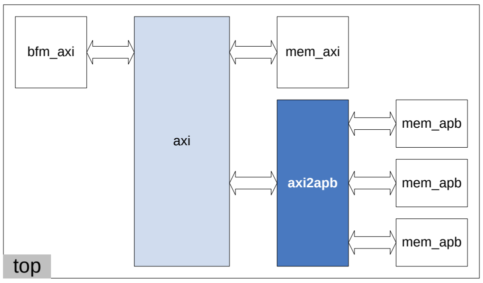
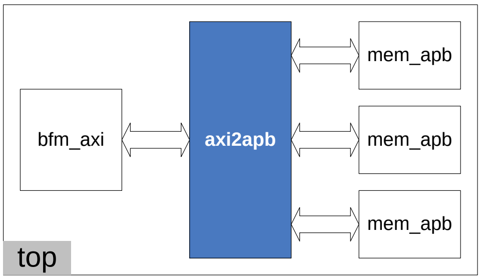
AMBA AXI to APB Design

2013 – 2017

Ando Ki
(adki@future-ds.com)

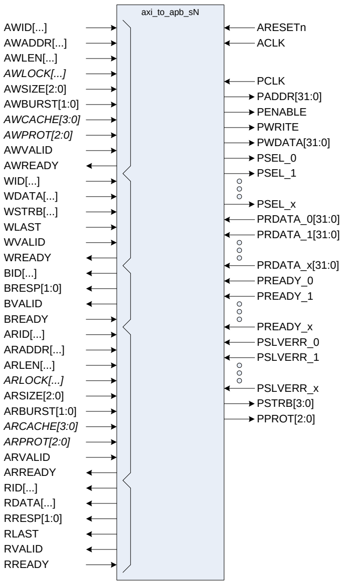
AMBA AXI-to-APB design & verification





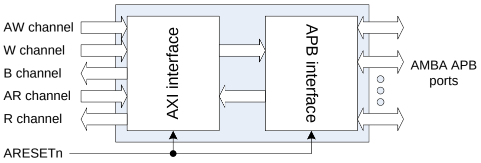
- Test-bench includes 'BFM', 'AMBA AXI', and 'MEMORY'.
- BFM generates test-pattern and test-vector.

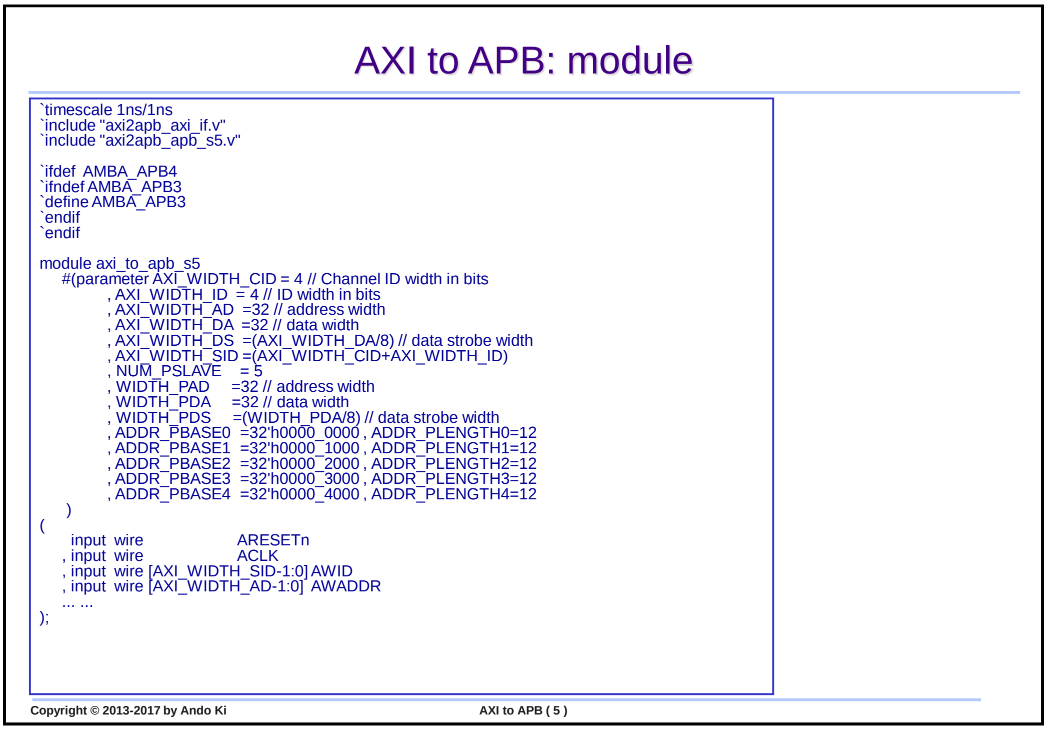
AXI-to-APB bus bridge



- ACLK and PCLK can be asynchronous

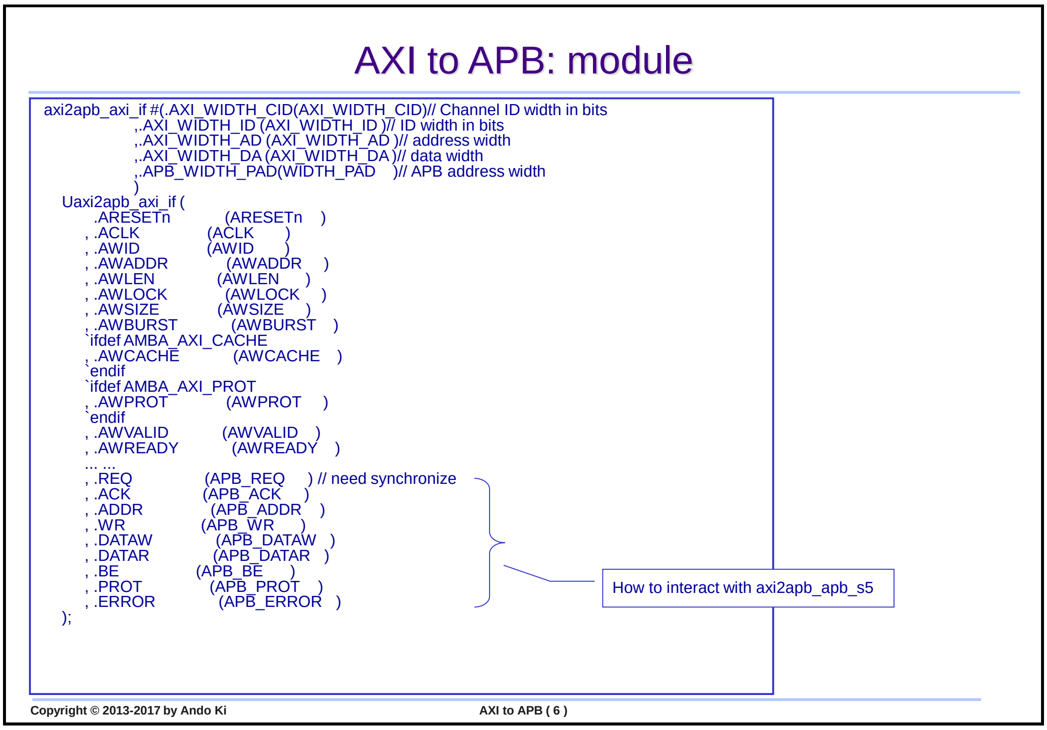
AXI-to-APB internal





AXI to APB: module

```
`timescale 1ns/1ns
`include "axi2apb_axi_if.v"
`include "axi2apb_apb_s5.v"

`ifdef  AMBA_APB4
`ifndef AMBA_APB3
`define AMBA_APB3
`endif
`endif

module axi_to_apb_s5
     #(parameter AXI_WIDTH_CID = 4 // Channel ID width in bits
               , AXI_WIDTH_ID  = 4 // ID width in bits
               , AXI_WIDTH_AD  =32 // address width
               , AXI_WIDTH_DA  =32 // data width
               , AXI_WIDTH_DS  =(AXI_WIDTH_DA/8) // data strobe width
               , AXI_WIDTH_SID =(AXI_WIDTH_CID+AXI_WIDTH_ID)
               , NUM_PSLAVE    = 5
               , WIDTH_PAD     =32 // address width
               , WIDTH_PDA     =32 // data width
               , WIDTH_PDS     =(WIDTH_PDA/8) // data strobe width
               , ADDR_PBASE0  =32'h0000_0000 , ADDR_PLENGTH0=12
               , ADDR_PBASE1  =32'h0000_1000 , ADDR_PLENGTH1=12
               , ADDR_PBASE2  =32'h0000_2000 , ADDR_PLENGTH2=12
               , ADDR_PBASE3  =32'h0000_3000 , ADDR_PLENGTH3=12
               , ADDR_PBASE4  =32'h0000_4000 , ADDR_PLENGTH4=12
      )
(
       input  wire                     ARESETn
     , input  wire                     ACLK
     , input  wire [AXI_WIDTH_SID-1:0] AWID
     , input  wire [AXI_WIDTH_AD-1:0]  AWADDR
     ... ...
);
```



AXI to APB: module

```
axi2apb_axi_if #(.AXI_WIDTH_CID(AXI_WIDTH_CID)// Channel ID width in bits
                    ,.AXI_WIDTH_ID (AXI_WIDTH_ID )// ID width in bits
                    ,.AXI_WIDTH_AD (AXI_WIDTH_AD )// address width
                    ,.AXI_WIDTH_DA (AXI_WIDTH_DA )// data width
                    ,.APB_WIDTH_PAD(WIDTH_PAD    )// APB address width
                    )
    Uaxi2apb_axi_if (
           .ARESETn          (ARESETn    )
         , .ACLK             (ACLK       )
         , .AWID             (AWID       )
         , .AWADDR           (AWADDR     )
         , .AWLEN            (AWLEN      )
         , .AWLOCK           (AWLOCK     )
         , .AWSIZE           (AWSIZE     )
         , .AWBURST          (AWBURST    )
         `ifdef AMBA_AXI_CACHE
         , .AWCACHE          (AWCACHE    )
         `endif
         `ifdef AMBA_AXI_PROT
         , .AWPROT           (AWPROT     )
         `endif
         , .AWVALID          (AWVALID    )
         , .AWREADY          (AWREADY    )
         ... ...
         , .REQ              (APB_REQ    ) // need synchronize
         , .ACK              (APB_ACK    )
         , .ADDR             (APB_ADDR   )
         , .WR               (APB_WR     )
         , .DATAW            (APB_DATAW  )
         , .DATAR            (APB_DATAR  )
         , .BE               (APB_BE     )
         , .PROT             (APB_PROT   )
         , .ERROR            (APB_ERROR  )
    );
```

How to interact with axi2apb_apb_s5

AXI to APB: module

```
axi2apb_apb_s5 #(.NUM_PSLAVE   (NUM_PSLAVE   )
                ,.WIDTH_PAD    (WIDTH_PAD    )
                ,.WIDTH_PDA    (WIDTH_PDA    )
                ,.ADDR_PBASE0  (ADDR_PBASE0  ) ,.ADDR_PLENGTH0(ADDR_PLENGTH0)
                ,.ADDR_PBASE1  (ADDR_PBASE1  ) ,.ADDR_PLENGTH1(ADDR_PLENGTH1)
                ,.ADDR_PBASE2  (ADDR_PBASE2  ) ,.ADDR_PLENGTH2(ADDR_PLENGTH2)
                ,.ADDR_PBASE3  (ADDR_PBASE3  ) ,.ADDR_PLENGTH3(ADDR_PLENGTH3)
                ,.ADDR_PBASE4  (ADDR_PBASE4  ) ,.ADDR_PLENGTH4(ADDR_PLENGTH4)
                )
  Uaxi2apb_apb (
       .PRESETn   (PRESETn   )
     , .PCLK      (PCLK      )
     , .PADDR     (PADDR     )
     , .PENABLE   (PENABLE   )
     , .PWRITE    (PWRITE    )
     , .PWDATA    (PWDATA    )
     , .PSEL_0    (PSEL_0    )
     ...
     , .PRDATA_0  (PRDATA_0  )
     ...
    `ifdef AMBA_APB3
     , .PREADY_0  (PREADY_0  )
     ...
     , .PSLVERR_0 (PSLVERR_0 )
     ...
    `endif
     ...
     , .REQ    (APB_REQ  )
     , .ACK    (APB_ACK  )
     , .ADDR   (APB_ADDR )
     , .WR     (APB_WR   )
     , .DATAW  (APB_DATAW)
     , .DATAR  (APB_DATAR)
     , .BE     (APB_BE   )
     , .PROT   (APB_PROT )
     , .ERROR  (APB_ERROR)
  );
```

How to interact with axi2apb_axi_if

Simulation with ModelSim (1/4)

```
# Makefile
SHELL                   = /bin/sh
MAKEFILE     = Makefile
#--------------------------------------------------------
VLIB         = $(shell which vlib)
VLOG         = $(shell which vlog)
VSIM         = $(shell which vsim)
WORK         = work
#--------------------------------------------------------
TOP          = top
#--------------------------------------------------------
all: vlib compile simulate

vlib:
             if [ -d $(WORK) ]; then /bin/rm -rf $(WORK); fi
             $(VLIB) $(WORK)

compile:
             $(VLOG) -lint -work $(WORK)  -f modelsim.args

simulate: compile
             $(VSIM) -novopt -c -do "run -all; quit" $(WORK).$(TOP)
```

Modelsim commands

Specify where to store compile results

Compilation

Simulation

```
@ECHO OFF
REM RunMe.bat
SET MODELSIMWORK=work
SET MODELSIMVLIB=vlib
SET MODELSIMVSIM=vsim
SET MODELSIMVCOM=vcom
SET MODELSIMVLOG=vlog

SET DESIGNTOP=top

IF EXIST %MODELSIMWORK% RMDIR /S/Q %MODELSIMWORK%

%MODELSIMVLIB% %MODELSIMWORK%
%MODELSIMVLOG% -work %MODELSIMWORK% -lint^
          -f modelsim.args
%MODELSIMVSIM% -novopt -c -do "run -all; quit"^
          %MODELSIMWORK%.%DESIGNTOP%
```

Simulation with ModelSim (2/4)

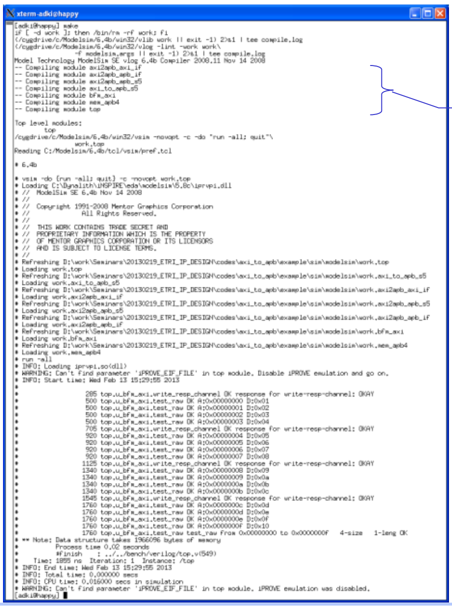
modelsim.args

```
//-------------------------------------------------------------------
+incdir+../../design/verilog
+incdir+../../../../bfm_axi_task/example/design/verilog
        ./sim_define.v
        ../../design/verilog/axi_to_apb_s5.v
        ../../design/verilog/bfm_axi.v
        ../../design/verilog/mem_apb4.v
//-------------------------------------------------------------------
// Below are test-bench
//-------------------------------------------------------------------
+incdir+../../bench/verilog
        ../../bench/verilog/top.v
//-------------------------------------------------------------------
```

sim_define.v

```
`ifndef _SIM_DEFINE_V_
`define _SIM_DEFNE_V_
//-----------------------------------------------------------------------
`define SIM      // define this for simulation case if you are not sure
`define VCD      // define this for VCD waveform dump
`define DEBUG
`define RIGOR
`define LOW_POWER
//-----------------------------------------------------------------------
`define AMBA_APB4
//-----------------------------------------------------------------------
`endif
```

Simulation with ModelSim (3/4)

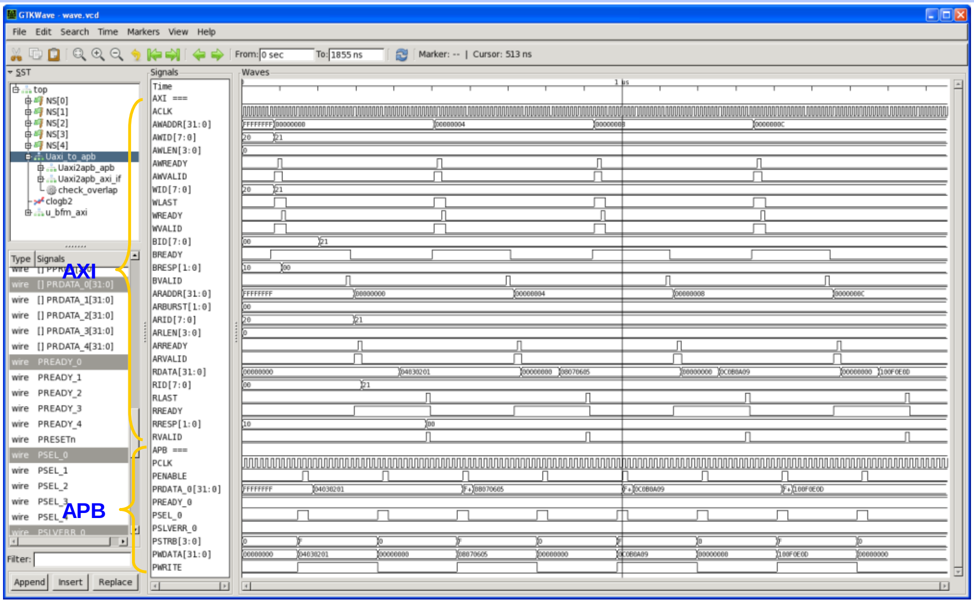


```
[adki@happy] make
if [ -d work ]; then /bin/rm -rf work; fi
(/cygdrive/c/Modelsim/6.4b/win32/vlib work || exit -1) 2>&1 | tee compile.log
(/cygdrive/c/Modelsim/6.4b/win32/vlog -lint -work work\
                -f modelsim.args || exit -1) 2>&1 | tee compile.log
Model Technology ModelSim SE vlog 6.4b Compiler 2008.11 Nov 14 2008
-- Compiling module axi2apb_axi_if
-- Compiling module axi2apb_apb_if
-- Compiling module axi2apb_apb_s5
-- Compiling module axi_to_apb_s5
-- Compiling module bfm_axi
-- Compiling module mem_apb4
-- Compiling module top

Top level modules:
        top
/cygdrive/c/Modelsim/6.4b/win32/vsim -novopt -c -do "run -all; quit"\
                work.top
Reading C:/Modelsim/6.4b/tcl/vsim/pref.tcl

# 6.4b

# vsim -do {run -all; quit} -c -novopt work.top
# Loading C:\Dynalith\iNSPIRE\eda\modelsim\5.8c\iprvpi.dll
# //  ModelSim SE 6.4b Nov 14 2008
# //
# //  Copyright 1991-2008 Mentor Graphics Corporation
# //              All Rights Reserved.
# //
# //  THIS WORK CONTAINS TRADE SECRET AND
# //  PROPRIETARY INFORMATION WHICH IS THE PROPERTY
# //  OF MENTOR GRAPHICS CORPORATION OR ITS LICENSORS
# //  AND IS SUBJECT TO LICENSE TERMS.
# //
# Refreshing D:\work\Seminars\20130219_ETRI_IP_DESIGN\codes\axi_to_apb\example\sim\modelsim\work.top
# Loading work.top
# Refreshing D:\work\Seminars\20130219_ETRI_IP_DESIGN\codes\axi_to_apb\example\sim\modelsim\work.axi_to_apb_s5
# Loading work.axi_to_apb_s5
# Refreshing D:\work\Seminars\20130219_ETRI_IP_DESIGN\codes\axi_to_apb\example\sim\modelsim\work.axi2apb_axi_if
# Loading work.axi2apb_axi_if
# Refreshing D:\work\Seminars\20130219_ETRI_IP_DESIGN\codes\axi_to_apb\example\sim\modelsim\work.axi2apb_apb_s5
# Loading work.axi2apb_apb_s5
# Refreshing D:\work\Seminars\20130219_ETRI_IP_DESIGN\codes\axi_to_apb\example\sim\modelsim\work.axi2apb_apb_if
# Loading work.axi2apb_apb_if
# Refreshing D:\work\Seminars\20130219_ETRI_IP_DESIGN\codes\axi_to_apb\example\sim\modelsim\work.bfm_axi
# Loading work.bfm_axi
# Refreshing D:\work\Seminars\20130219_ETRI_IP_DESIGN\codes\axi_to_apb\example\sim\modelsim\work.mem_apb4
# Loading work.mem_apb4
# run -all
# INFO: Loading iprvpi.so(dll)
# WARNING: Can't find parameter 'iPROVE_EIF_FILE' in top module. Disable iPROVE emulation and go on.
# INFO: Start time: Wed Feb 13 15:29:55 2013
#
#                  285 top.u_bfm_axi.write_resp_channel OK response for write-resp-channel: OKAY
#                  500 top.u_bfm_axi.test_raw OK A:0x00000000 D:0x01
#                  500 top.u_bfm_axi.test_raw OK A:0x00000001 D:0x02
#                  500 top.u_bfm_axi.test_raw OK A:0x00000002 D:0x03
#                  500 top.u_bfm_axi.test_raw OK A:0x00000003 D:0x04
#                  705 top.u_bfm_axi.write_resp_channel OK response for write-resp-channel: OKAY
#                  920 top.u_bfm_axi.test_raw OK A:0x00000004 D:0x05
#                  920 top.u_bfm_axi.test_raw OK A:0x00000005 D:0x06
#                  920 top.u_bfm_axi.test_raw OK A:0x00000006 D:0x07
#                  920 top.u_bfm_axi.test_raw OK A:0x00000007 D:0x08
#                 1125 top.u_bfm_axi.write_resp_channel OK response for write-resp-channel: OKAY
#                 1340 top.u_bfm_axi.test_raw OK A:0x00000008 D:0x09
#                 1340 top.u_bfm_axi.test_raw OK A:0x00000009 D:0x0a
#                 1340 top.u_bfm_axi.test_raw OK A:0x0000000a D:0x0b
#                 1340 top.u_bfm_axi.test_raw OK A:0x0000000b D:0x0c
#                 1545 top.u_bfm_axi.write_resp_channel OK response for write-resp-channel: OKAY
#                 1760 top.u_bfm_axi.test_raw OK A:0x0000000c D:0x0d
#                 1760 top.u_bfm_axi.test_raw OK A:0x0000000d D:0x0e
#                 1760 top.u_bfm_axi.test_raw OK A:0x0000000e D:0x0f
#                 1760 top.u_bfm_axi.test_raw OK A:0x0000000f D:0x10
#                 1760 top.u_bfm_axi.test_raw test_raw from 0x00000000 to 0x0000000f   4-size   1-leng OK
# ** Note: Data structure takes 1966096 bytes of memory
#          Process time 0.02 seconds
#          $finish    : ../../bench/verilog/top.v(549)
#    Time: 1855 ns  Iteration: 1  Instance: /top
# INFO: End time: Wed Feb 13 15:29:55 2013
# INFO: Total time: 0.000000 secs
# INFO: CPU time: 0.016000 secs in simulation
# WARNING: Can't find parameter 'iPROVE_EIF_FILE' in top module. iPROVE emulation was disabled.
[adki@happy] █
```

Compilation

Simulation according to the scenario

Simulation with ModelSim (4/4)



Example: AXI to APB case

- This example shows how to use BFM with tasks
  - Note that this design uses codes in "$(PROJECT)/codes/bfm_axi_task/design/verilog"
  - Step 1: go to your project directory
    - [user@host] cd $(PROJECT)/codes/axi_to_apb
  - Step 2: see the codes
    - [user@host] cd $(PROJECT)/codes/axi_to_apb/desing/verilog
  - Step 3: compile and run
    - [user@host] cd $(PROJECT)/codes/axi_to_apb/sim/modelsim
    - [user@host] make
  - Step 4: waveform view
    - [user@host] gtkwave wave.vcd &

```
[user@host] cd $(PROJECT)/codes/axi_to_apb/sim/modelsim
[user@host] make
[user@host] gtkwave wave.vcd &
```