AMBA APB Checker VIP

Datasheet (v1.0)

https://roalogic.github.io/amba_pkg

17-Apr-2024

Contents

1 Preface **1**

1.1 Roa Logic Open Source Commitment . . . 1

1.2 Copyright . . . 1

1.3 License . . . 1

1.4 Disclaimer . . . 1

1.5 Third-Party . . . 1

2 Introduction **2**

2.1 Features . . . 2

2.2 Benefits . . . 2

3 Specifications **3**

3.1 Functional Description . . . 3

3.2 Operating Modes . . . 3

3.2.1 PCLK . . . 3

4 Configurations **4**

4.1 Introduction . . . 4

4.2 Core Configuration . . . 4

4.3 Core Parameters . . . 4

4.3.1 ADDR_WIDTH . . . 4

4.3.2 DATA_WIDTH . . . 4

4.3.3 USER_REQ_WIDTH . . . 5

4.3.4 USER_DATA_WIDTH . . . 5

4.3.5 USER_RESP_WIDTH . . . 5

4.3.6 CHECK_PSTRB . . . 5

4.3.7 CHECK_PPROT . . . 5

4.3.8 CHECK_PSLVERR . . . 5

4.3.9 WATCHDOG_TIMEOUT . . . 5

4.4 Functions for Dynamic Configuration . . . 5

4.4.1 get_severity . . . 6

4.4.2 set_severity . . . 6

5 Interfaces **7**

5.1 APB Interface . . . 7

5.1.1 PRESETn . . . 7
5.1.2 PCLK . . . 7
5.1.3 PSEL . . . 7
5.1.4 PENABLE . . . 8
5.1.5 PADDR . . . 8
5.1.6 PWRITE . . . 8
5.1.7 PSTRB . . . 8
5.1.8 PPROT . . . 8
5.1.9 PWDATA . . . 8
5.1.10 PRDATA . . . 8
5.1.11 PREADY . . . 8
5.1.12 PSLVERR . . . 8
5.1.13 PWAKEUP . . . 8
5.1.14 PAUSER . . . 9
5.1.15 PWUSER . . . 9
5.1.16 PRUSER . . . 9
5.1.17 PBUSER . . . 9

6 Rules **10**
6.1 Introduction . . . 10
6.2 Rules . . . 11
6.2.1 PSEL Must remain high for the entire transfer . . . 11
6.2.2 PSEL Undefined . . . 11
6.2.3 PENABLE must be low during Setup Phase . . . 11
6.2.4 PENABLE must be high during Access Phase . . . 12
6.2.5 PENABLE undefined . . . 12
6.2.6 PADDR must remain stable for the entire transfer . . . 12
6.2.7 PADDR versus PSTRB misaligned . . . 13
6.2.8 PADDR should be aligned to DATA_WIDTH . . . 13
6.2.9 PADDR undefined . . . 13
6.2.10 PWRITE must remain stable for the entire transfer . . . 14
6.2.11 PWRITE undefined . . . 14
6.2.12 PSTRB value non byte/word/dword/... . . . 14
6.2.13 PSTRB must remain stable for the entire transfer . . . 15
6.2.14 PSTRB undefined . . . 15
6.2.15 PPROT must remain stable for the entire transfer . . . 15

6.2.16 PPROT undefined . . . 16
6.2.17 PWDATA must remain stable for the entire transfer . . . 16
6.2.18 PWDATA contains 'x' . . . 16
6.2.19 PWDATA contains 'x' . . . 17
6.2.20 PRDATA contains 'x' . . . 17
6.2.21 PREADY undefined during Access phase . . . 18
6.2.22 PSLVERR undefined . . . 18
6.2.23 Watchdog expired . . . 19
6.2.24 PWAKEUP must remain high until the end of the transfer . . . 19
6.2.25 PWAKEUP should be asserted at least one cycle before PSEL . . . 20
6.2.26 PWAKEUP raised without starting a transfer . . . 20
6.2.27 PWAKEUP undefined . . . 20
6.2.28 PAUSER must remain stable for the entire transfer . . . 21
6.2.29 PAUSER undefined . . . 21
6.2.30 PAUSER should be max 128 bits . . . 21
6.2.31 PWUSER must remain stable for the entire transfer . . . 22
6.2.32 PWUSER undefined . . . 22
6.2.33 PWUSER should be max DATA_WIDTH/2 bits . . . 22
6.2.34 PRUSER contains 'x' . . . 23
6.2.35 PRUSER should be max DATA_WIDTH/2 bits . . . 23
6.2.36 PBUSER contains 'x' . . . 23
6.2.37 PBUSER should be max 16 bits . . . 23
6.2.38 PSTRB must be low during read transfer . . . 24
6.2.39 PADDR should be max 32 bits . . . 24
6.2.40 PWDATA should be max 8, 16, or 32 bits wide . . . 24
6.2.41 PRDATA should be max 8, 16, or 32 bits wide . . . 24
6.2.42 PRESETn Undefined . . . 24
6.2.43 PCLK Undefined . . . 25
6.3 Rules per signal . . . 26

7 Extending the VIP **27**
7.1 Introduction . . . 27

8 Bibliography **28**

9 Revision History **29**

List of Figures

6.1 APB Transfer Examples 10
6.2 APB-1 Example 11
6.3 APB-2 Example 11
6.4 APB-3 Example 11
6.5 APB-4 Example 12
6.6 APB-5 Example 12
6.7 APB-6 Example 12
6.8 APB-7 Example 13
6.9 APB-8 Example 13
6.10 APB-9 Example 13
6.11 APB-10 Example 14
6.12 APB-11 Example 14
6.13 APB-12 Example 14
6.14 APB-13 Example 15
6.15 APB-14 Example 15
6.16 APB-15 Example 15
6.17 APB-16 Example 16
6.18 APB-17 Example 16
6.19 APB-18 Example 16
6.20 APB-19 Example 17
6.21 APB-20 Example 17
6.22 APB-21 Example 18
6.23 APB-22 Example 18
6.24 APB-23 Example 19
6.25 APB-24 Example 19
6.26 APB-25 Example 20
6.27 APB-26 Example 20
6.28 APB-27 Example 20
6.29 APB-28 Example 21
6.30 APB-29 Example 21
6.31 APB-31 Example 22
6.32 APB-32 Example 22
6.33 APB-34 Example 23

6.34 APB-36 Example 23
6.35 APB-38 Example 24
6.36 APB-42 Example 24
6.37 APB-43 Example 25

List of Tables

4.1 Core Parameters 4

5.1 APB Interface Ports 7

6.1 Rules per signal 26

9.1 Revision History 29

1. Preface

1.1 Roa Logic Open Source Commitment

Roa Logic is commited to open source software and the open source community and we uphold the values of the free and open source software movement. We believe that contributing, and contributing to, open source software is a way to teach, and to learn, to share experience, and to build experience, and to come together as developers, users, and as humans. We hope that this software will be valuable to someone and enables and motivates new contributors and contributions to the open source community.

Links:
www.roalogic.com
www.github.com/roalogic

1.2 Copyright

1.3 License

1.4 Disclaimer

1.5 Third-Party

2. Introduction

The Roa Logic APB Checker Verification IP helps designers use the ARM® AMBA® APB[1] bus in their designs. Throughout this document ARM® AMBA® APB will be simply refered to as APB. The Roa Logic APB Checker VIP continuously snoops the APB bus and reports any protocol issues it detects. In addition to rules checking, the ABP Checker VIP contains an optional watchdog that fires when the APB bus becomes unresponsive. The APB Bus Checker VIP supports the following APB versions:

- AMBA 2 APB Specification (Issue A), commonly known as APB2
- AMBA 3 APB Specification (Issue B), commonly known as APB3
- AMBA APB Specification (Issue C), commonly known as APB4
- AMBA APB Specification (Issue D), commonly known as APB5

The Roa Logic APB Bus Checker VIP is released under the permissive GPLv3 license.

2.1 Features

- Plug 'n Play APB Checker
- Compliant with APB2, APB3, APB4, and APB5 Bus protocols
- Supports configurable APB address, data, user signal, and response widths
- Autonomously checks APB transactions and signals
- User configurable severity per rule
- Easily extensible with custom rules
- Configurable watchdog fires when the APB bus is unresponsive
- Delivered in plain text SystemVerilog format

2.2 Benefits

- Faster debug due to customised reports
- Integrates into existing Verilog and SystemVerilog testbenches
- Works with existing OVM and UVM test environments
- Open Source, therefore code can be reviewed and extended
- Permissive license

3. Specifications

3.1 Functional Description

The Roa Logic APB Checker VIP is a configurable, fully parameterized Verification IP (VIP) that continuously and autonomosly observes and verifies all APB transactions. The APB Checker VIP is fully compliant with the AMBA APB2, APB3, APB4, and APB5 protocols.

3.2 Operating Modes

The APB Checker VIP supports the APB2, APB3, APB4, and APB5 bus protocols. The protocol to verify is selected using a define statement;

```
‘define APB_VERSION_APB5
‘define APB_VERSION_APB4
‘define APB_VERSION_APB3
```

The default APB2 protocol is used when no define is set. When APB_VERSION_APB5 is defined, then APB_VERSION_APB4 is automatically defined. When APB_VERSION_APB4 is defined, then APB_VERSION_APB3 is automatically defined. The module ports and executed rules reflect the selected protocol.

3.2.1 PCLK

APB is a synchronous protocol. All transactions take place on the rising edge of `PCLK`. Most of the rules are triggered on the rising edge of `PCLK`. This has the advantage of simple rule design and fast execution. The protocol checker has a minimal simulation performance effect. The disadvantage is that the checker does not look at values inbetween clock edges. It is assumed that all APB signals, except for `PRESETn` and `PCLK`, are driven by registers or at least behave like being driven by registers.

4. Configurations

4.1 Introduction

The Roa Logic APB Checker VIP is a configurable Verification IP for the APB Bus. The core parameters, static configuration options, and functions for dynamic configuration are described in this section.

4.2 Core Configuration

The APB Checker VIP supports APB2, APB3, APB4, and APB5. The APB version is selected by setting either of these defines:

```
`define APB_VERSION_APB5
`define APB_VERSION_APB4
`define APB_VERSION_APB3
```

If no define is set, then the default is APB2.

4.3 Core Parameters

The parameter names used by the core are as specified by the APB Specification documents, which are owned and governed by ARM Ltd.

Parameter	Type	Default	Description
ADDR_WIDTH	Integer	32	Address bus width
DATA_WIDTH	Integer	32	Data bus widths
USER_REQ_WIDTH	Integer	0	User address bus width
USER_DATA_WDITH	Integer	0	User data bus widths
USER_RESP_WIDTH	Integer	0	User response bus width
CHECK_PSTRB	Integer	1	Enable PSTRB checking
CHECK_PPROT	Integer	1	Enable PPROT checking
CHECK_PSLVERR	Integer	1	Enable PSLVERR checking
WATCHDOG_TIMEOUT	Integer	128	Watchdog counter timeout value

Table 4.1: Core Parameters

4.3.1 ADDR_WIDTH

The ADDR_WIDTH parameter specifies the width of the APB2 and above PADDR signal. The default value of the ADDR_WIDTH parameter is 32.

4.3.2 DATA_WIDTH

The DATA_WIDTH parameter specifies the width of the APB2 and above PRDATA and PWDATA signals. The default value of the DATA_WIDTH parameter is 32.

4.3.3 USER_REQ_WIDTH

The USER_REQ_WIDTH parameter specifies the width of the APB5 PAUSER signal. A value of zero ('0') indicates the signal is not present in the APB bus and checking is disabled. The default value of the USER_REQ_WIDTH parameter is 0; i.e. disabled.

4.3.4 USER_DATA_WIDTH

The USER_DATA_WIDTH parameter specifies the width of the APB5 PRUSER and PWUSER signals. A value of zero ('0') indicates the signals are not present in the APB bus and checking is disabled. The default value of the USER_DATA_WIDTH parameter is 0; i.e. disabled.

4.3.5 USER_RESP_WIDTH

The USER_RESP_WIDTH parameter specifies the width of the APB5 PBUSER bus. A value of zero ('0') indicates the signal is not present in the APB bus and checking is disabled. The default value of the USER_RESP_WIDTH parameter is 0; i.e. disabled.

4.3.6 CHECK_PSTRB

The CHECK_PSTRB parameter enables or disables checking of the optional APB4 and above PSTRB signal. If CHECK_PSTRB has a value of zero (0), then checking the PSTRB signal is disabled. Any other value enables checking the PSTRB signal. The default value of the CHECK_PSTRB parameter is 1; i.e. enabled.

4.3.7 CHECK_PPROT

The CHECK_PPROT parameter enables or disables checking of the optional APB4 and above PPROT signal. If CHECK_PPROT has a value of zero (0), then checking the PPROT signal is disabled. Any other value enables checking the PPROT signal. The default value of the CHECK_PPROT parameter is 1; i.e. enabled.

4.3.8 CHECK_PSLVERR

The CHECK_PSLVERR parameter enables or disables checking of the optional APB3 and above PSLVERR signal. If CHECK_PSLVERR has a value of zero (0), then checking the PSLVERR signal is disabled. Any other value enables checking the PSLVERR signal. The default value of the CHECK_PSLVERR parameter is 1; i.e. enabled.

4.3.9 WATCHDOG_TIMEOUT

The WATCHDOG_TIMEOUT parameter sets the expiration counter value for the optional watchdog. A value of zero ('0') indicates the watchdog is disabled. The default value of the WATCHDOG_TIMEOUT parameter is 128.

4.4 Functions for Dynamic Configuration

The APB Checker VIP allows the user to dynamically change the severity level of each rule. Changing the severity level allows the user to stop the simulation when hitting a

certain rule, or completely ignoring a rule, for example. See the extending section for more details.

4.4.1 get_severity

Synopsis: `function automatic severity_t get_severity (input int msg_no)`

The `get_severity` function returns the severity level of message `msg_no`. Note that the rule number is one higher than the message number; `msg_no`=0 means rule #1.

4.4.2 set_severity

Synopsis: `task automatic set_severity (input int msg_no, severity_t severity)`

The `set_severity` function set the severity level of message `msg_no` to `severity`. Note that the rule number is one higher than the message number; `msg_no`=0 means rule #1.

5. Interfaces

5.1 APB Interface

The APB Interface is a configurable APB Interface. All signals defined in the protocol are supported as described below. See the *AMBA APB Protocol Specifications* for a complete description of the signals.

Port	Size	Direction	Version	Description
PRESETn	1	Input	APB2	Reset
PCLK	1	Input	APB2	Clock
PSEL	1	Input	APB2	Select
PENABLE	1	Input	APB2	Enable
PADDR	ADDR_WIDTH	Input	APB2	Address
PWRITE	1	Input	APB2	Direction
PSTRB	DATA_WIDTH/8	Input	APB4	Write Strobe
PPROT	3	Input	APB4	Protection Type
PWDATA	DATA_WIDTH	Input	APB2	Write Data
PRDATA	DATA_WIDTH	Input	APB2	Read Data
PREADY	1	Input	APB3	Ready
PSLVERR	1	Input	APB3	Transfer Error
PWAKEUP	1	Input	APB5	Wake-up
PAUSER	USER_REQ_WIDTH	Input	APB5	User request attribute
PWUSER	USER_DATA_WIDTH	Input	APB5	User write data attribute
PRUSER	USER_DATA_WIDTH	Input	APB5	User read data attribute
PBUSER	USER_RESP_WIDTH	Input	APB5	User response attribute

Table 5.1: APB Interface Ports

Signals for an APB version higher than selected are not present on the interface. See the Core Configuration section.

5.1.1 PRESETn

When the active low asynchronous PRESETn input is asserted ('0'), the APB interface is put into its initial reset state.

5.1.2 PCLK

PCLK is the APB interface clock. All APB signals are timed against the rising edge of PCLK.

The APB Bus Checker VIP requires a valid PCLK. All checks and rules trigger on the rising edge of PCLK.

5.1.3 PSEL

The APB *Requester* generates PSEL, signaling to a *Completer* that it is selected and that a data transfer is required.

5.1.4 PENABLE

PENABLE indicates the second and subsequent cycles of a transfer. The cycles when PENABLE is asserted ('1') are called the *Access Phase*. It is driven by the *Requester*.

5.1.5 PADDR

PADDR is the APB address bus. The bus width is defined by the ADDR_WIDTH parameter.

5.1.6 PWRITE

PWRITE indicates the direction of the transfer. When PWRITE is asserted ('1') it indicates a write access and a read data access when de-asserted ('0'). It is driven by the *Requester*.

5.1.7 PSTRB

PSTRB is an optional APB4 signal driven by the *Requester*.. It indicates which byte lane to update during a write transfer. There is one PSTRB signal per byte lane of the APB write data bus (PWDATA), such that PSTRB[n] corresponds to PWDATA[(8n+7):8n].

5.1.8 PPROT

PPROT is an optional APB4 signal driven by the *Requester*. It indicates the normal, privileged, or secure protection level of the transaction and whether the transaction is a data or an instruction access. PPROT has a width of 3 bits.

5.1.9 PWDATA

PWDATA is the APB write data bus and is driven by the *Requester* during write cycles, when PWRITE is asserted ('1'). The bus width is defined by the DATA_WIDTH parameter.

5.1.10 PRDATA

PRDATA is the APB read data bus and is driven by the *Completer* during read cycles, when PWRITE is de-asserted ('0'). The bus width is defined by the DATA_WIDTH parameter.

5.1.11 PREADY

PREADY is an APB3 signal driven by the *Completer*. It is used to extend an APB transfer.

5.1.12 PSLVERR

PSLVERR is an optional APB3 signal driven by the *Completer*. It indicates an error condition on the APB bus when asserted ('1').

5.1.13 PWAKEUP

PWAKEUP is an optional APB5 signal driven by the *Requester*. It indicates any activity associated with an APB interface.

5.1.14 PAUSER

PAUSER is an optional APB5 signal driven by the *Requester*. The bus width is defined by the USER_REQ_WIDTH parameter.

5.1.15 PWUSER

PWUSER is an optional APB5 signal driven by the *Requester*. The bus width is defined by the USER_DATA_WIDTH parameter.

5.1.16 PRUSER

PRUSER is an optional APB5 signal driven by the *Requester*. The bus width is defined by the USER_DATA_WIDTH parameter.

5.1.17 PBUSER

PBUSER is an optional APB5 signal driven by the *Requester*. The bus width is defined by the USER_RESP_WIDTH parameter.

6. Rules

6.1 Introduction

This section describes all the rules in numerical order. Waveform examples showing how a rule is triggered are provided for each rule. Only the relevant signals are shown in each waveform. The failing conditions are shown in red. Also the rule trigger is shown as a pseudo-signal in the waveform.

For reference, shown below is a waveform with examples of APB transfers with a 32bit data bus. The waveform shows how the core is brought out of sleep after the initial reset, followed by a read from address A with a single wait state, and a single byte write to address B with no wait states. The waveform also shows the Idle, Setup, and Access phases of the APB transfer.

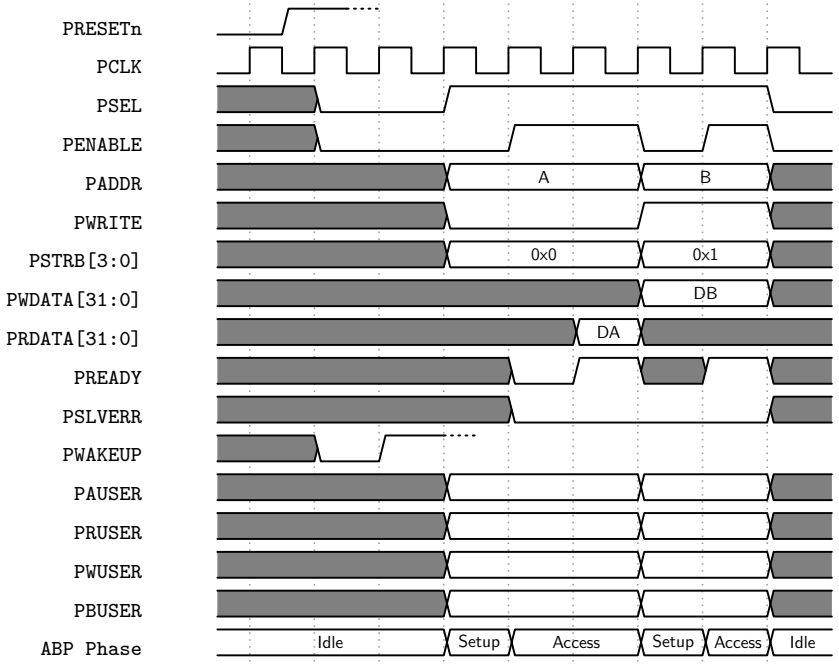

Figure 6.1: APB Transfer Examples

6.2 Rules

6.2.1 PSEL Must remain high for the entire transfer

Message: APB-1
Severity: ERROR
Description: The PSEL signal must remain asserted (‘1’) during the entire transfer.
APB Version: All

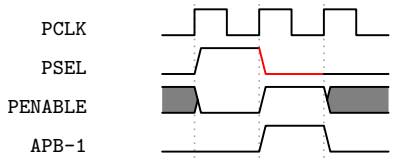

Figure 6.2: APB-1 Example

6.2.2 PSEL Undefined

Message: APB-2
Severity: ERROR
Description: The PSEL signal may never be undefined (‘x’) or (‘z’).
APB Version: All

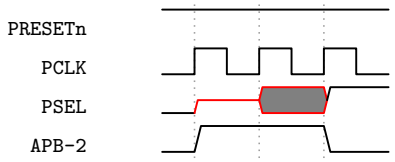

Figure 6.3: APB-2 Example

6.2.3 PENABLE must be low during Setup Phase

Message: APB-3
Severity: ERROR
Description: The PENABLE signal must be low (‘0’) during the first cycle (the setup phase) of a transfer.
APB Version: All

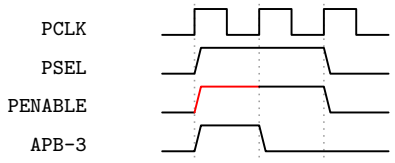

Figure 6.4: APB-3 Example

6.2.4 PENABLE must be high during Access Phase

Message: APB-4
Severity: ERROR
Description: The PENABLE signal must be high ('1') during the second and consecutive cycles (the access phases) of a transfer.
APB Version: All

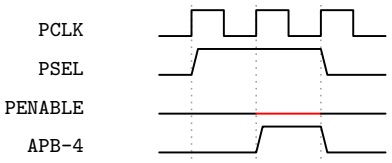

Figure 6.5: APB-4 Example

6.2.5 PENABLE undefined

Message: APB-5
Severity: ERROR
Description: The PENABLED signal may never be undefined ('x') or ('z') during a transfer.
APB Version: All

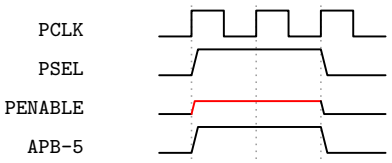

Figure 6.6: APB-5 Example

6.2.6 PADDR must remain stable for the entire transfer

Message: APB-6
Severity: ERROR
Description: The PADDR signal may not change during a transfer.
APB Version: All

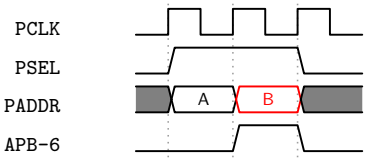

Figure 6.7: APB-6 Example

6.2.7 PADDR versus PSTRB misaligned

Message: APB-7
Severity: ERROR
Description: The PADDR signal value must be aligned with the transfer size indicated by the PSTRB signal value during a write transfer.
APB Version: from APB4

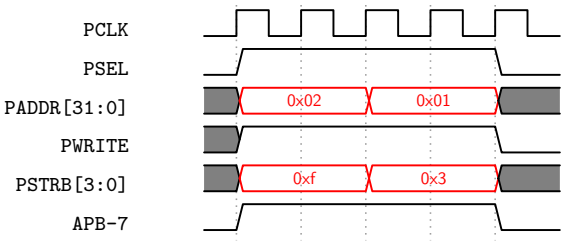

Figure 6.8: APB-7 Example

6.2.8 PADDR should be aligned to DATA_WIDTH

Message: APB-8
Severity: ERROR
Description: The PADDR signal value must be aligned with the DATA_WIDTH parameter value during a transfer.
APB Version: All

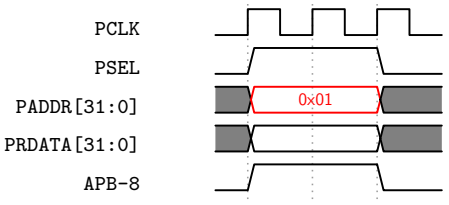

Figure 6.9: APB-8 Example

6.2.9 PADDR undefined

Message: APB-9
Severity: ERROR
Description: The PADDR signal may never be undefined (‘x’) or (‘z’) during a transfer.
APB Version: All

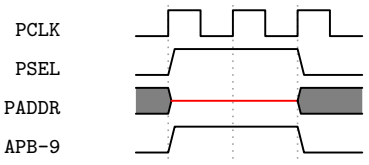

Figure 6.10: APB-9 Example

6.2.10 PWRITE must remain stable for the entire transfer

Message: APB-10
Severity: ERROR
Description: The PWRITE signal may not change during a transfer.
APB Version: All

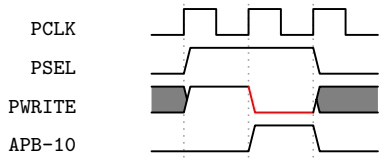

Figure 6.11: APB-10 Example

6.2.11 PWRITE undefined

Message: APB-11
Severity: ERROR
Description: The PWRITE signal may never be undefined ('x') or ('z') during a transfer.
APB Version: All

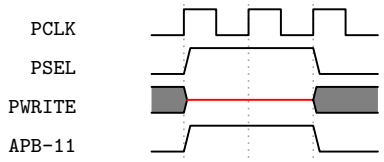

Figure 6.12: APB-11 Example

6.2.12 PSTRB value non byte/word/dword/...

Message: APB-12
Severity: WARNING
Description: The PSTRB signal holds a strange value during a write transfer.
APB Version: from APB4

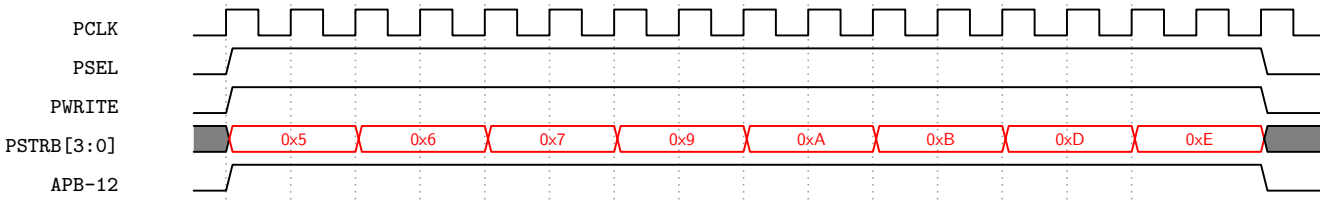

Figure 6.13: APB-12 Example

6.2.13 PSTRB must remain stable for the entire transfer

Message: APB-13
Severity: ERROR
Description: The PSTRB signal may not change during a transfer.
APB Version: from APB4

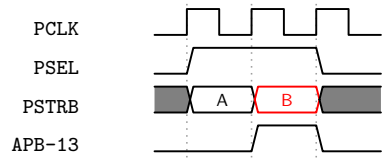

Figure 6.14: APB-13 Example

6.2.14 PSTRB undefined

Message: APB-14
Severity: ERROR
Description: The PSTRB signal may never be undefined ('x') or ('z') during a transfer.
APB Version: from APB4

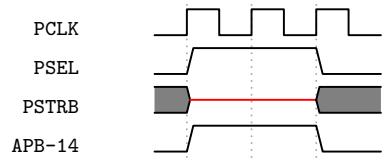

Figure 6.15: APB-14 Example

6.2.15 PPROT must remain stable for the entire transfer

Message: APB-15
Severity: ERROR
Description: The PPROT signal may not change during a transfer.
APB Version: from APB4

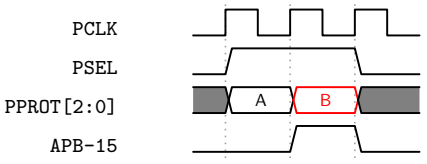

Figure 6.16: APB-15 Example

6.2.16 PPROT undefined

Message: APB-16
Severity: ERROR
Description: The PPROT signal may never be undefined (‘x’) or (‘z’) during a transfer.
APB Version: from APB4

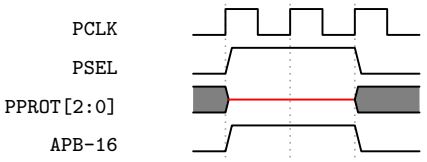

Figure 6.17: APB-16 Example

6.2.17 PWDATA must remain stable for the entire transfer

Message: APB-17
Severity: ERROR
Description: The PWDATA signal may not change during a write transfer.
APB Version: All

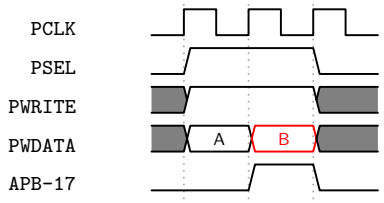

Figure 6.18: APB-17 Example

6.2.18 PWDATA contains ’x’

Message: APB-18
Severity: WARNING
Description: One or more bits of the PWDATA signal are undefined (‘x’) or (‘z’) during a write transfer.
APB Version: APB2, APB3

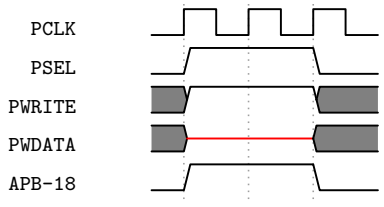

Figure 6.19: APB-18 Example

6.2.19 PWDATA contains 'x'

Message: APB-19
Severity: WARNING
Description: One or more bits of the PWDATA signal, in a byte not masked by PSTRB, are undefined ('x') or ('z') during a write transfer.
APB Version: from APB4

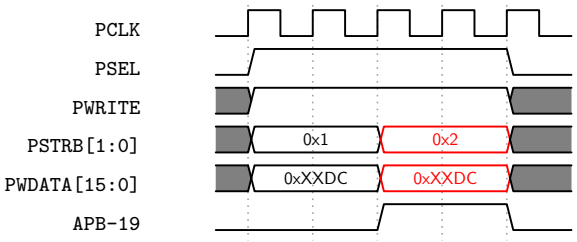

Figure 6.20: APB-19 Example

6.2.20 PRDATA contains 'x'

Message: APB-20
Severity: WARNING
Description: One or more bits of the PRDATA signal are undefined ('x') or ('z') during a read transfer.
APB Version: All

Figure 6.21: APB-20 Example

6.2.21 PREADY undefined during Access phase

Message: APB-21
Severity: ERROR
Description: The PREADY signal is undefined ('x') or ('z') during the access phase of a transfer.
APB Version: from APB3

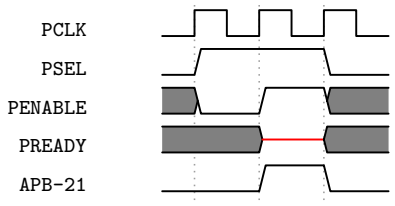

Figure 6.22: APB-21 Example

6.2.22 PSLVERR undefined

Message: APB-22
Severity: ERROR
Description: The PSLVERR signal is undefined ('x') or ('z') during the final cycle of a transfer.
APB Version: from APB3

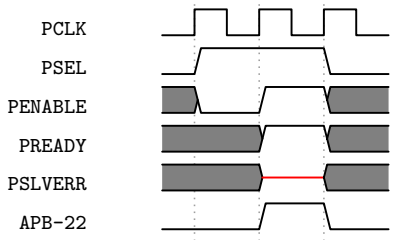

Figure 6.23: APB-22 Example

6.2.23 Watchdog expired

Message: APB-23
Severity: FATAL
Description: The optional watchdog counter expired.
APB Version: from APB3

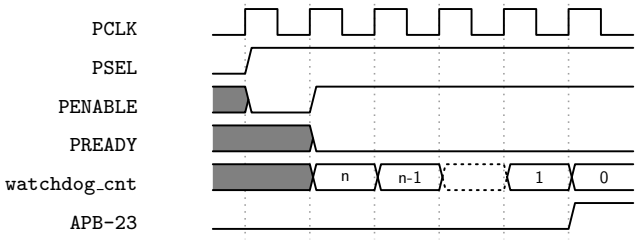

Figure 6.24: APB-23 Example

6.2.24 PWAKEUP must remain high until the end of the transfer

Message: APB-24
Severity: ERROR
Description: PWAKEUP must remain high ('1') until PREADY is high ('1'), if both PSEL and PWAKEUP are high ('1').
APB Version: from APB5

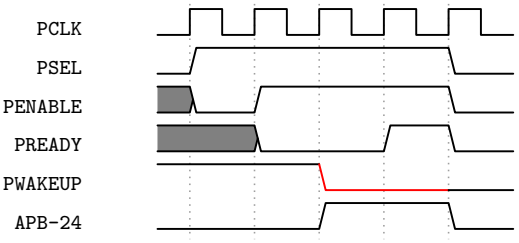

Figure 6.25: APB-24 Example

6.2.25 PWAKEUP should be asserted at least one cycle before PSEL

Message: APB-25
Severity: WARNING
Description: PWAKEUP should be high (‘1’) at least 1 PCLK cycle before PSEL goes high (‘1’).
APB Version: from APB5

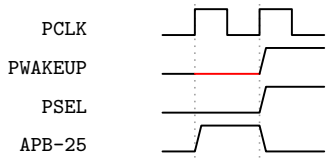

Figure 6.26: APB-25 Example

6.2.26 PWAKEUP raised without starting a transfer

Message: APB-26
Severity: WARNING
Description: PWAKEUP should not be raised without starting a transfer.
APB Version: from APB5

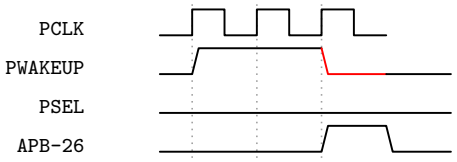

Figure 6.27: APB-26 Example

6.2.27 PWAKEUP undefined

Message: APB-27
Severity: ERROR
Description: The PWAKEUP signal may never be undefined (‘x’) or (‘z’).
APB Version: from APB5

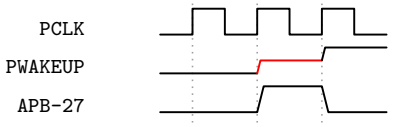

Figure 6.28: APB-27 Example

6.2.28 PAUSER must remain stable for the entire transfer

Message: APB-28
Severity: ERROR
Description: The PAUSER signal may not change during a transfer.
APB Version: from APB5

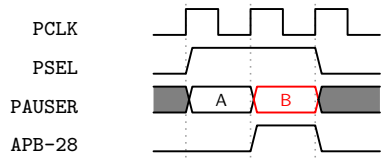

Figure 6.29: APB-28 Example

6.2.29 PAUSER undefined

Message: APB-29
Severity: ERROR
Description: The PAUSER signal may never be undefined ('x') or ('z') during a transfer.
APB Version: from APB5

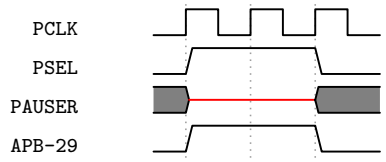

Figure 6.30: APB-29 Example

6.2.30 PAUSER should be max 128 bits

Message: APB-30
Severity: WARNING
Description: The PAUSER signal width should be less than 128 bits.
APB Version: from APB5

6.2.31 PWUSER must remain stable for the entire transfer

Message: APB-31
Severity: ERROR
Description: The PWUSER signal may not change during a transfer.
APB Version: from APB5

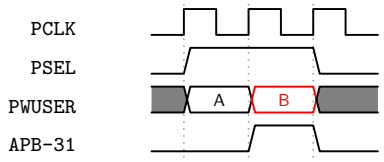

Figure 6.31: APB-31 Example

6.2.32 PWUSER undefined

Message: APB-32
Severity: ERROR
Description: The PWUSER signal may never be undefined ('x') or ('z') during a transfer.
APB Version: from APB5

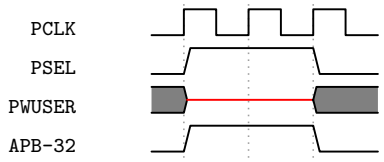

Figure 6.32: APB-32 Example

6.2.33 PWUSER should be max DATA_WIDTH/2 bits

Message: APB-33
Severity: WARNING
Description: The PWUSER signal width should be less than DATA_WIDTH/2 bits.
APB Version: from APB5

6.2.34 PRUSER contains 'x'

Message: APB-34
Severity: WARNING
Description: One or more bits of the PRUSER signal are undefined ('x') or ('z') during a read transfer.
APB Version: from APB5

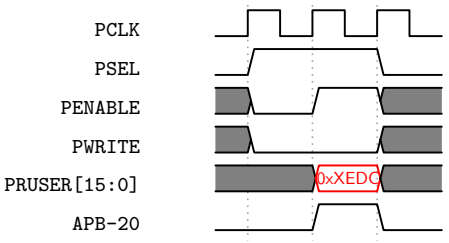

Figure 6.33: APB-34 Example

6.2.35 PRUSER should be max DATA_WIDTH/2 bits

Message: APB-35
Severity: WARNING
Description: The PRUSER signal width should be less than DATA_WIDTH/2 bits.
APB Version: from APB5

6.2.36 PBUSER contains 'x'

Message: APB-36
Severity: WARNING
Description: One or more bits of the PBUSER signal are undefined ('x') or ('z') during the final cycle of a transfer.
APB Version: from APB5

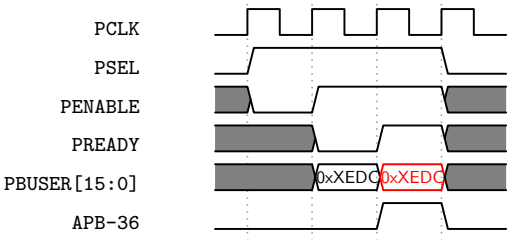

Figure 6.34: APB-36 Example

6.2.37 PBUSER should be max 16 bits

Message: APB-37
Severity: WARNING
Description: The PBUSER signal width should be less than 16 bits.
APB Version: from APB5

6.2.38 PSTRB must be low during read transfer

Message: APB-38
Severity: ERROR
Description: The PSTRB signal must be low (all '0') during a read transfer.
APB Version: from APB4

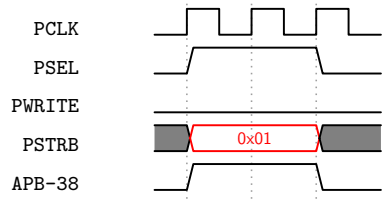

Figure 6.35: APB-38 Example

6.2.39 PADDR should be max 32 bits

Message: APB-39
Severity: WARNING
Description: The PADDR signal width should be less than 32 bits.
APB Version: All

6.2.40 PWDATA should be max 8, 16, or 32 bits wide

Message: APB-40
Severity: WARNING
Description: The PWDATA signal width should be either 8, 16, or 32 bits.
APB Version: All

6.2.41 PRDATA should be max 8, 16, or 32 bits wide

Message: APB-41
Severity: WARNING
Description: The PRDATA signal width should be either 8, 16, or 32 bits.
APB Version: All

6.2.42 PRESETn Undefined

Message: APB-42
Severity: ERROR
Description: The PRESETn signal may never be undefined ('x') or ('z').
APB Version: All

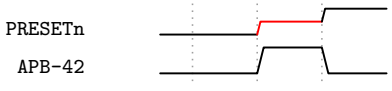

Figure 6.36: APB-42 Example

6.2.43 PCLK Undefined

Message: APB-43
Severity: ERROR
Description: The PCLK signal may never be undefined (‘x’) or (‘z’).
APB Version: All

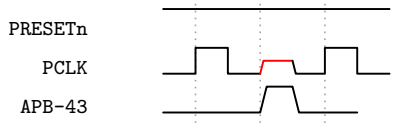

Figure 6.37: APB-43 Example

6.3 Rules per signal

Signal	MsgNo	Message
PRESETn	42	PRESETn Undefined
PCLK	43	PCLK Undefined
PSEL	1	PSEL Must remain high for the entire transfer
PSEL	2	PSEL Undefined
PENABLE	3	PENABLE must be low during Setup Phase
PENABLE	4	PENABLE must be high during Access Phase
PENABLE	5	PENABLE undefined
PADDR	6	PADDR must remain stable for the entire transfer
PADDR	7	PADDR versus PSTRB misaligned
PADDR	8	PADDR should be aligned to DATA_WIDTH
PADDR	9	PADDR undefined
PADDR	39	PADDR should be max 32 bits
PWRITE	10	PWRITE must remain stable for the entire transfer
PWRITE	11	PWRITE undefined
PSTRB	12	PSTRB value non byte/word/dword/..
PSTRB	13	PSTRB must remain stable for the entire transfer
PSTRB	14	PSTRB undefined
PSTRB	38	PSTRB must be low during read transfer
PPROT	15	PPROT must remain stable for the entire transfer
PPROT	16	PPROT undefined
PWDATA	17	PWDATA must remain stable for the entire transfer
PWDATA	18	PWDATA contains 'x'
PWDATA	19	PWDATA contains 'x'
PWDATA	40	PWDATA should be max 8, 16, or 32 bits wide
PRDATA	20	PRDATA contains 'x'
PRDATA	41	PRDATA should be max 8, 16, or 32 bits wide
PREADY	21	PREADY undefined during Access phase
PSLVERR	22	PSLVERR undefined
PWAKEUP	24	PWAKEUP must remain high until the end of the transfer
PWAKEUP	25	PWAKEUP should be asserted at least one cycle before PSEL
PWAKEUP	26	PWAKEUP raised without starting a transfer
PWAKEUP	27	PWAKEUP undefined
PAUSER	28	PAUSER must remain stable for the entire transfer
PAUSER	29	PAUSER undefined
PAUSER	30	PAUSER should be max 128 bits
PWUSER	31	PWUSER must remain stable for the entire transfer
PWUSER	32	PWUSER undefined
PWUSER	33	PWUSER should be max DATA_WIDTH/2 bits
PRUSER	34	PRUSER contains 'x'
PRUSER	35	PRUSER should be max DATA_WIDTH/2 bits
PBUSER	36	PBUSER contains 'x'
PBUSER	37	PBUSER should be max 16 bits

Table 6.1: Rules per signal

7. Extending the VIP

7.1 Introduction

Extending the VIP

8. Bibliography

[1] Arm Ltd., "AMBA APB Protocol Specifications," *https://developer.arm.com/documentation/ihi0024/latest/*, 2021.

9. Revision History

Date	Rev.	Comments
17-Apr-2024	1.0	Initial Release Added preface

Table 9.1: Revision History