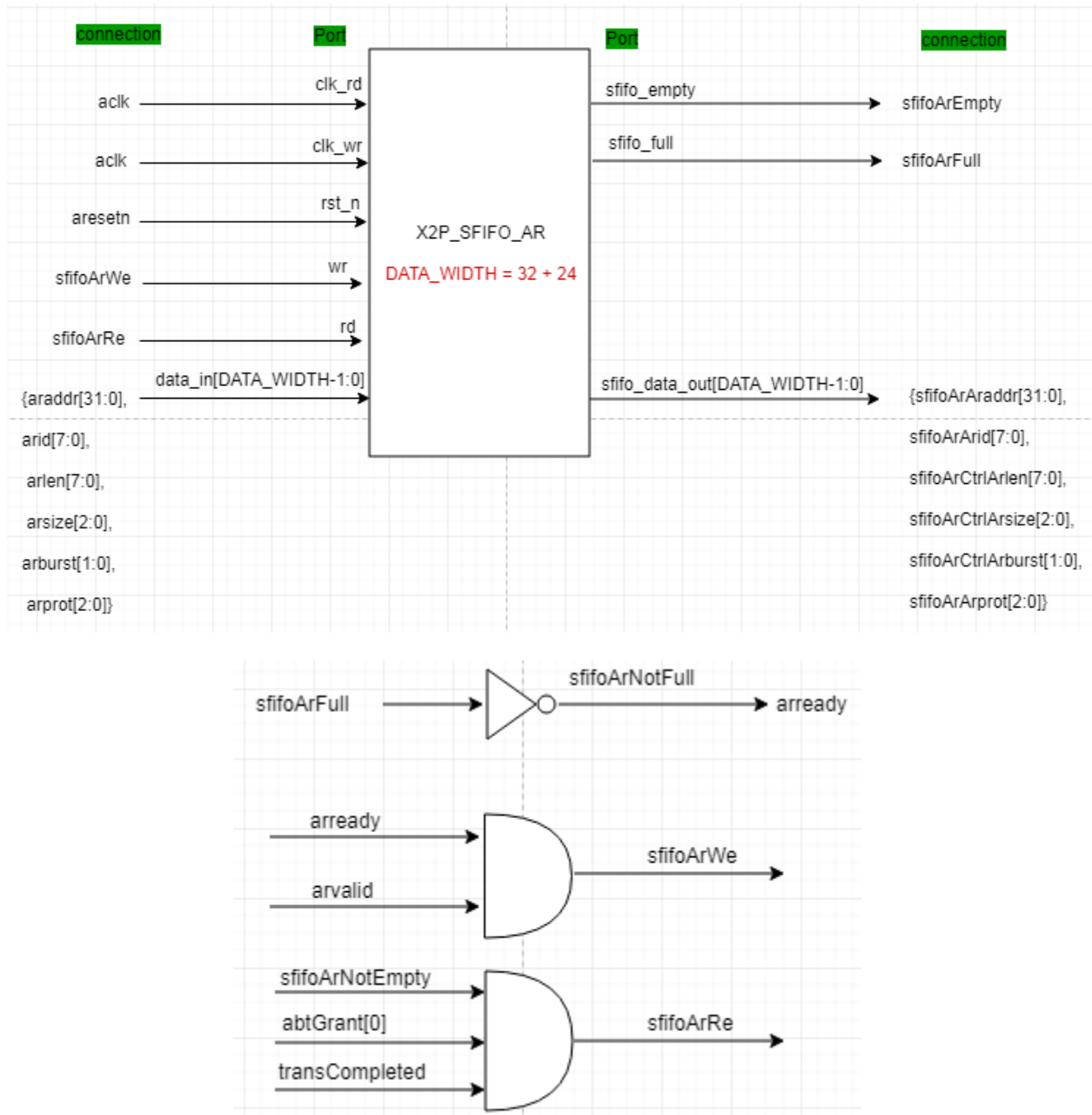
Ý nghĩa của các tín hiệu: Các tín hiệu của khối **X2P_REGISTER** đã được giải thích ở phần trên.

2.12 Bảng giá trị địa chỉ của các Slave APB và của khối Register

2.13 Phân tích chi tiết khối X2P_SFIFO_AR

Khối **X2P_SFIFO_AR** sẽ sử dụng khối SFIFO đã được thiết kế. Để tìm hiểu chi tiết về cách thiết kế SFIFO, các bạn tìm lại bài viết về cách thiết kế SFIFO ở trên blog để đọc. Ở đây ta sẽ chỉ cấu hình lại khối SFIFO đã thiết kế để được khối **X2P_SFIFO_AR**. Chức năng của khối này là lưu thông tin địa chỉ của transfer đầu tiên và các thông tin điều khiển.

X2P IP core



Hình 2.13: Sơ đồ nguyên lý của khối X2P_SFIFO_AR

- **sfifoArWe:** Tín hiệu này là tín hiệu điều khiển việc ghi vào X2P_SFIFO_AR. Chỉ được phép ghi vào X2P_SFIFO_AR khi thỏa mãn các điều kiện sau:
 - Khi X2P_SFIFO_AR không đầy, tức là khi **sfifoArNotFull** = 1. Khi X2P_SFIFO_AR không đầy thì tín hiệu **arready** sẽ được đẩy lên mức tích cực để báo hiệu cho Master biết sẵn sàng nhận dữ liệu. Dữ liệu ở đây là thông tin về địa chỉ của transfer đầu tiên và các thông tin điều khiển.
 - Khi tín hiệu **arvalid** tích cực.

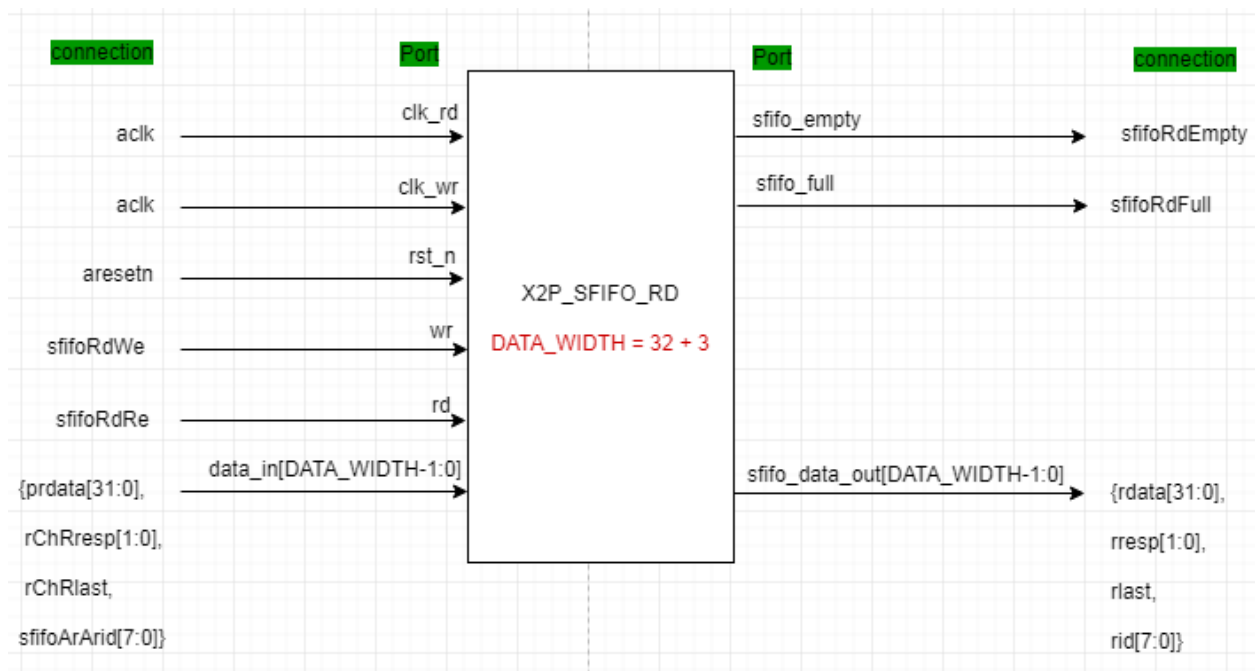
X2P IP core

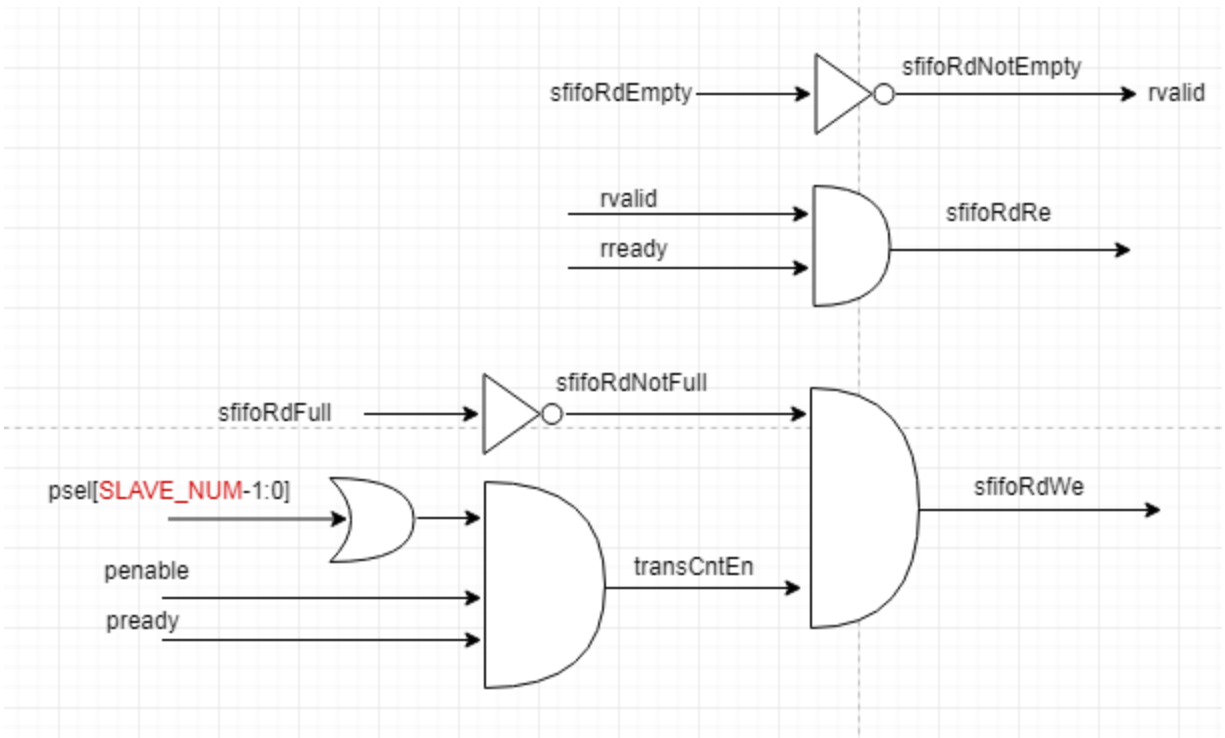
- **sfifoArRe**: Tín hiệu này điều khiển việc đọc dữ liệu ra từ **X2P_SFIFO_AR**. Chỉ được phép đọc dữ liệu ra khỏi **X2P_SFIFO_AR** khi thỏa mãn các điều kiện sau:
 - Khi truy nhập đọc hoàn thành, tức là khi tín hiệu `transCompleted = 1`.
 - Khi truy nhập đọc được phân quyền, tức là khi tín hiệu `abtGrant[0] = 1`.
 - Khi **X2P_SFIFO_AR** không rỗng.

2.14 Phân tích chi tiết khối X2P_SFIFO_RD

Khối **X2P_SFIFO_RD** cũng sẽ sử dụng lại khối SFIFO đã được thiết kế sẵn.

Khối **X2P_SFIFO_RD** dùng để lưu dữ liệu đọc, thông tin phản hồi `respond`, thông tin ID của transaction đọc và lưu tín hiệu `Rlast`.





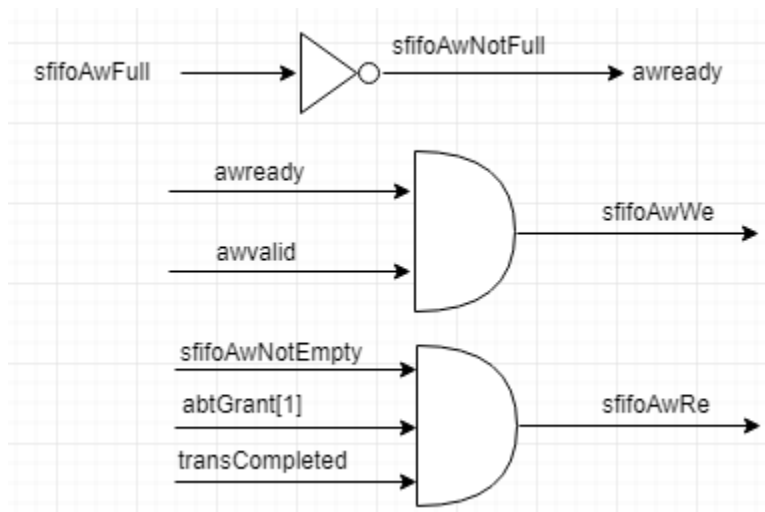
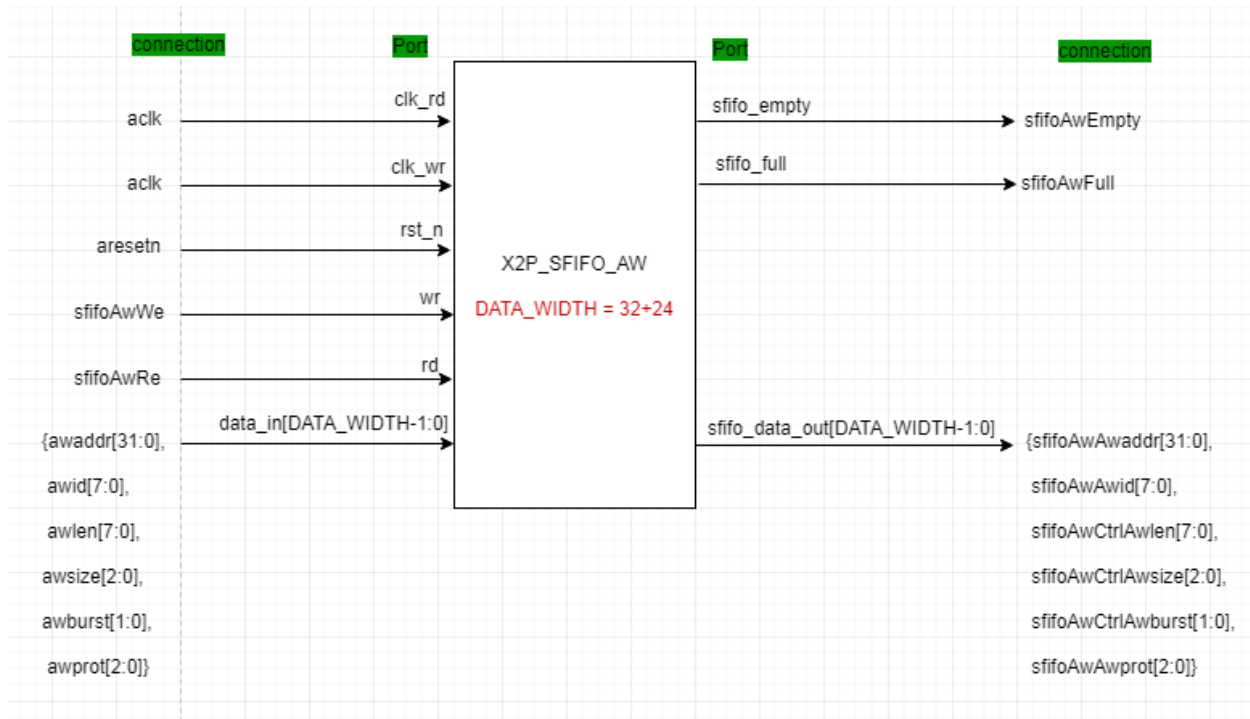
Hình 2.14: Sơ đồ nguyên lý của khối X2P_SFIFO_RD

- **sfifoRdWe:** Tín hiệu này điều khiển việc ghi vào khối X2P_SFIFO_RD. Chỉ được phép ghi vào X2P_SFIFO_RD khi thỏa mãn các điều kiện sau:
 - Khi X2P_SFIFO_RD không đầy, tức là tín hiệu **sfifoRdNotFull** = 1.
 - Khi các tín hiệu psel, penable, pready bằng 1, tức là tín hiệu **transCntEn** = 1.
- **sfifoRdRe:** Tín hiệu này điều khiển việc đọc dữ liệu từ khối X2P_SFIFO_RD. Chỉ được phép đọc dữ liệu khi thỏa mãn các điều kiện sau:
 - Khi X2P_SFIFO_RD không rỗng.
 - Khi tín hiệu **rready** tích cực.

2.15 Phân tích chi tiết khối X2P_SFIFO_AW

Khối X2P_SFIFO_AW cũng sẽ sử dụng lại khối SFIFO đã được thiết kế. Khối X2P_SFIFO_AW dùng để lưu địa chỉ của byte đầu tiên của truy nhập ghi và các thông tin điều khiển.

X2P IP core



Hình 2.15: Sơ đồ nguyên lý của khối X2P_SFIFO_AW

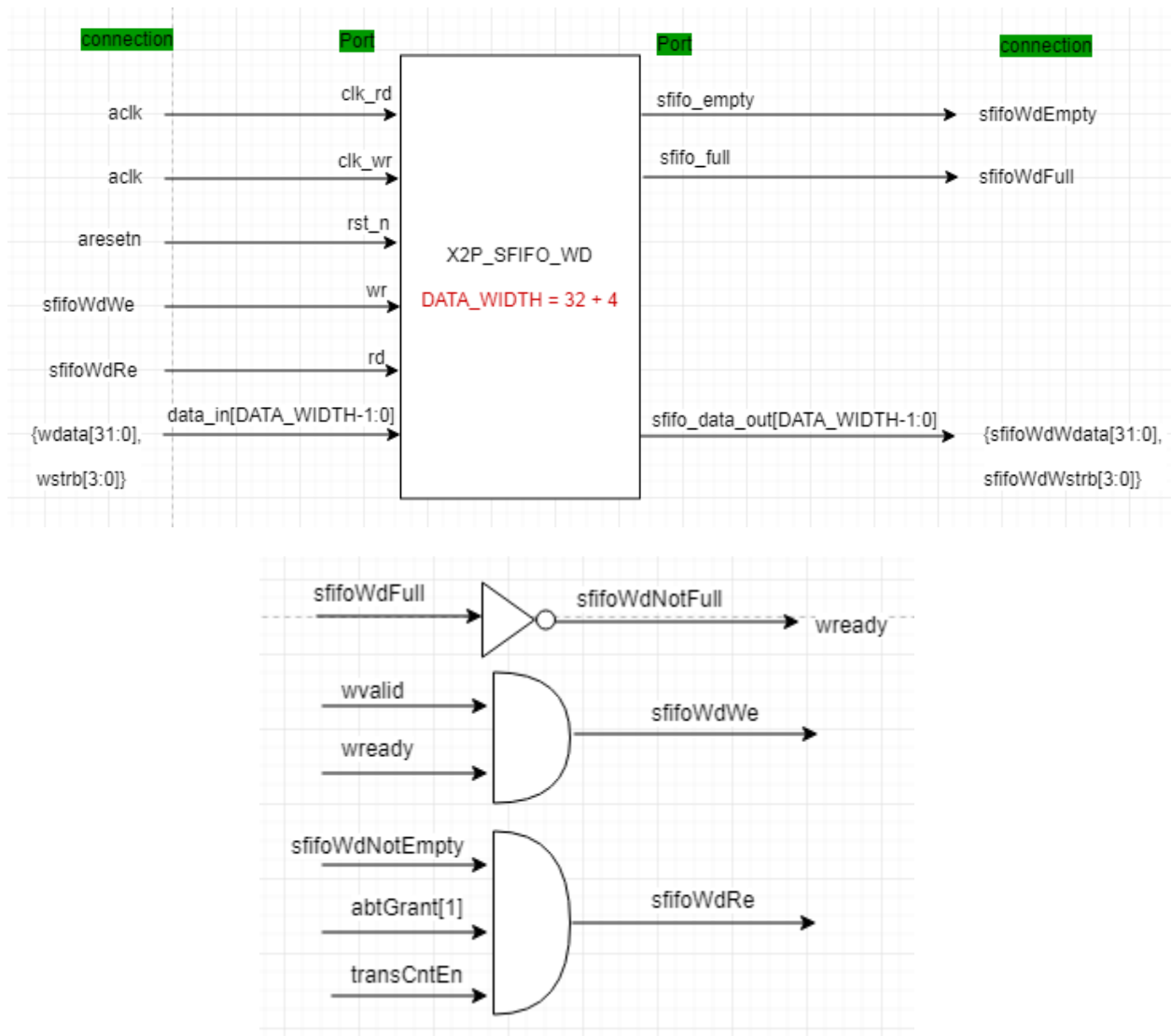
- **sfifoAwWe:** Tín hiệu này điều khiển việc ghi vào khối **X2P_SFIFO_AW**. Chỉ được phép ghi vào **X2P_SFIFO_AW** khi thỏa mãn các điều kiện sau:
 - Khi **X2P_SFIFO_AW** không đầy, tức là tín hiệu **sfifoAwNotFull** = 1.
 - Khi tín hiệu **awvalid** tích cực.
- **sfifoAwRe:** Tín hiệu này điều khiển việc đọc dữ liệu ra từ khối **X2P_SFIFO_AW**. Chỉ được phép đọc dữ liệu ra khi thỏa mãn các điều kiện sau:

X2P IP core

- Khi **X2P_SFIFO_AW** không rỗng, tức là khi tín hiệu **sfifoAwNotEmpty** = 1.
- Khi transaction ghi được phân quyền, tức là tín hiệu **abtGrant[1]** = 1.
- Khi 1 transaction ghi kết thúc, tức là khi tín hiệu **transCompleted** = 1.

2.16 Phân tích chi tiết khối X2P_SFIFO_WD

Khối **X2P_SFIFO_WD** có chức năng là dữ liệu ghi **wdata** trên kênh W và lưu tín hiệu **wstrb**



Hình 2.16: Sơ đồ nguyên lý của khối X2P_SFIFO_WD

- **sfifoWdWe**: Tín hiệu này có chức năng điều khiển việc ghi dữ liệu vào khối **X2P_SFIFO_WD**. Chỉ được phép ghi vào **X2P_SFIFO_WD** khi thỏa mãn các điều kiện sau:
 - Khi **X2P_SFIFO_WD** không đầy, tức là tín hiệu **sfifoWdNotFull** = 1;
 - Khi tín hiệu **wvalid** tích cực.
- **sfifoWdRe**: Tín hiệu này có chức năng điều khiển việc đọc dữ liệu từ khối **X2P_SFIFO_WD**. Chỉ được phép đọc dữ liệu từ khối **X2P_SFIFO_WD** khi thoãn mãn các điều kiện sau:
 - Khi **X2P_SFIFO_WD** không rỗng, tức là tín hiệu **sfifoWdNotEmpty** = 1.
 - Khi transaction ghi được phân quyền, tức là tín hiệu **abtGrant[1]** = 1.
 - Khi **psel**, **penable**, **pready** bằng 1, tức là tín hiệu **transCntEn** = 1.

2.18 Phân tích sơ đồ chi tiết của khối X2P_ARBITER

2.18.1 Phân tích các thiết kế bộ phân xử Arbiter

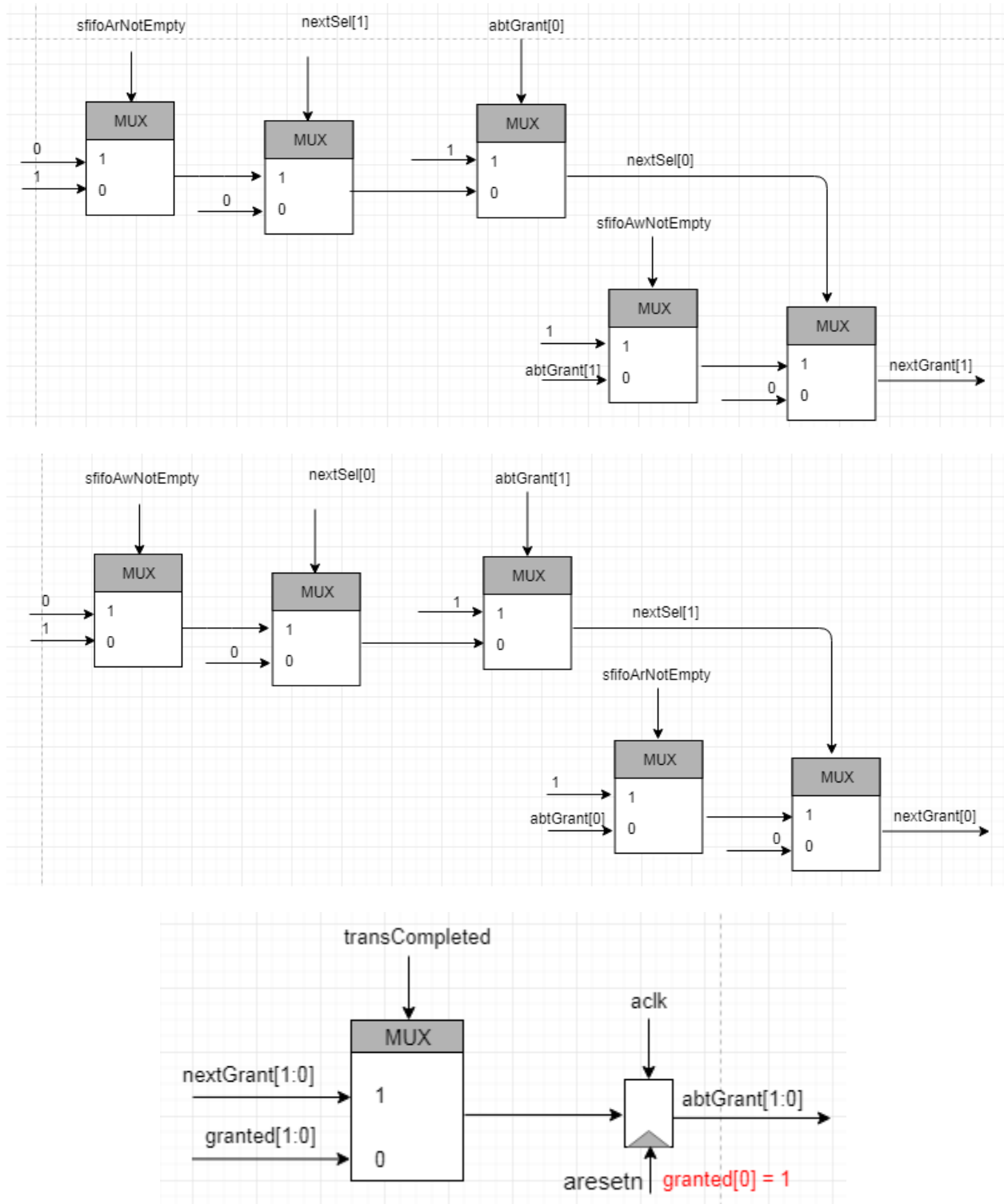
Bộ **Arbiter** trong thiết kế lõi IP X2P sẽ được thiết kế theo nguyên lý sau:

- Tại thời điểm sau khi reset thì truy nhập đọc sẽ được phân quyền truy cập.
- Khi đó trong lần phân quyền thứ 2, thì truy nhập ghi sẽ có quyền ưu tiên cao hơn. Tức là, trong lần phân quyền thứ 2 nếu có 2 truy nhập đọc và ghi cùng yêu cầu 1 lúc thì truy nhập ghi sẽ được phân quyền.
- Nếu trong lần phân quyền thứ 2 mà không có yêu cầu truy nhập ghi, mà truy nhập đọc lại có yêu cầu thì trong trường hợp này bộ phân xử sẽ phân quyền có truy nhập đọc.
- Nếu trong lần phân quyền thứ 2 mà không có yêu cầu truy nhập nào, thì trong lần phân quyền thứ 3 thì truy nhập ghi sẽ có phân quyền cao hơn.

2.18.2 Phân tích sơ đồ nguyên lý chi tiết của bộ Arbiter

- **sfifoArNotEmpty**: Tín hiệu này tích cực thì thông báo trạng thái FIFO **X2P_SFIFO_AR** có dữ liệu, tức là FIFO **X2P_SFIFO_AR** không rỗng. Khi FIFO **X2P_SFIFO_AR** không rỗng thì FIFO **X2P_SFIFO_AR** sẽ đưa ra yêu cầu phân xử truy nhập đọc.
- **sfifoAwNotEmpty**: Tín hiệu này tích cực thì thông báo trạng thái FIFO **X2P_SFIFO_AW** có dữ liệu, tức là FIFO **X2P_SFIFO_AW** không rỗng. Khi FIFO **X2P_SFIFO_AW** không rỗng thì FIFO **X2P_SFIFO_AW** sẽ đưa ra yêu cầu phân xử truy nhập ghi.

X2P IP core



Hình 2.18: Sơ đồ nguyên lý của Arbiter

- **nextSel[1]:** Tín hiệu điều khiển truy nhập ghi có được phân quyền hay không.
- **nextSel[0]:** Tín hiệu điều khiển truy nhập đọc có được phân quyền hay không.

X2P IP core

NOTE

- support unalign: Transfer 1 là unalign thì transfer 2 là align.