Session 7->9: Project 1
8-bit timer verification using Systemverilog

Project 1

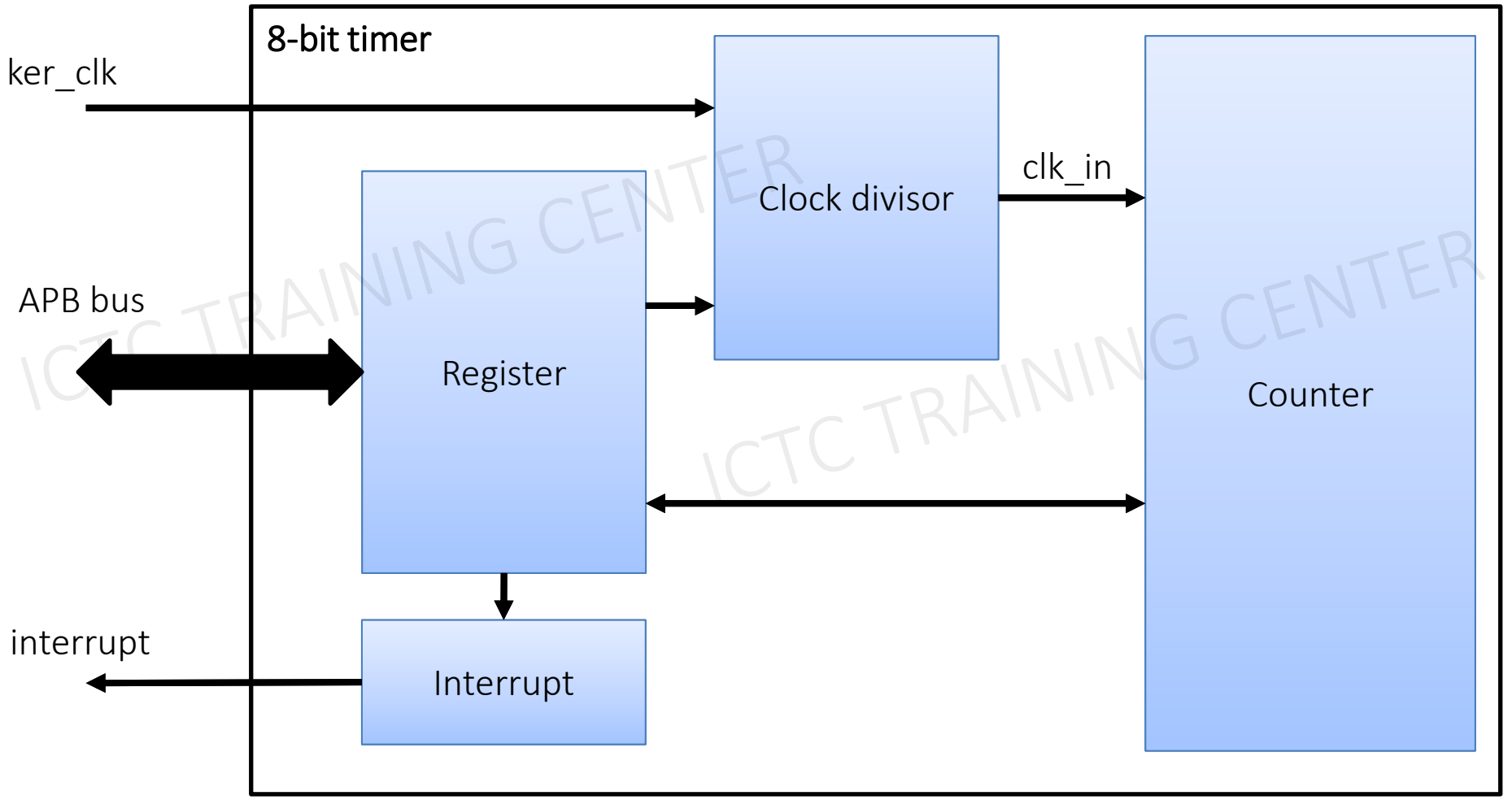
8-bit Timer Feature

Timer 8-bit feature list shown as below:

- ❑ Configure register via APB protocol (IP is APB slave).
- ❑ Only support APB transfer with no wait states and no error handling.
- ❑ Timer can count up and down.
- ❑ Detect overflow (count reach 255) and underflow (count reach 0).
- ❑ Support polling and interrupt (can be enable or disable).
- ❑ 2 clocks domain:
  - One for register configuration (pclk): 50 MHz
  - Other for clock divisor (ker_clk): 200 MHz

Project 1: 8-bit timer

Block diagram



Project 1: 8-bit timer

Signal definition

Signal name	Width	I/O type	Description
ker_clk	1	input	
pclk	1	input	
presetn	1	input	
paddr	[7:0]	Input	
psel	1	input	
penable	1	input	
pwrite	1	Input	
pwdata	[7:0]	input	
pready	1	output	
prdata	[7:0]	output	
interrupt	1	output	

Project 1: 8-bit timer

Register summary

Register attribute

Register attribute	Description
RO	Read-Only
RW	Read-Write
R/W1C	Read-Write 1 to clear. This only set by hardware and clear by software
Rsvd	Write not affect, Read as Zero

Register summary

Address	Register name	Description
0x00	Timer Configuration Register (TCR)	
0x01	Timer Status Register (TSR)	
0x02	Timer Data Register (TDR)	
0x03	Timer Interrupt Enable Register (TIE)	
0x04 -> 0xFF	Reserved region	Write not affect, Read as Zero

Project 1: 8-bit timer

Register detail (1)

0x00: Timer Configuration Register (TCR)

Bit	Name	Type	Default value	Description
7:5	-	-	Rsvd	
4:3	clk_div	R/W	2'b00	Clock divisor Divide ker_clk and supply to Counter • 2'b00: No divide • 2'b01: Divide by 2, clock supply to Counter block will be 100 MHz • 2'b10: Divide by 4, clock supply to Counter block will be 50 MHz • 2'b11: Divide by 8, clock supply to Counter block will be 25 MHz
2	load	R/W	1'b0	Load data from TDR into Counter as initial value • 1: Load data to Counter • 0: Normal operate When write value of this bit is 1, Counter will stop counting and data in TDR register will load into counter
1	count_down	R/W	1'b0	Counter up and down • 1: count down • 0: count up
0	timer_en	R/W	1'b0	Timer count enable • 0: Timer not count • 1: Timer start count **Note:** clk_div, load and count_down should configure when this bit is 1'b0

Project 1: 8-bit timer

Register detail (2)

0x01: Timer Status Register (TSR)

Bit	Name	Type	Default value	Description
7:2	-	-	Rsvd	
1	underflow	R/W1C	1'b0	Underflow This bit will trigger when Counter transit from 0 -> 255 Write 1'b1 to clear status of this bit
0	overflow	R/W1C	1'b0	Overflow This bit will trigger when Counter transit from 255 -> 0 Write 1'b1 to clear status of this bit

0x02: Timer Data Register (TDR)

Bit	Name	Type	Default value	Description
7:0	data	R/W	8'h00	Data load into Counter when register TCR bit 2 (LOAD) is set

Project 1: 8-bit timer

Register detail (3)

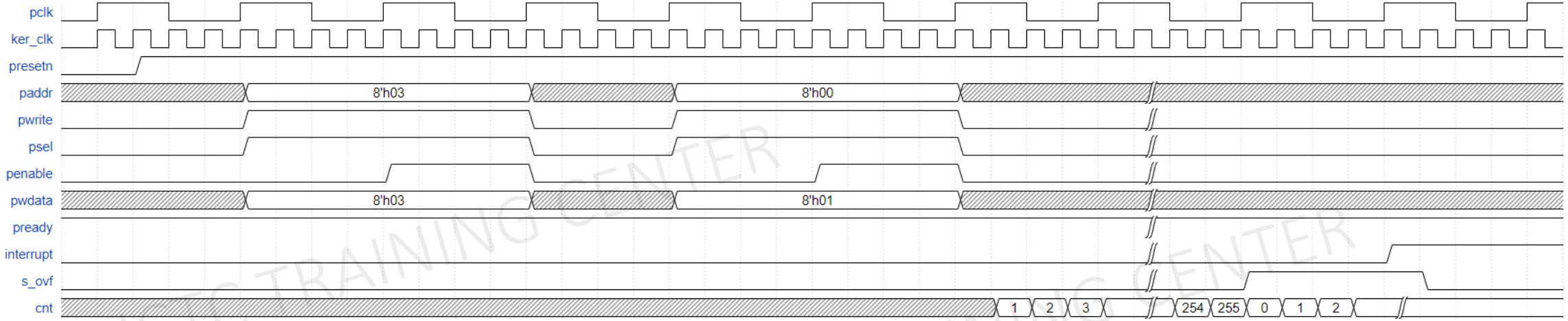
0x03: Timer Interrupt Enable Register (TIE)

Bit	Name	Type	Default value	Description
7:2	-	-	Rsvd	
1	underflow_en	R/W	1'b0	Underflow Interrupt Enable This bit is allow interrupt signal is trigger when underflow status tri ggered, Otherwise, the interrupt signal will not.
0	overflow_en	R/W	1'b0	Overflow Interrupt Enable This bit is allow interrupt signal is trigger when overflow status trig gered, Otherwise, the interrupt signal will not.

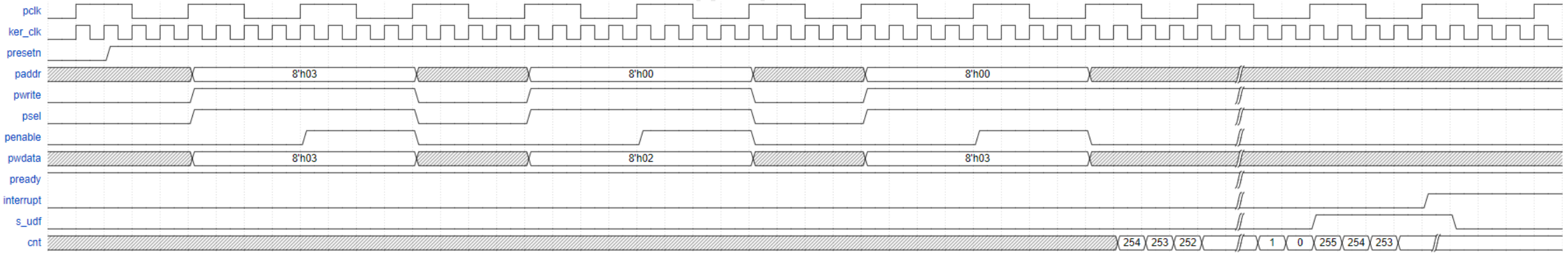
Project 1: 8-bit timer

Waveform

❑ Count up with interrupt waveform (No clock divisor):

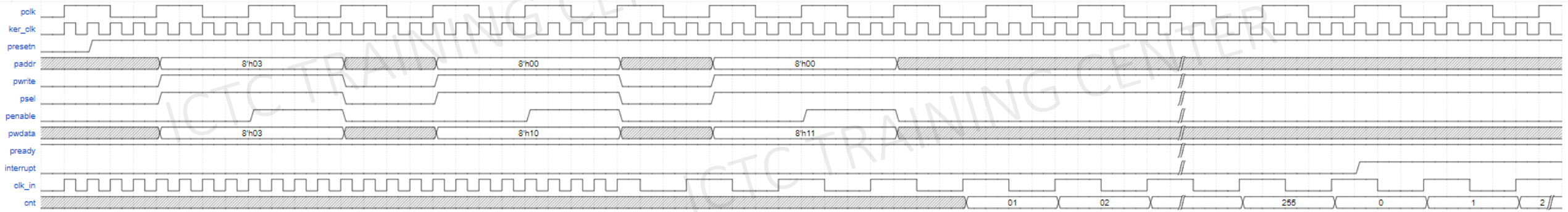


❑ Count down with interrupt waveform (No clock divisor):



Project 1: 8-bit timer

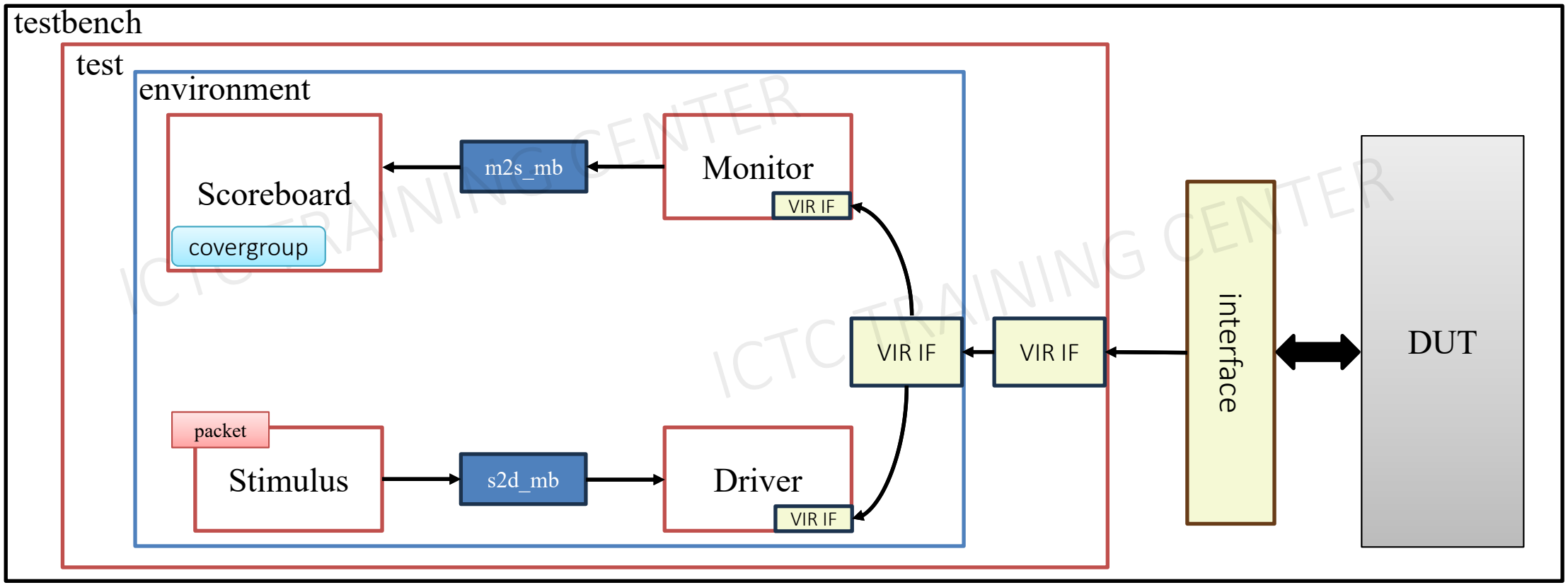
Waveform



Count up with Clock divide by 4

Project 1: 8-bit timer

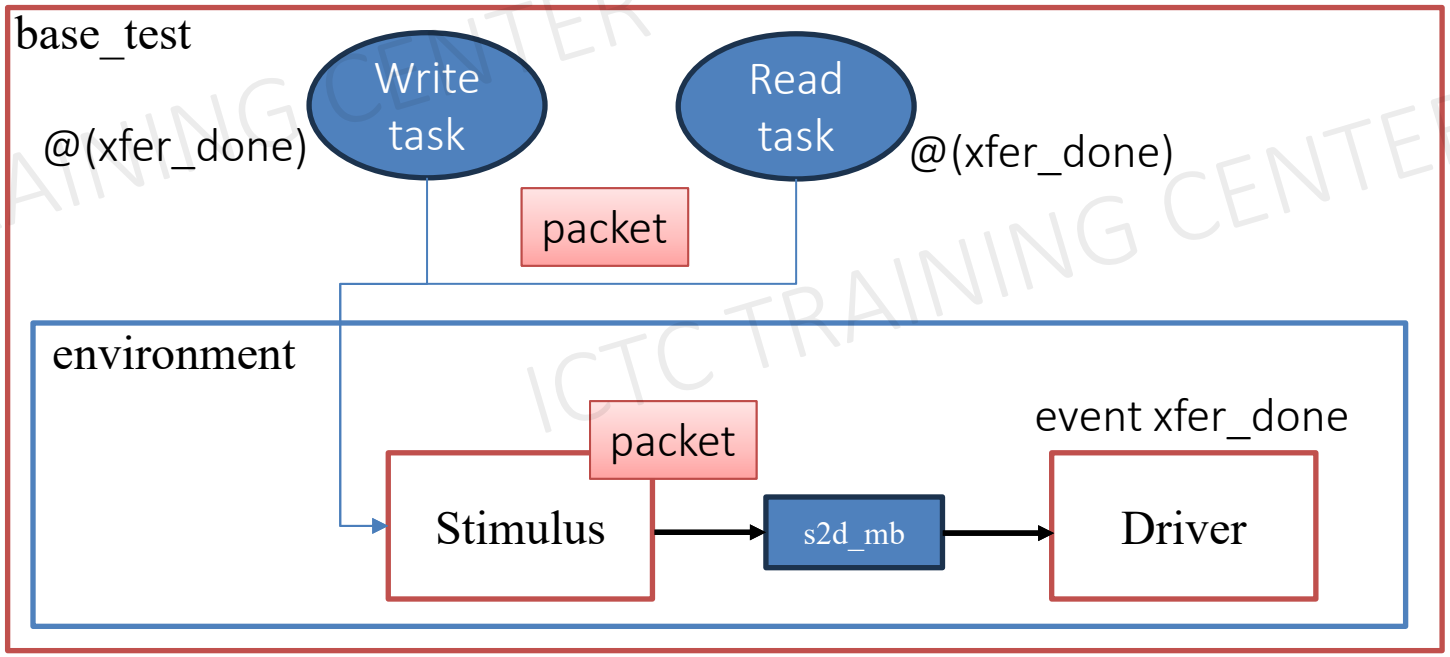
Testbench structure



Project 1: 8-bit timer

Handshake in testbench

Guiding handshake between test -> stimulus -> driver



Project 1: 8-bit timer

Project Requirement

Instructions

1. Copy /ictc/student_data/share/dv_advanced/project1 to your working directory
2. Self estimate your schedule.
3. Investiage RTL specification (from slide 182 to slide 190)
4. Create Verification plan (Vplan). Refer Vpan template: Verification Plan Template
5. Do verification.
   - Build up testbench by using systemverilog. Refer testbench structure in slide 191
   - Create testcase and checker according to Vplan to verify RTL

   Extra: Implement functional coverage