

# HDD(Hardware Description Document)

## AHB to APB conversion IP 및 APB interface 갖는 Slave 구현

차세대반도체학과  
20202840 한정호  
(tbdk5771@naver.com)

## Revision History

Date	Version	Description	Author
2025/07/07	V0.0	HDD template	한정호
2025/07/09	V0.1	Intial HDD	한정호
2025/07/14	V1.0	Finail HDD	한정호

## Table of Contents

1.1. Overview.....	5
1.1.1. Description.....	5
1.1.2. Feature .....	9
1.2. Overall block .....	9
1.2.1. Block diagram .....	9
1.2.2. IO description .....	10
1.2.3. Memory list .....	오류! 책갈피가 정의되어 있지 않습니다.
1.2.4. Register map.....	11
1.3. Sub block description .....	12
1.3.1. Sub block #1 .....	오류! 책갈피가 정의되어 있지 않습니다. <a href="#">3</a>
1.3.2. Sub block #2 .....	<a href="#">14</a>
1.4. Further work.....	16
1.5. Appendix .....	16

## 1.1. Overview

대부분의 SoC(System-on-chip)는 내부 데이터 경로를 관리하기 위해 Arm의 AMBA(Advanced Microcontroller Bus Architecture protocol)를 채택하여 사용하고 있다. AMBA는 대표적으로 고성능의 AXI(Advanced eXtensible Interface)와 AHB(Advanced High-performance Bus), 저전력/저속 주변장치를 위한 APB(Advanced Peripheral Bus)로 구성되어 있다. SoC내에서 CPU Core, DMA Controller와 같은 고성능 Master에 연결되는 Main bus는 AXI 혹은 AHB로 구성되어 빠른 데이터 처리를 담당한다. 반면에 동작 속도가 상대적으로 느린 주변장치들은 APB를 활용하여 UART, SPI, USB 등의 peripheral들을 제어한다. 서로 다른 프로토콜을 연결하기 위해선 신호 체계를 중계하고 변환하는 Bridge를 구현해야 한다. SoC 기초 설계 역량을 쌓고자 이번 과제에선 AHB와 APB를 연결하는 AHB2APB bridge을 구현하였다. 본 문서는 '1) AHB to APB protocol conversion HW block 구현, '2) APB와 연결된 memory mapped register block'의 구현 및 동작을 설명하고 있다. 3) 최종적으로 AHB transaction 동작을 의도한 Testbench를 생성하여 AHB to APB protocol conversion 과정과 APB와 연결된 memory mapped register block'의 동작을 검증하는 결과를 설명하고 있다.

### 1.1.1.Description

APB(Advanced Peripheral Bus) Protocol 은 저전력/저속 주변장치용 버스이다.

AMBA 에서 가장 간단한 Protocol 을 활용하여 시스템의 효율성과 비용 최적화를 극대화 할 수 있다.

1 개의 Master 가 여러 개의 Slave 에 연결되어 있으며 Data bus 가 Write Data 와 Read Data 로 분리.

[1-a] Signal of APB. (\*Signal 의 명확한 구분을 위해 APB 는 P, AHB 는 H 로 명명하였다.)

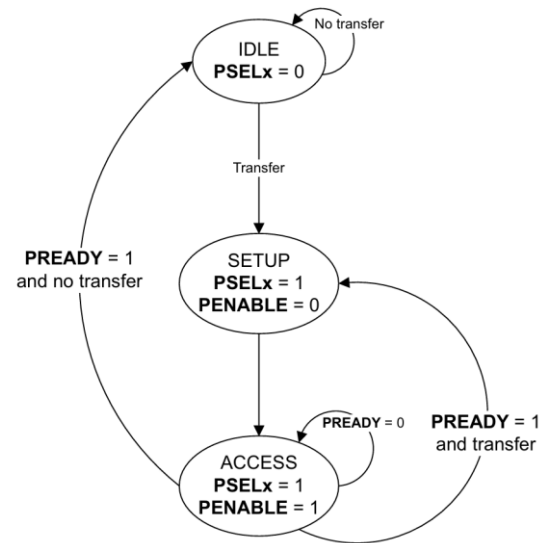
Signal	Source	Description
<b>PCLK</b>	Clock source	Clock. The rising edge of <b>PCLK</b> times all transfers on the APB.
<b>PRESETn</b>	System bus equivalent	Reset. The APB reset signal is active LOW. This signal is normally connected directly to the system bus reset signal.
<b>PADDR</b>	APB bridge	Address. This is the APB address bus. It can be up to 32 bits wide and is driven by the peripheral bus bridge unit.
<b>PSELx</b>	APB bridge	Select. The APB bridge unit generates this signal to each peripheral bus slave. It indicates that the slave device is selected and that a data transfer is required. There is a <b>PSELx</b> signal for each slave.
<b>PENABLE</b>	APB bridge	Enable. This signal indicates the second and subsequent cycles of an APB transfer.
<b>PWRITE</b>	APB bridge	Direction. This signal indicates an APB write access when HIGH and an APB read access when LOW.
<b>PWDATA</b>	APB bridge	Write data. This bus is driven by the peripheral bus bridge unit during write cycles when <b>PWRITE</b> is HIGH. This bus can be up to 32 bits wide.
<b>PREADY</b>	Slave interface	Ready. The slave uses this signal to extend an APB transfer.
<b>PRDATA</b>	Slave interface	Read Data. The selected slave drives this bus during read cycles when <b>PWRITE</b> is LOW. This bus can be up to 32-bits wide.
<del><b>PSLVERR</b></del>	Slave interface	This signal indicates a transfer failure. APB peripherals are not required to support the <b>PSLVERR</b> pin. This is true for both existing and new APB peripheral designs. Where a peripheral does not include this pin then the appropriate input to the APB bridge is tied LOW.

\*PREADY(Slave -> bridge) : slave 의 준비 여부를 의미하는 신호이며, 전송 연장 기능을 구현한다.

\*이 외에 다른 신호는 직관적으로 이해가 가능하므로 추가적인 설명은 하지 않는다.

### [1-b] FSM of APB Protocol

1. IDLE : APB 의 기본 상태이다.
2. SETUP : Transfer 시작 될 시 진입하는 첫 단계,  
PSELx = 1, PENABLE=0 이며, PADDR, PWRITE, PWDATA 유효  
SETUP State 는 1clock 만 유지한다.
3. ACCESS : Slave 에게 Write or Read 동작을 지시하는 단계  
PSELx = 1, PENABLE=1 이며 SETUP state 다음 Clcok 에 진입.  
PREADY = 0 일 시, ACCESS State 에 머무른다 = [Wait State]  
PREADY = 1 일 시, 추가적인 Transfer 가 없을 시 IDLE State,  
있을 시 SETUP State 로 향한다.

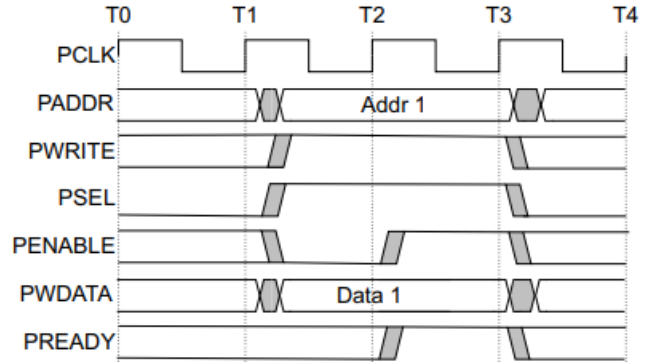


### Goal of APB Module

- \*APB Slave – PREADY as fast as possible
- \*APB MASTER - Minimize the gap between PSELs

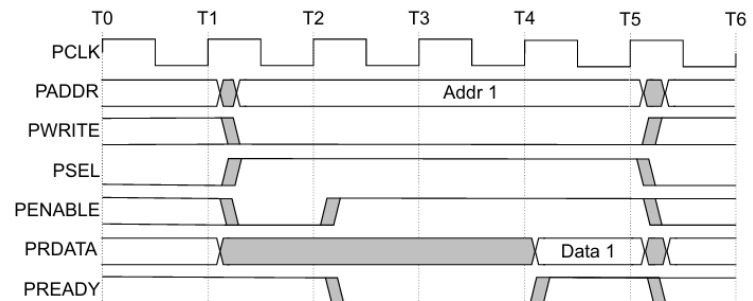
### [1-c] Write Trasnfer of APB Protocol [with no wait]

- T0 : IDLE State  
 T1 : SETUP State  
 T2 : ACCESS State (PREADY = Rising edge)  
 T3 : IDLE State



### [1-d] Write Trasnfer of APB Protocol [with wait]

- T0 : IDLE State  
 T1 : SETUP State  
 T2 : ACCESS State (PREADY = Falling edge)  
 T3 : ACCESS State (PREADY=0)  
 T4 : ACCESS State (PREADY = 1)  
 T5 : IDLE State

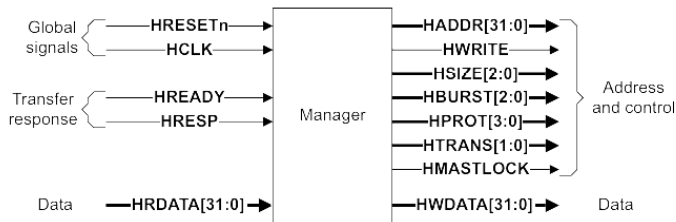


AHB(Advanced High Performance Bus) Protocol 은 고성능 버스이다.

고속 동작, Pipeline, Burst transfer 등 다양한 기능을 통해 병목 현상을 해결하며 고성능 장치의 성능을 최대한 보장함을 통해 시스템의 데이터 처리 효율을 극대화 한다.

여러 개의 MASTER 와 여러 개의 SLAVE 로 구성되어 있으며 Arbiter, Decoder, Mux 를 활용하여 여러 개의 MODULE 을 제어한다. Address phase 와 Data phase 가 중첩되어 있다.

[2-a] Signal and Diagram of AHB master & Arbiter.



\*HBUSREQ (M -> Arbiter)

- Master 의 Bus 사용 요청

\*HLOCKS (M -> Arbiter)

- Master 의 Bus 독점 사용 요청

\*HGRANTs (Arbiter -> M)

- Master 의 Bus 사용 허가

\*HTRANS = AHB State

- IDLE, BUSY, NONSEQ, SEQ

\*HSIZE =  $2^{\text{HSIZE}}$  byte = beat 크기

- byte, half-word, word

- beat = BURST 내의 데이터 전송 최소 단위

\*HBURST \*BURST : 데이터 연속 전송 묶음

- Single / Incr N / wrap N / Incr

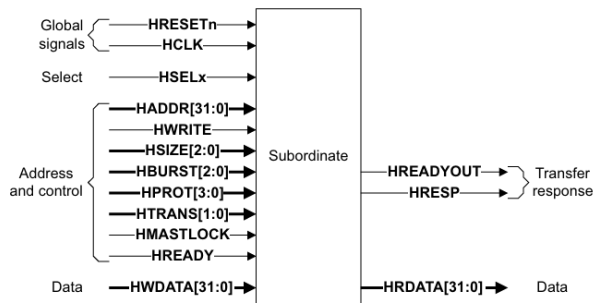
\*HPROT = H-Protection : 권한 정보 제공

Opcode, Privileged, Bufferable, Modifiable

Name	Source	Description
<b>HBUSREQx</b> Bus request	Master	A signal from bus master x to the bus arbiter which indicates that the bus master requires the bus. There is an <b>HBUSREQx</b> signal for each bus master in the system, up to a maximum of 16 bus masters.
<b>HLOCKx</b> Locked transfers	Master	When HIGH this signal indicates that the master requires locked access to the bus and no other master should be granted the bus until this signal is LOW.
<b>HGRANTx</b> Bus grant	Arbiter	This signal indicates that bus master x is currently the highest priority master. Ownership of the address/control signals changes at the end of a transfer when <b>HREADY</b> is HIGH, so a master gets access to the bus when both <b>HREADY</b> and <b>HGRANTx</b> are HIGH.

Name	Source	Description
<b>HCLK</b> Bus clock	Clock source	This clock times all bus transfers. All signal timings are related to the rising edge of HCLK.
<b>HRESETn</b> Reset	Reset controller	The bus reset signal is active LOW and is used to reset the system and the bus. This is the only active LOW signal.
<b>HADDR[31:0]</b> Address bus	Master	The 32-bit system address bus.
<b>HTRANS[1:0]</b> Transfer type	Master	Indicates the type of the current transfer, which can be NONSEQUENTIAL, SEQUENTIAL, IDLE or BUSY.
<b>HWRITE</b> Transfer direction	Master	When HIGH this signal indicates a write transfer and when LOW a read transfer.
<b>HSIZE[2:0]</b> Transfer size	Master	Indicates the size of the transfer, which is typically byte (8-bit), halfword (16-bit) or word (32-bit). The protocol allows for larger transfer sizes up to a maximum of 1024 bits.
<b>HBURST[2:0]</b> Burst type	Master	Indicates if the transfer forms part of a burst. Four, eight and sixteen beat bursts are supported and the burst may be either incrementing or wrapping.
<b>HPROT[3:0]</b> Protection control	Master	The protection control signals provide additional information about a bus access and are primarily intended for use by any module that wishes to implement some level of protection.  The signals indicate if the transfer is an opcode fetch or data access, as well as if the transfer is a privileged mode access or user mode access. For bus masters with a memory management unit these signals also indicate whether the current access is cacheable or bufferable.
<b>HWDATA[31:0]</b> Write data bus	Master	The write data bus is used to transfer data from the master to the bus slaves during write operations. A minimum data bus width of 32 bits is recommended. However, this may easily be extended to allow for higher bandwidth operation.

## [2-b] Signal and Diagram of AHB Slave



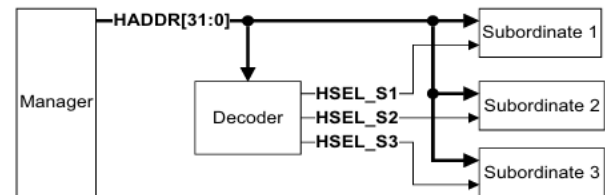
Name	Source	Description
<b>HSELx</b> Slave select	Decoder	Each AHB slave has its own slave select signal and this signal indicates that the current transfer is intended for the selected slave. This signal is simply a combinatorial decode of the address bus.
<b>HRDATA[31:0]</b> Read data bus	Slave	The read data bus is used to transfer data from bus slaves to the bus master during read operations. A minimum data bus width of 32 bits is recommended. However, this may easily be extended to allow for higher bandwidth operation.
<b>HREADY</b> Transfer done	Slave	When HIGH the <b>HREADY</b> signal indicates that a transfer has finished on the bus. This signal may be driven LOW to extend a transfer. Note: Slaves on the bus require <b>HREADY</b> as both an input and an output signal.
<b>HRESP[1:0]</b> Transfer response	Slave	The transfer response provides additional information on the status of a transfer. Four different responses are provided, OKAY, ERROR, RETRY and SPLIT.

\*HREADY, HRESP = 전송 결과 : ERROR, OK 이 외의 신호선은 MASTER 와 동일

## [2-c] Signal and Diagram of AHB Decoder

- Decoder 활용하여 HSEL 제어

HSEL, HADDR 를 조합하여 원하는 SLAVE 와 통신



## [2-d] Signal and Diagram of AHB Mux

- Decoder 에서 Mux 로 제어신호 전달

Mux 와 제어신호 활용하여 SLAVE 선택 후

HRDATA, HRESP, HREADY 결정

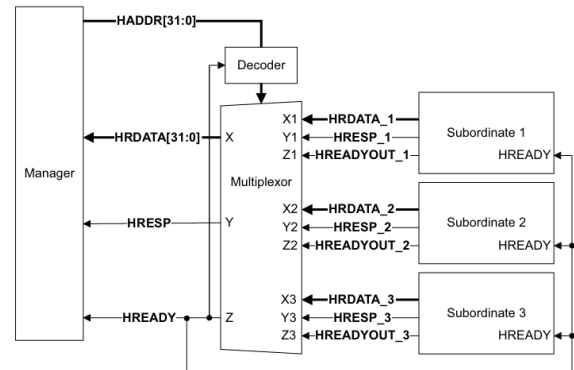


Figure 4-2 Multiplexor interconnection

## [2-e] FSM of AHB Protocol

\*HTRANS Signal 의 상태 값이 실질적인 FSM 의 State 를 담당 하고 있다.

00 = IDLE : 대기 상태, 01 = BUSY : NEXT Transfer 전송 없음,

10 = NONSEQ : 단일 전송, 첫 번째 전송, 11 = SEQ : Burst 진행 중

## Goal of AHB Module

\*AHB Slave – PREADY as fast as possible & Support Burst

\*AHB MASTER – Keep the Pipeline & Maximize Burst Usage

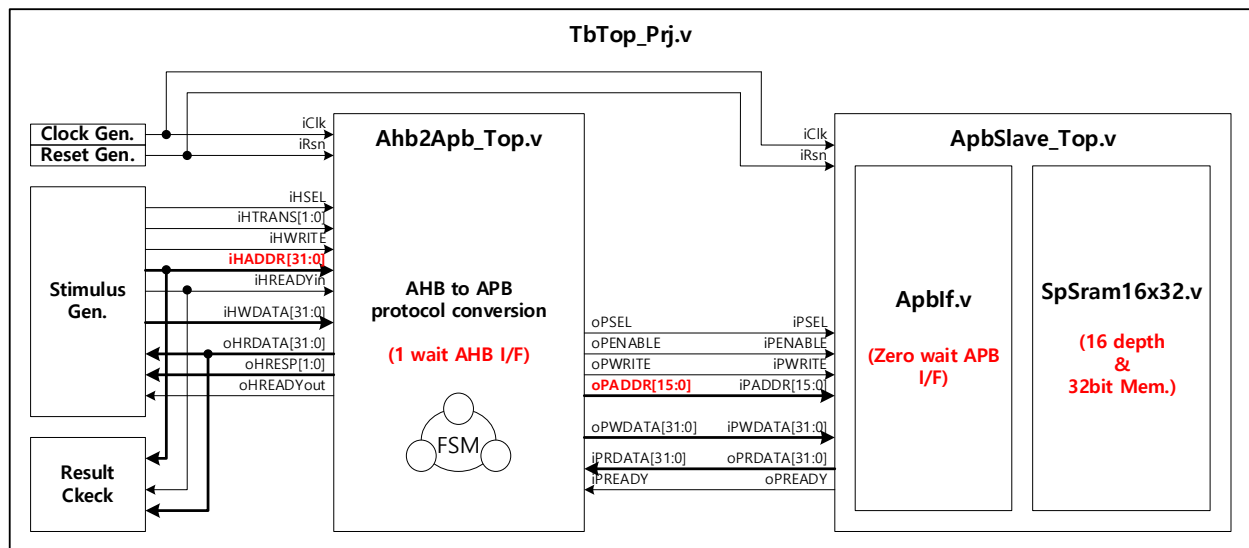
### 1.1.2.Feature

이번 인턴과정에서 구현한 Ahb2Apb는 AHB MASTER의 요청을 받아 APB SLAVE로 전달하며 상이한 프로토콜 간의 동작 방식 차이를 안정적으로 변환하는 브릿지이다. FSM기반으로 코드를 작성했으며 1 wait를 구현하고자 Data Latching Logic을, 0 wait를 구현하고자 Output Logic을 작성하였다.

ApbSlave는 APB 프로토콜 기반으로 16x32 bit 크기의 SRAM에 데이터를 저장하고 읽어오는 메모리 맵 주변 장치이다. SpSram16x32 Module에서 SRAM function을 구현하였으며, Apblf Moudle에서는 APB protocol 기반으로 Sram과 통신할 수 있는 동작을 기술하였다.

## 1.2. Overall block

### 1.2.1.Block diagram



\*최상위 모듈을 Project\_Top 이라 명명했으며, 주어진 Blcok Diagram 에 따라 최상위 모듈의 Testbench 를 TbTop\_Prj 라 명명하였다.

Project\_Top 은 Ahb2Apb\_Top 와 ApbSlave\_Top 를 인스턴스화 하며 두 모듈간의 APB interface signal 을 내부 Wire 로 연결하여 하나의 완성된 시스템을 구성하고 있는 최상위 모듈이다.

TbTop\_Prj 는 Project\_Top 가 정확히 동작하는지 검증하기 위해 설계된 Testbench 이다. Module 내부에는 AHB master 의 동작을 구현한 AHB\_write 와 AHB\_read Task 를 작성하였으며, Address Phase 와 Data Phase 를 순차적으로 생성함을 통해 제어신호가 AHB Protocol 에 맞게 동작하도록 한다. Expected\_data 인자를 삽입하여 이 값을 oHRDATA 와 비교한 후 Error 여부를 판별할 수 있는 Result-check 기능을 구현하였다.



## 1.2.2.IO description

Table 1 Input

Name	Bit width	Initial value	Source Block	Description
iClk	1	0	TbTop_Prj	100MHz clock input
iRsn	1	1	TbTop_Prj	Active low reset
iHSEL	1	0	TbTop_Prj	Select signal for the AHB slave
iHTRANS	2	0	TbTop_Prj	Transfer type
iHWRITE	1	0	TbTop_Prj	1: Write / 0: Read
iHADDR	32	0	TbTop_Prj	32bit address bus
iHREADYin	1	x	TbTop_Prj	Ready signal from the AHB
iHWDATA	32	0	TbTop_Prj	Write data from the AHB Master
iPSEL	1	0	Ahb2Apb_Top	Select signal for the APB slave
iPENABLE	1	0	Ahb2Apb_Top	Strobe for the access phase
iPWRITE	1	0	Ahb2Apb_Top	1: Write / 0: Read
iPADDR	16	0	Ahb2Apb_Top	16bit address bus
iPWDATA	32	0	Ahb2Apb_Top	APB write data

Table 2 Output

Name	Bit width	Initial value	Destination Block	Description
oHRDATA	32	0	TbTop_Prj	Read data from the AHB slave
oHRESP	1	xx	TbTop_Prj	Transfer response (Okay, Error)
oHREADYout	1	x	TbTop_Prj	Bridge Ready Signal
oPWDATA	32	0	Ahb2Apb_Top	Write data to APB
oPRDATA	32	0	ApbSlave_Top	Read data from the APB
oPREADY	1	0	ApbSlave_Top	APB Slave Ready Signal

### 1.2.3.Memory list

Table 3 Memory list

Name	Location	Type	Size	Description
rMEM	SPram16x32.rMEM	SP SRAM	16*32	16-depth & 32bit memory mapped register

### 1.2.4.Register map

	NAME	Offset	Range	Field	Reset	Access	Description
B	Project	0x7000_0000	0xFFFF				
S	InBuf						
R	MemBufAddr_0	0x8000	[31:0]		0x0	R/W	
F			[31:0]	MemBufAddr_0	0x0		MemBuf address 0 data write
R	MemBufAddr_1	0x8004	[31:0]		0x0	R/W	
F			[31:0]	MemBufAddr_1	0x0		MemBuf address 1 data write
R	MemBufAddr_2	0x8008	[31:0]		0x0	R/W	...
F			[31:0]	MemBufAddr_2	0x0		MemBuf address 2 data write
R	MemBufAddr_3	0x800C	[31:0]		0x0	R/W	
F			[31:0]	MemBufAddr_3	0x0		MemBuf address 3 data write
R	MemBufAddr_4	0x8010	[31:0]		0x0	R/W	
F			[31:0]	MemBufAddr_0	0x0		MemBuf address 4 data write
R	MemBufAddr_5	0x8014	[31:0]		0x0	R/W	
F			[31:0]	MemBufAddr_1	0x0		MemBuf address 5 data write
R	MemBufAddr_6	0x8018	[31:0]		0x0	R/W	...
F			[31:0]	MemBufAddr_2	0x0		MemBuf address 6 data write
R	MemBufAddr_7	0x801C	[31:0]		0x0	R/W	
F			[31:0]	MemBufAddr_3	0x0		MemBuf address 7 data write
R	MemBufAddr_8	0x8020	[31:0]		0x0	R/W	
F			[31:0]	MemBufAddr_0	0x0		MemBuf address 8 data write
R	MemBufAddr_9	0x8024	[31:0]		0x0	R/W	
F			[31:0]	MemBufAddr_1	0x0		MemBuf address 9 data write
R	MemBufAddr_10	0x8028	[31:0]		0x0	R/W	...
F			[31:0]	MemBufAddr_2	0x0		MemBuf address 10 data write
R	MemBufAddr_11	0x802C	[31:0]		0x0	R/W	
F			[31:0]	MemBufAddr_3	0x0		MemBuf address 11 data write
R	MemBufAddr_12	0x8030	[31:0]		0x0	R/W	
F			[31:0]	MemBufAddr_0	0x0		MemBuf address 12 data write
R	MemBufAddr_13	0x8034	[31:0]		0x0	R/W	
F			[31:0]	MemBufAddr_1	0x0		MemBuf address 13 data write
R	MemBufAddr_14	0x8038	[31:0]		0x0	R/W	...
F			[31:0]	MemBufAddr_2	0x0		MemBuf address 14 data write
R	MemBufAddr_15	0x803C	[31:0]		0x0	R/W	
F			[31:0]	MemBufAddr_3	0x0		MemBuf address 15 data write

\* Base Address : 0x7000\_0000 [실제 유효 주소 범위 결정]

\* 실제 접근 주소 : 0x7000\_8000, 메모리 워드 4 바이트 크기,

### 1.3. Sub block description

설계된 HW는 계층적 구조로 구성되어 있으며 최상위 Project\_Top Module 하에 AHB to APB protocol conversion을 담당하는 Ahb2Apb\_Top과 SRAM과 APB Interface가 포함된 주변장치 역할을 하는 ApbSlave\_Top라는 두개의 주요 Sub block으로 나뉜다.

ApbSlave\_Top Module 은 APB interface 와 SpSram 16x32 으로 세분화되어 구성되어 있다.

#### 1.3.1.Ahb2Apb\_Top

Ahb2Apb\_Top 모듈은 고성능 AHB 와 저전력 APB 를 연결하는 Bridge이다. AHB 에서는 Slave로 동작하여 Master의 요청을 받고, APB 에서는 마스터Master로 동작하여 APB Slave와 통신을 수행한다. Module은 두 BUS 간의 속도 및 동작 방식의 차이를 안정적으로 변환하고 중계하는 것을 목표로 한다.

Ahb2Apb\_Top Block Diagram

*Ahb2Apb\_Top Module 은 AHB interface, APB interface, FSM, main Logic 으로 구성되어 있다.*

Interface

- AHB interface[Input] : AHB Master로부터 iHESEL, iHTRANS, iHADDR, iHREADYin, iHWDATA 입력받는다.
- AHB interface[Output] : AHB Master에게 oHREADYout, oHRESP, oHRDATA 출력한다.
- APB interface[Input] : APB Slave로부터 iPRDATA, iPREADY 입력받는다.
- APB interface[Output] : APB Slave에게 oPSEL, oPENABLE, oPWRITE, oPADDR, oPWDATA 출력한다.

FSM

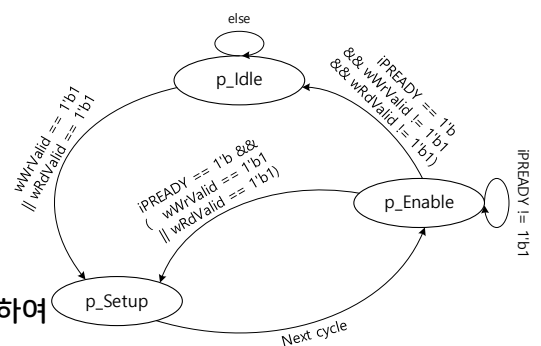
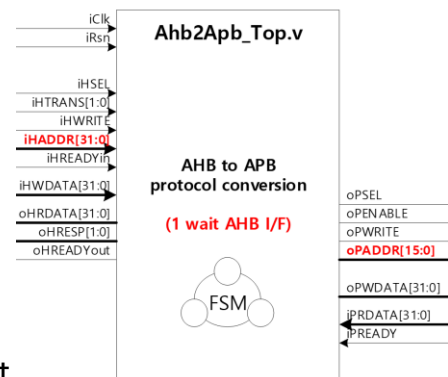
3 State 기반으로 동작한다.

1. p\_Idle : 기본 대기 상태이며, iHSEL, iTRANS 을 모니터링하여 AHB req 가 (wWrvalid or wRdValid)가 감지되면 iHADDR 과 iHWRITE 내부 Register 에 Latch 후 p\_Setup 으로 전환한다.

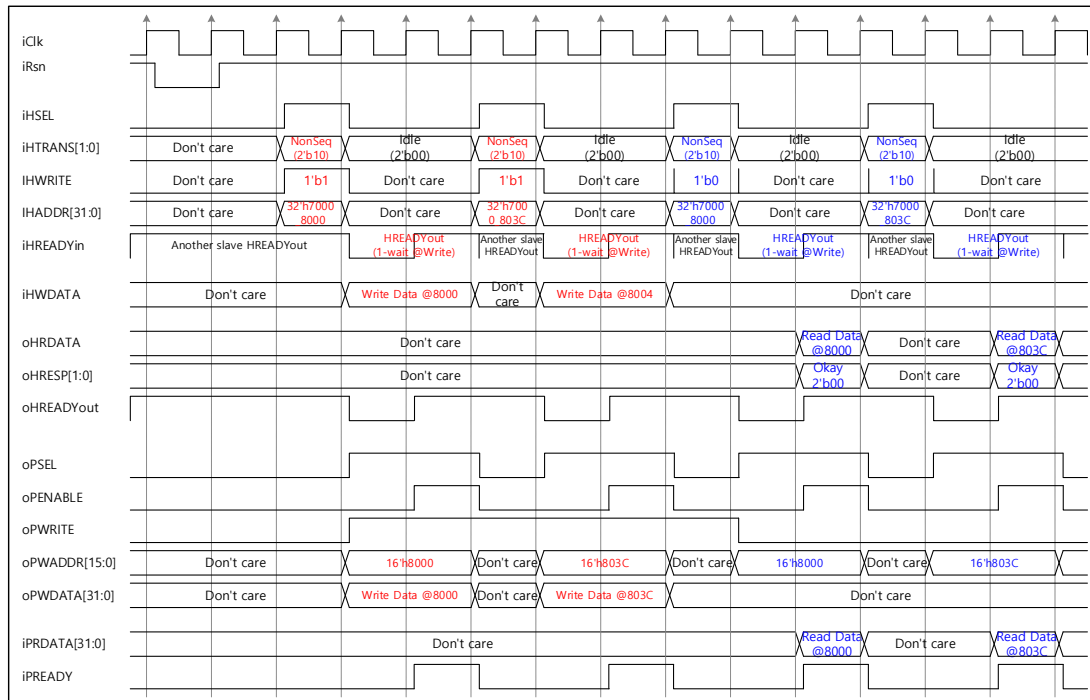
2. p\_setup : APB transaction 위한 첫 번째 단계, oPSEL 활성화하여

Target Slave 선택한다. Bridge 는 wait state 위해 oHREADYout 을 0 으로 만든다.

3. p\_Enable : APB transaction 위한 두 번째 단계, oPEANBLE 활성화하여 Slave 에게 DATA 전송 전송완료 확인 전까지 p\_ENABLE 에 머무르며, iPREADY=1 확인하여 전송확인이 완료된다면 다음 AHB req 가 있을 시 p\_Setup 으로 전환되며 없을 시 p\_Idle 로 복귀한다.



## Timing Diagram



2 번의 Write 동작과 2 번의 Read 동작이 순차적으로 일어나는 과정을 보여주는 Timing Diagram 이다.

왼쪽에서 오른쪽으로 시간이 흐르며 상태변화는 Rising Edge 일 때 발생한다.

1. 첫 번째 Write 동작(ADDR : 0x8000) : iHTRANS = 2'b10 설정하여 전송이 시작됨을 알린 후, HWRITE=High 설정하여 Write 동작임을 명시한다. HADDR 에 Target addr 인 32'h7000\_8000 을 입력한 후, HREADYout 을 Low 설정하여 1Wait State 를 삽입한다. 1Clock 이후 PENABLE 를 활성화 하여 iHWDATA 를 APB Slave 에 전달한다. 전달 완료 후 APB 가 PREADY 로 응답하면 동작이 완료된다.
2. 두 번째 Write 동작(ADDR : 0x803C) : 첫 번째 Write 동작과 동일, Target Addr 를 변경하여 진행.
3. 첫 번째 Read 동작(ADDR : 0x8000) : Write 동작과 동일하게 HREADYout = Low 설정하여 1Wait State 를 삽입한다. Bridge 가 PSEL =High 설정하여 APB 에 REQ 함. iHTRANS = 2'b10 설정하여 전송이 시작됨을 알린 후, HWRITE=Low 로 설정하여 Read 동작임을 명시한다. APB 는 PRDATA 를 통해 응답하며, Bridge 는 HREADYOUT 이 다시 High 가되는 시점에 iPRADATA 를 oHRDATA 로 출력한다. 동시에 정상적인 전송을 의미하는 HRESP = 2'b00 을 전송한다.
4. 두 번째 Read 동작 (ADDR : 0x803C) : 첫 번째 Read 동작과 동일, Target Addr 를 변경하여 진행

### 구현된 알고리즘

1. 1Wait -state : AHB REQ 이후 p\_SETUP에서 의도적으로 oHREADYout= Low로 만들어 Wait State를 발생시킵니다. SETUP 단계에서 작업을 안정적으로 수행할 시간을 확보해주어, 프로토콜 간의 타이밍 제약을 단순화하고 설계 안정성을 높인다.
2. 전송 최적화 : ENABLE 상태에서 전송 완료 후, 다음 AHB Req 가 대기중이라면 바로 SETUP 상태로 전환한다. 해당 기능을 통하여 BUS 의 데이터 처리를 최적화 하였다.

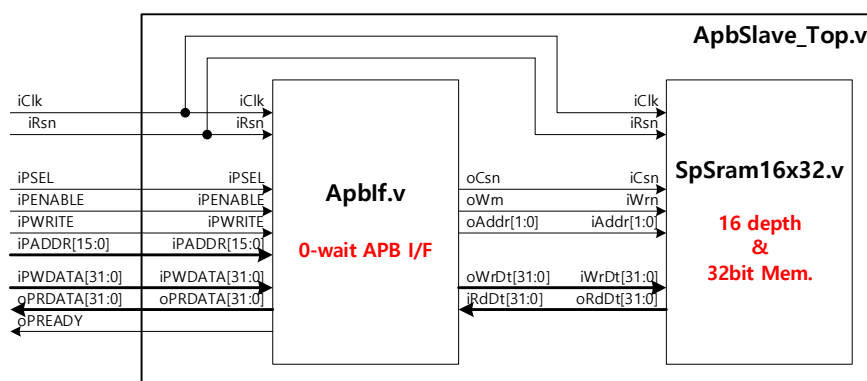
### 1.3.2. ApbSlave\_Top

ApbSlave\_Top Module은 Bridge로부터 오는 요청 받아 16x32 bit SRAM에 DATA를 쓰거나 읽는 역할을 한다. 설계된 모듈은 APB Protocol 기반의 Apblf와 메모리 소자인 SpSRAM16x32로 구성되어 있다.

ApbSlave\_Top Blocok Diagram

ApbSlave\_Top Module 은 Bridge interface,

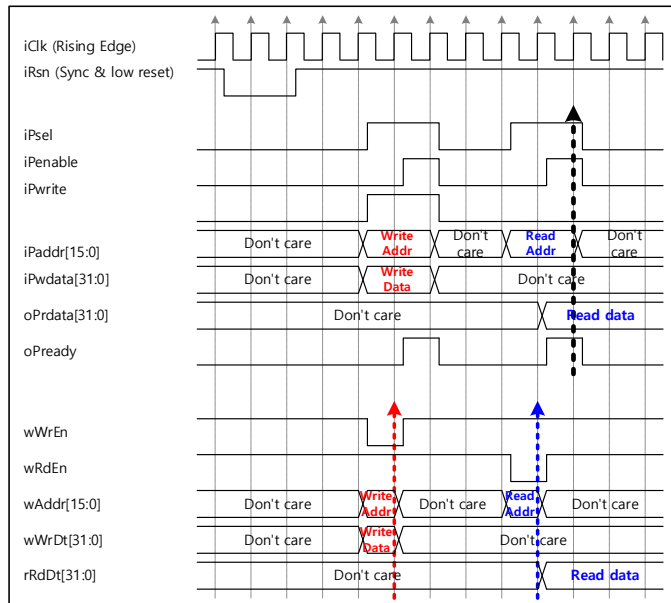
APblf, SpSRAM16x32 으로 구성되어 있다.



#### Interface

- Bridge interface[Input] : Bridge로부터 iPESEL, iPENABLE, iPWRITE, iPADDR, iPWDATA, iHWDATA 입력받는다.
- Bridge interface[Output] : Bridge에게 oPRDATA, oPREADY 출력한다.
- Wire & reg : Apblf가 SpSram을 제어할 수 있는 Signal인 Csn, Wrn, Addr, RdDt를 구현하였다.  
\*Csn [ 0=Sel], Wrn[0=Write]

## Timing diagram



1 번의 Write, Read 동작이 순차적으로 일어남을 보여주는 Timing diagram 이다.

Addr 와 Data 전달이 2Clock Cycle 로 이루어져있다.

### 1. Write 동작

(iPSEL=1 & iPEanble=0 & iPwrite=1) 만족하여 wWrEn =High 설정 후 Csn=1, Wrn=0 설정

동시에 iPaddr 및 iPwdata 전달. SRAM 은 Module 은 이전 Cycle 에 준비된 Addr 와 Data 로 Write 수행, Zero wait 이기 때문에 Write 를 완료함과 동시에 oPready=High 설정하여, 추가적인 대기시간 발생을 없앤다.

### 2. Read 동작

Bridge 가 wRdEn = High 설정 후 ipaadr 전송, 이후 Sram Read 활성화되어 iPaddr 읽은 후 다음 Clock 에 iRdDt 출력, 출력을 완료함과 동시에 oPready=High 설정하여, 추가적인 대기시간 발생을 없앤다.

## 구현된 알고리즘

1. Zero Wait State : APB Protocol 을 성능을 극대화 하기 위하여 Enable 활성화하여 Write, Read 동작이 진행됨과 동시에 oPREADY 로 응답하는 알고리즘을 구현하였다. 이를 통해 시스템 전체의 효율성을 증대시켰다.

## 1.4. Further work

### Ahb2Apb 개선 방향

1. Burst Logic 구현 : 현재 구현된 Ahb2Apb Module 은 AHB Burst 를 지원하지 않아 개별적인 단일 APB 전송으로 데이터를 처리하고 있다. 이는 대량의 데이터를 연속적으로 전송할 시 상당히 비효율적이다. 이를 개선하기 위해 향후 AHB Burst Logic 을 구현하고자 한다. 구현계획은 다음과 같다.

HBURST(ex. INCR, wrap) 요청이 들어오면, Module 내부에 카운터를 두어 요청을 기억한다. 이후 첫 NONSEQ 전송 이후, 카운터가 0 이 될 때까지 연속적인 SETUP-ACCESS Cycles 을 생성하여 데이터를 연속적이며 효율적으로 처리할 수 있도록 한다.

2. HPROT Signal 구현 : 현재 구현된 Ahb2Apb Module 은 HPROT Signal 를 지원하지 않아 메모리 보호 속성을 구현하지 않은 채 모든 접근을 동일하게 처리하고 있다. 시스템의 보안은 필수적이므로 다음과 같은 계획을 통해 구현하고자 한다.

Module 내부에 새로운 레지스터를 선언하고 관리자 영역의 주소를 설정한다. 관리자 영역의 주소와 HPROT signal 을 조합하여 시스템 보안 로직을 생성한다.

### ApbSlave 개선 방향

1. PSLVERR Signal 활용하여 Error 처리 기능 구현 : 현재 ApbSlave 는 잘못된 요청에 대해 아무런 응답을 하지 않도록 구현되어 있어 상위 시스템이 에러 상황을 인지할 수가 없다. 이는 시스템의 디버깅을 어렵게함과 동시에 오동작의 주 원인이 된다. 다음과 같은 계획을 통해 에러 로직을 구현하고자 한다.

주소 디코딩 로직과 입,출력 순서에 관한 로직을 새롭게 설정한다. 이후 해당 로직을 PSLVERR 와 연결하여 에러 발생시 상위 시스템에 에러를 전달할 수 있도록 PSLVERR 을 설정한다.

## 1.5. Appendix

- [1] Arm Limited. *AMBA APB Protocol Specification*. Version E.
- [2] Arm Limited. *AMBA AHB Protocol Specification*. Version C.
- [3] ARM bus protocol 이해 및 bus interface 를 갖는 HW IP 설계 실습\_03 일차
- [4] ARM bus protocol 이해 및 bus interface 를 갖는 HW IP 설계 실습\_04 일차
- [5] ARM bus protocol 이해 및 bus interface 를 갖는 HW IP 설계 실습\_07~08 일차
- [6] ARM bus protocol 이해 및 bus interface 를 갖는 HW IP 설계 실습\_9 일차
- [7] ARM bus protocol 이해 및 bus interface 를 갖는 HW IP 설계 실습\_10 일차