FPGA 开源工作室将通过五篇文章来给大家讲解 **xilinx FPGA** 使用 **mig IP** 对 **DDR3** 的读写控制，旨在让大家更快的学习和应用 **DDR3**。

本实验和工程基于 **Digilent** 的 **Arty Artix-35T FPGA** 开发板完成。

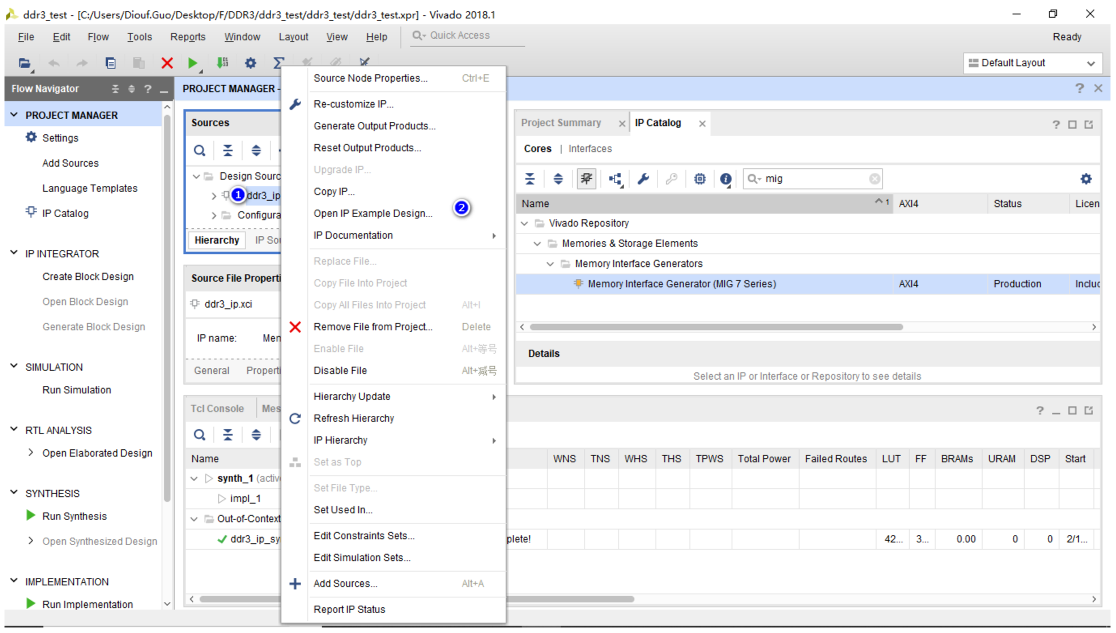
软件使用 **Vivado 2018.1**。

参考工程：**ddr3_sim**。

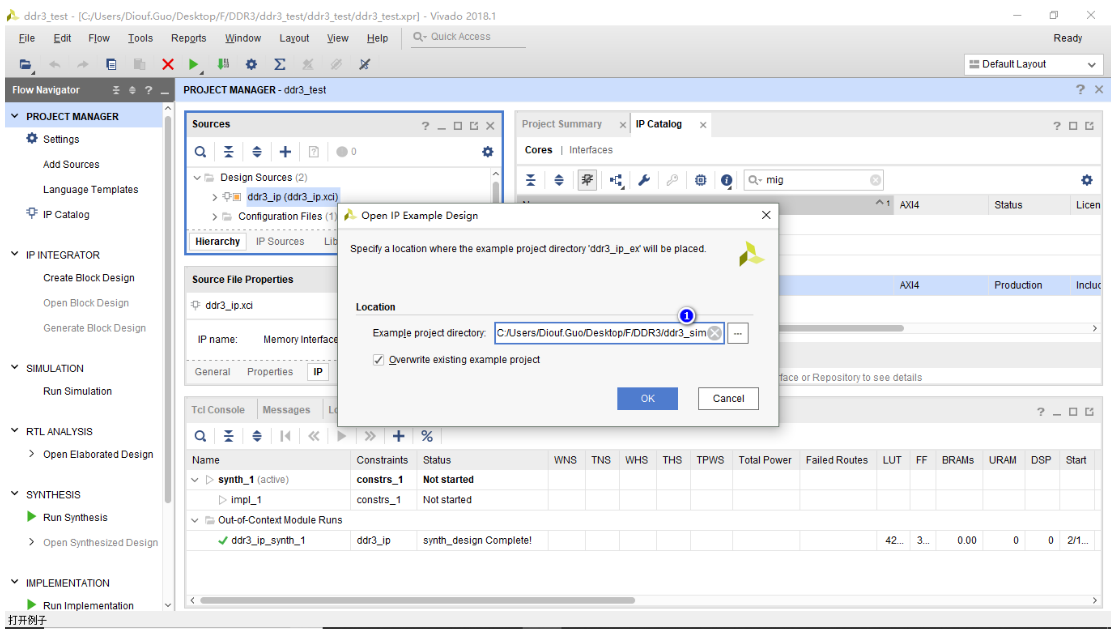
# 第四篇：mig IP 的仿真

## 1 快速仿真

### 1>①ddr3_ip->②Open IP Example Design



### 2>选择 ddr3 仿真生成的路径。



Open IP Example Design

Specify a location where the example project directory 'ddr3_ip_ex' will be placed.

Location

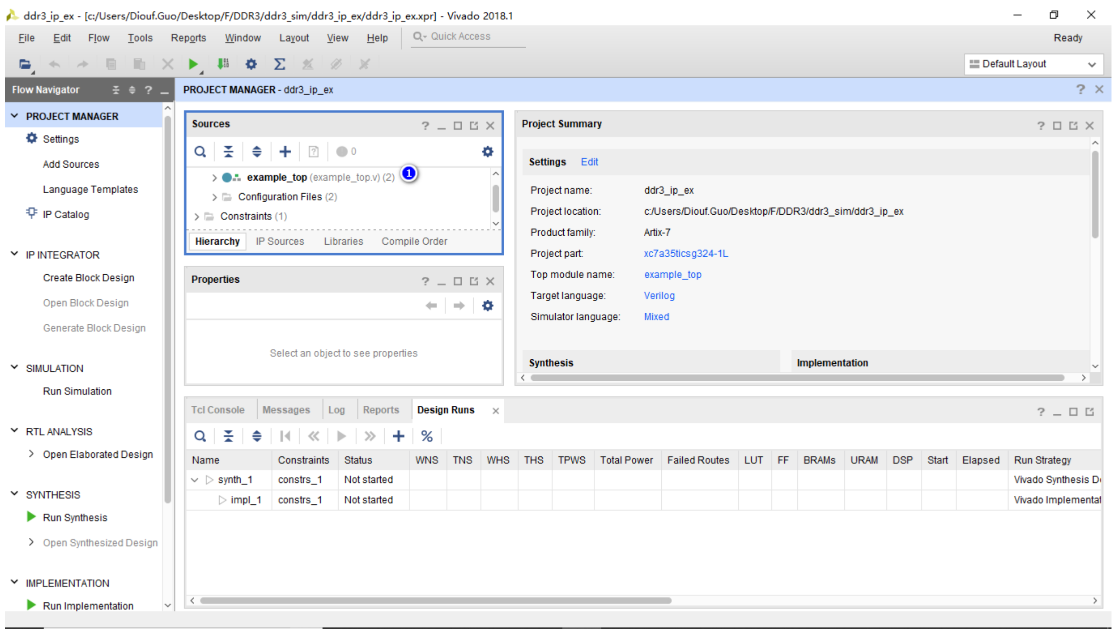
Example project directory: C:/Users/Diouf.Guo/Desktop/F/DDR3/ddr3_sim

☑ Overwrite existing example project

OK Cancel

| Name | Constraints | Status | WNS | TNS | WHS | THS | TPWS | Total Power | Failed Routes | LUT | FF | BRAMs | URAM | DSP | Start |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| synth_1 (active) | constrs_1 | Not started | | | | | | | | | | | | | |
| impl_1 | constrs_1 | Not started | | | | | | | | | | | | | |
| Out-of-Context Module Runs | | | | | | | | | | | | | | | |
| ddr3_ip_synth_1 | ddr3_ip | synth_design Complete! | | | | | | | | 42... | 3... | 0.00 | 0 | 0 | 2/1... |

## 3>DDR3 自带仿真工程生成完毕。



Project Summary

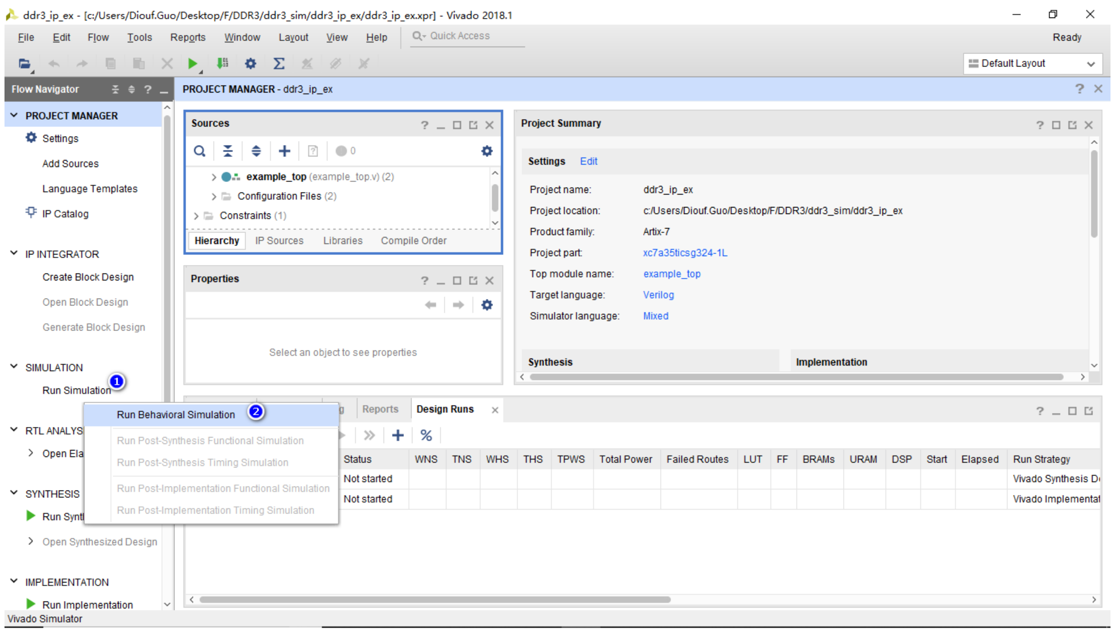
Settings Edit

| | |
|---|---|
| Project name: | ddr3_ip_ex |
| Project location: | c:/Users/Diouf.Guo/Desktop/F/DDR3/ddr3_sim/ddr3_ip_ex |
| Product family: | Artix-7 |
| Project part: | xc7a35ticsg324-1L |
| Top module name: | example_top |
| Target language: | Verilog |
| Simulator language: | Mixed |

Synthesis Implementation

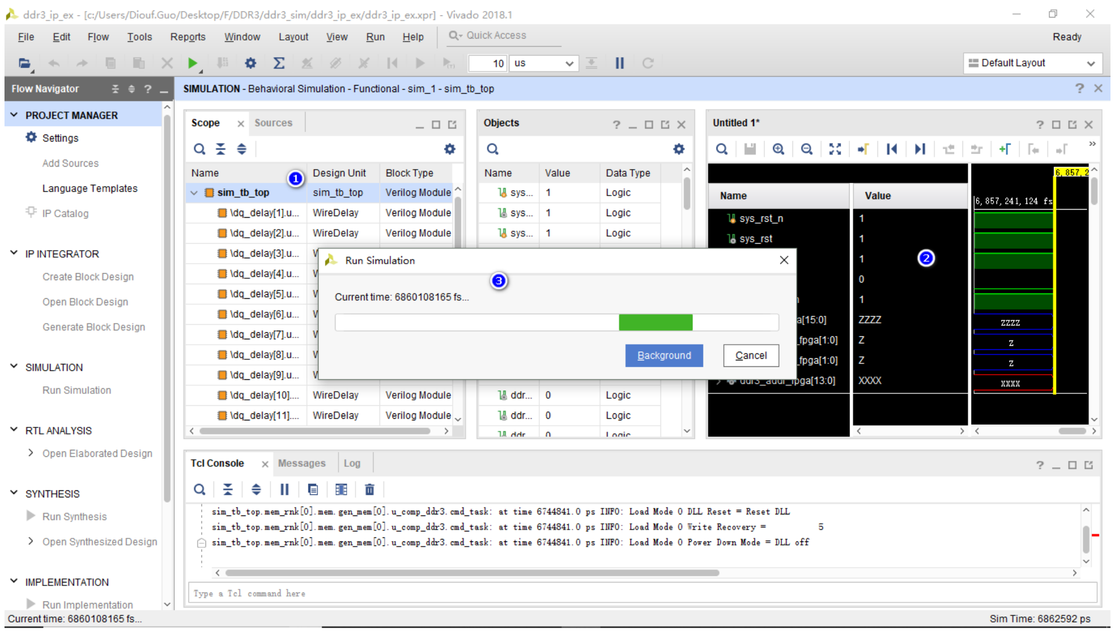
| Name | Constraints | Status | WNS | TNS | WHS | THS | TPWS | Total Power | Failed Routes | LUT | FF | BRAMs | URAM | DSP | Start | Elapsed | Run Strategy |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| synth_1 | constrs_1 | Not started | | | | | | | | | | | | | | | Vivado Synthesis D |
| impl_1 | constrs_1 | Not started | | | | | | | | | | | | | | | Vivado Implementat |

## 4>①Run Simulation ->②Run Behavioral Simulation。

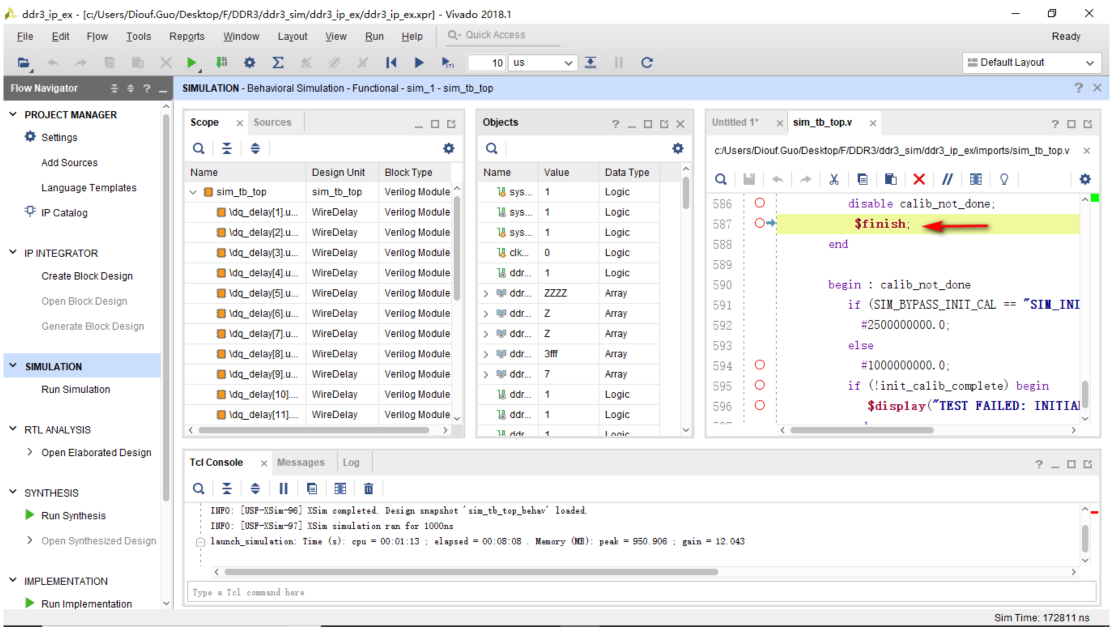


Project name: ddr3_ip_ex
Project location: c:/Users/Diouf.Guo/Desktop/F/DDR3/ddr3_sim/ddr3_ip_ex
Product family: Artix-7
Project part: xc7a35ticsg324-1L
Top module name: example_top
Target language: Verilog
Simulator language: Mixed

**5>等待 10 几分钟左右仿真完成。**


ddr3_ip_ex - [c:/Users/Diouf.Guo/Desktop/F/DDR3/ddr3_sim/ddr3_ip_ex/ddr3_ip_ex.xpr] - Vivado 2018.1
File Edit Flow Tools Reports Window Layout View Run Help
Quick Access
Ready
10 us
Default Layout
SIMULATION - Behavioral Simulation - Functional - sim_1 - sim_tb_top
Scope
Sources
Name
Design Unit
Block Type
sim_tb_top
1
\dq_delay[1].u...
WireDelay
Verilog Module
Objects
Value
Data Type
Logic
Untitled 1*
sys_rst_n
sys_rst
2
ZZZZ
XXXX
Run Simulation
3
Current time: 6860108165 fs...
Background
Cancel
Tcl Console
Messages
Log
Current time: 6860108165 fs...
Sim Time: 6862592 ps

```
sim_tb_top.mem_rnk[0].mem.gen_mem[0].u_comp_ddr3.cmd_task: at time 6744841.0 ps INFO: Load Mode 0 DLL Reset = Reset DLL
sim_tb_top.mem_rnk[0].mem.gen_mem[0].u_comp_ddr3.cmd_task: at time 6744841.0 ps INFO: Load Mode 0 Write Recovery =          5
sim_tb_top.mem_rnk[0].mem.gen_mem[0].u_comp_ddr3.cmd_task: at time 6744841.0 ps INFO: Load Mode 0 Power Down Mode = DLL off
```



ddr3_ip_ex - [c:/Users/Diouf.Guo/Desktop/F/DDR3/ddr3_sim/ddr3_ip_ex/ddr3_ip_ex.xpr] - Vivado 2018.1

SIMULATION - Behavioral Simulation - Functional - sim_1 - sim_tb_top

| Name | Design Unit | Block Type |
|---|---|---|
| sim_tb_top | sim_tb_top | Verilog Module |
| \dq_delay[1].u... | WireDelay | Verilog Module |
| \dq_delay[2].u... | WireDelay | Verilog Module |
| \dq_delay[3].u... | WireDelay | Verilog Module |
| \dq_delay[4].u... | WireDelay | Verilog Module |
| \dq_delay[5].u... | WireDelay | Verilog Module |
| \dq_delay[6].u... | WireDelay | Verilog Module |
| \dq_delay[7].u... | WireDelay | Verilog Module |
| \dq_delay[8].u... | WireDelay | Verilog Module |
| \dq_delay[9].u... | WireDelay | Verilog Module |
| \dq_delay[10].... | WireDelay | Verilog Module |
| \dq_delay[11].... | WireDelay | Verilog Module |

| Name | Value | Data Type |
|---|---|---|
| sys... | 1 | Logic |
| sys... | 1 | Logic |
| sys... | 1 | Logic |
| clk... | 0 | Logic |
| ddr... | 1 | Logic |
| ddr... | ZZZZ | Array |
| ddr... | Z | Array |
| ddr... | Z | Array |
| ddr... | 3ff | Array |
| ddr... | 7 | Array |
| ddr... | 1 | Logic |
| ddr... | 1 | Logic |

c:/Users/Diouf.Guo/Desktop/F/DDR3/ddr3_sim/ddr3_ip_ex/imports/sim_tb_top.v

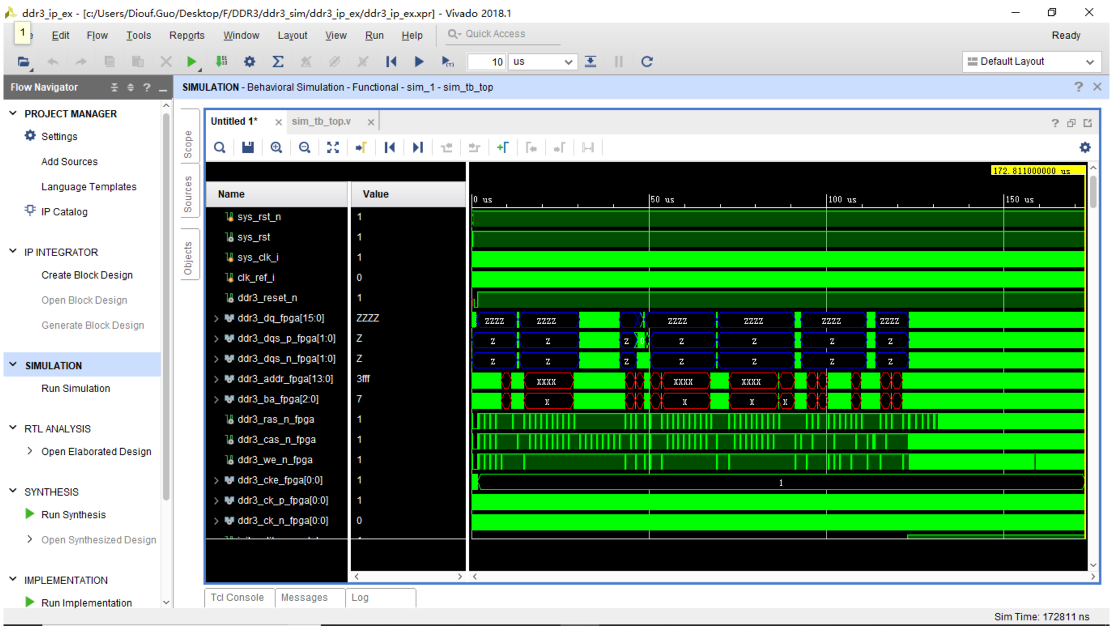
```
               disable calib_not_done;
                $finish;
           end

           begin : calib_not_done
               if (SIM_BYPASS_INIT_CAL == "SIM_INI
                 #2500000000.0;
               else
                 #1000000000.0;
               if (!init_calib_complete) begin
                 $display("TEST FAILED: INITIA
```

```
INFO: [USF-XSim-96] XSim completed. Design snapshot 'sim_tb_top_behav' loaded.
INFO: [USF-XSim-97] XSim simulation ran for 1000ns
launch_simulation: Time (s): cpu = 00:01:13 ; elapsed = 00:08:08 . Memory (MB): peak = 950.906 ; gain = 12.043
```

Sim Time: 172811 ns

**6>**仿真完成查看波形。

对于 **xiinx** 官方自带的 **DDR3** 仿真的例子大家可以参看 **UG586**（**https://china.xilinx.com/support/documentation/ip_documentation/mig_7series/v4_2/ug586_7Series_MIS.pdf**）。



ddr3_ip_ex - [c:/Users/Diouf.Guo/Desktop/F/DDR3/ddr3_sim/ddr3_ip_ex/ddr3_ip_ex.xpr] - Vivado 2018.1

SIMULATION - Behavioral Simulation - Functional - sim_1 - sim_tb_top

| Name | Value |
|---|---|
| sys_rst_n | 1 |
| sys_rst | 1 |
| sys_clk_i | 1 |
| clk_ref_i | 0 |
| ddr3_reset_n | 1 |
| ddr3_dq_fpga[15:0] | ZZZZ |
| ddr3_dqs_p_fpga[1:0] | Z |
| ddr3_dqs_n_fpga[1:0] | Z |
| ddr3_addr_fpga[13:0] | 3ff |
| ddr3_ba_fpga[2:0] | 7 |
| ddr3_ras_n_fpga | 1 |
| ddr3_cas_n_fpga | 1 |
| ddr3_we_n_fpga | 1 |
| ddr3_cke_fpga[0:0] | 1 |
| ddr3_ck_p_fpga[0:0] | 1 |
| ddr3_ck_n_fpga[0:0] | 0 |

Sim Time: 172811 ns

## 2 仿真

目标：对 **DDR3** 的 **8** 个 **bank** 从 **0 bank** 开始对每个 **bank** 写入 **0-99**，然后依次读出，循环读写。

修改 **example_top** 模块如下所示：

```
`timescale 1ps/1ps
`define CMD_WR 3'b000
`define CMD_RD 3'b001
module example_top
  (

    // Inouts
    inout [15:0]                ddr3_dq,
    inout [1:0]               ddr3_dqs_n,
    inout [1:0]               ddr3_dqs_p,

    // Outputs
    output [13:0]              ddr3_addr,
    output [2:0]             ddr3_ba,
    output                          ddr3_ras_n,
    output                          ddr3_cas_n,
    output                          ddr3_we_n,
    output                          ddr3_reset_n,
    output [0:0]               ddr3_ck_p,
    output [0:0]               ddr3_ck_n,
    output [0:0]              ddr3_cke,

output [0:0]        ddr3_cs_n,

output [1:0]                ddr3_dm,

output [0:0]                ddr3_odt,


// Inputs

// Single-ended system clock
input                               sys_clk_i,

// Single-ended iodelayctrl clk (reference clock)
input                               clk_ref_i,

output                                tg_compare_error,
output                                init_calib_complete,



// System reset - Default polarity of sys_rst pin is Active Low.
// System reset polarity will change based on the option
// selected in GUI.
input                           sys_rst
);

//*************************************************************************
**
parameter IDLE  = 5'd0,
        WR1  = 5'd1,
        WR2  = 5'd2,
        WR3  = 5'd3,
        WR4  = 5'd4,
        WR5  = 5'd5,
        WR6  = 5'd6,
        WR7  = 5'd7,
        WR8  = 5'd8,
        RD1  = 5'd9,
        RD2  = 5'd10,
        RD3  = 5'd11,
        RD4  = 5'd12,
        RD5  = 5'd13,
        RD6  = 5'd14,
        RD7  = 5'd15,
    RD8  = 5'd16;

// user interface signals
wire [27:0]     app_addr;//i
reg [2:0]       app_cmd;//i
wire            app_en;//i
reg [127:0]     app_wdf_data;//i
wire            app_wdf_end;//i
wire [15:0]     app_wdf_mask;//i
wire           app_wdf_wren;//i
wire [127:0]     app_rd_data;
wire            app_rd_data_end;
wire            app_rd_data_valid;
wire            app_rdy;
wire            app_wdf_rdy;
wire            app_sr_req;//i
wire            app_ref_req;//i
wire            app_zq_req;//i
wire            app_sr_active;
wire            app_ref_ack;
wire            app_zq_ack;
wire            ui_clk;
wire          ui_clk_sync_rst;
//wire          init_calib_complete;
wire          sys_rst_n;//i

reg [4:0] cstate,nstate;
//wire [27:0]     wr_addr;//i  bank row column [2:0] [13:0] [9:0]
//wire [27:0]     rd_addr;//i bank row column [2:0] [13:0] [9:0]
wire wr1_done;
wire wr2_done;
wire wr3_done;
wire wr4_done;
wire wr5_done;
wire wr6_done;
wire wr7_done;
wire wr8_done;

wire rd1_done;
wire rd2_done;
wire rd3_done;
wire rd4_done;
wire rd5_done;
wire rd6_done;
wire rd7_done;
wire rd8_done;

reg [2:0] bank;

reg [23:0] addr;
assign app_addr={1'b0,bank,addr};
assign  app_sr_req = 1'b0;
assign app_ref_req = 1'b0;
assign app_zq_req  = 1'b0;
//assign app_addr = (app_cmd ==`CMD_WR && app_en == 1'b1)?wr_addr:rd_addr;
assign app_wdf_mask = 16'h0000;
assign app_wdf_end = app_wdf_wren;
assign wr1_done = (app_cmd == `CMD_WR && addr == 28'd800 && bank == 3'b000) ? 1'b1:1'b0;
assign wr2_done = (app_cmd == `CMD_WR && addr == 28'd800 && bank == 3'b001) ? 1'b1:1'b0;
assign wr3_done = (app_cmd == `CMD_WR && addr == 28'd800 && bank == 3'b010) ? 1'b1:1'b0;
assign wr4_done = (app_cmd == `CMD_WR && addr == 28'd800 && bank == 3'b011) ? 1'b1:1'b0;
assign wr5_done = (app_cmd ==`CMD_WR && addr == 28'd800 && bank == 3'b100) ? 1'b1:1'b0;
assign wr6_done = (app_cmd ==`CMD_WR && addr == 28'd800 && bank == 3'b101) ? 1'b1:1'b0;
assign wr7_done = (app_cmd ==`CMD_WR && addr == 28'd800 && bank == 3'b110) ? 1'b1:1'b0;
assign wr8_done = (app_cmd ==`CMD_WR && addr == 28'd800 && bank == 3'b111) ? 1'b1:1'b0;

assign rd1_done = (cstate == RD1 && app_cmd ==`CMD_RD && addr == 28'd800) ? 1'b1:1'b0;
assign rd2_done = (cstate == RD2 && app_cmd ==`CMD_RD && addr == 28'd800) ? 1'b1:1'b0;
assign rd3_done = (cstate == RD3 && app_cmd ==`CMD_RD && addr == 28'd800) ? 1'b1:1'b0;
assign rd4_done = (cstate == RD4 && app_cmd ==`CMD_RD && addr == 28'd800) ? 1'b1:1'b0;
assign rd5_done = (cstate == RD5 && app_cmd ==`CMD_RD && addr == 28'd800) ? 1'b1:1'b0;
assign rd6_done = (cstate == RD6 && app_cmd ==`CMD_RD && addr == 28'd800) ? 1'b1:1'b0;
assign rd7_done = (cstate == RD7 && app_cmd ==`CMD_RD && addr == 28'd800) ? 1'b1:1'b0;
assign rd8_done = (cstate == RD8 && app_cmd ==`CMD_RD && addr == 28'd800) ? 1'b1:1'b0;

assign done_flag= (cstate == DONE)?1'b1:1'b0;

assign app_en =(((cstate == WR1 ||cstate == WR2 ||cstate == WR3 ||cstate == WR4 ||cstate == WR5 ||cstate == WR6 ||cstate == WR7 ||cstate == WR8)&& app_wdf_rdy == 1'b1&&app_rdy == 1'b1)||((cstate == RD1 ||cstate == RD2 ||cstate == RD3 ||cstate == RD4 ||cstate == RD5 ||cstate == RD6 ||cstate == RD7 ||cstate == RD8)&& app_rdy == 1'b1)&&((!wr1_done)||(!wr2_done)||(!wr3_done)||(!wr4_done)||(!wr5_done)||(!wr6_done)||(!wr7_done)||(!wr8_done)||(!rd1_done)||(!rd2_done)||(!rd3_done)||(!rd4_done)||(!rd5_done)||(!rd6_done)||(!rd7_done)||(!rd8_done))?1'b1:1'b0;
assign app_wdf_wren =(((cstate == WR1 ||cstate == WR2 ||cstate == WR3 ||cstate == WR4 ||cstate == WR5 ||cstate == WR6 ||cstate == WR7 ||cstate == WR8)&& app_wdf_rdy == 1'b1&&app_rdy == 1'b1)&&((!wr1_done)||(!wr2_done)||(!wr3_done)||(!wr4_done)||(!wr5_done)||(!wr6_done)||(!wr7_done)||(!wr8_done)))?1'b1:1'b0;
always @(posedge ui_clk or posedge ui_clk_sync_rst) begin
  if(ui_clk_sync_rst == 1'b1)
```

```
    cstate <= IDLE;
  else
    cstate <= nstate;
end

always @(*) begin
  nstate = IDLE;
  case(cstate)
    IDLE:begin
      if(init_calib_complete == 1'b1)
        nstate = WR1;
      else
        nstate = IDLE;
    end
    WR1:begin
      if(wr1_done == 1'b1)
        nstate = WR2;
      else
        nstate = WR1;
    end
    WR2:begin
      if(wr2_done == 1'b1)
    nstate = WR3;
  else
    nstate = WR2;
end
WR3:begin
  if(wr3_done == 1'b1)
    nstate = WR4;
  else
    nstate = WR3;
end
WR4:begin
  if(wr4_done == 1'b1)
    nstate = WR5;
  else
    nstate = WR4;
end
WR5:begin
  if(wr5_done == 1'b1)
    nstate = WR6;
  else
    nstate = WR5;
end
```

```
WR6:begin
  if(wr6_done == 1'b1)
    nstate = WR7;
  else
    nstate = WR6;
end
WR7:begin
  if(wr7_done == 1'b1)
    nstate = WR8;
  else
    nstate = WR7;
end
WR8:begin
  if(wr8_done == 1'b1)
    nstate = RD1;
  else
    nstate = WR8;
end
RD1:begin
  if(rd1_done == 1'b1)
    nstate = RD2;
  else
```

```
      nstate = RD1;
    end
    RD2:begin
     if(rd2_done == 1'b1)
      nstate = RD3;
     else
      nstate = RD2;
    end
    RD3:begin
      if(rd3_done == 1'b1)
       nstate = RD4;
      else
       nstate = RD3;
    end
    RD4:begin
      if(rd4_done == 1'b1)
       nstate = RD5;
      else
       nstate = RD4;
    end
    RD5:begin
      if(rd5_done == 1'b1)
    nstate = RD6;
  else
    nstate = RD5;
end
RD6:begin
  if(rd6_done == 1'b1)
    nstate = RD7;
  else
    nstate = RD6;
end
RD7:begin
  if(rd7_done == 1'b1)
    nstate = RD8;
  else
    nstate = RD7;
end
RD8:begin
  if(rd8_done == 1'b1)
    nstate = WR1;
  else
    nstate = RD8;
end
```

```
 endcase
end

always @(posedge ui_clk or posedge ui_clk_sync_rst) begin
 if(ui_clk_sync_rst == 1'b1) begin
  app_cmd <= `CMD_WR;
  app_wdf_data <= 128'b0;

  bank <= 3'b000;
  addr <=24'b0;
 end
 else
  case(cstate)
   WR1:begin
    app_cmd <= `CMD_WR;
    bank <= 3'b000;
    if(wr1_done == 1'b1) begin
     addr <=24'b0;
     app_wdf_data <= 128'b0;
    end
    else if(app_wdf_rdy== 1'b1&&app_rdy== 1'b1) begin
     addr <= addr+8;
     app_wdf_data <= app_wdf_data + 1;
    end
    else begin
     addr <= addr;
     app_wdf_data <= app_wdf_data;
    end
   end
   WR2:begin
    app_cmd <= `CMD_WR;
    bank <= 3'b001;
    if(wr2_done == 1'b1) begin
     addr <=24'b0;
     app_wdf_data <= 128'b0;
    end
    else if(app_wdf_rdy== 1'b1&&app_rdy== 1'b1) begin
     addr <= addr+8;
     app_wdf_data <= app_wdf_data + 1;
    end
    else begin
     addr <= addr;
     app_wdf_data <= app_wdf_data;
    end
```

```
end
WR3:begin
  app_cmd <= `CMD_WR;
  bank <= 3'b010;
  if(wr3_done == 1'b1) begin
    addr <=24'b0;
    app_wdf_data <= 128'b0;
  end
  else if(app_wdf_rdy== 1'b1&&app_rdy== 1'b1) begin
    addr <= addr+8;
    app_wdf_data <= app_wdf_data +1;
  end
  else begin
    addr <= addr;
    app_wdf_data <= app_wdf_data;
  end
end
WR4:begin
  app_cmd <= `CMD_WR;
  bank <= 3'b011;
  if(wr4_done == 1'b1) begin
    addr <=24'b0;
```

```
     app_wdf_data <= 128'b0;
    end
    else if(app_wdf_rdy== 1'b1&&app_rdy== 1'b1) begin
     addr <= addr+8;
     app_wdf_data <= app_wdf_data +1;
    end
    else begin
     addr <= addr;
     app_wdf_data <= app_wdf_data;
    end
   end
   WR5:begin
    app_cmd <= `CMD_WR;
    bank <= 3'b100;
    if(wr5_done == 1'b1) begin
     addr <=24'b0;
     app_wdf_data <= 128'b0;
    end
    else if(app_wdf_rdy== 1'b1&&app_rdy== 1'b1) begin
     addr <= addr+8;
     app_wdf_data <= app_wdf_data +1;
   end
   else begin
    addr <= addr;
    app_wdf_data <= app_wdf_data;
   end
  end
  WR6:begin
   app_cmd <= `CMD_WR;
   bank <= 3'b101;
  if(wr6_done == 1'b1) begin
    addr <=24'b0;
    app_wdf_data <= 128'b0;
   end
   else if(app_wdf_rdy== 1'b1&&app_rdy== 1'b1) begin

    addr <= addr+8;
    app_wdf_data <= app_wdf_data +1;
   end
   else begin
    addr <= addr;
    app_wdf_data <= app_wdf_data;
   end
```

```
end
WR7:begin
 app_cmd <= `CMD_WR;
 bank <= 3'b110;
 if(wr7_done == 1'b1) begin
  addr <=24'b0;
  app_wdf_data <= 128'b0;
 end
 else if(app_wdf_rdy== 1'b1&&app_rdy== 1'b1) begin

  addr <= addr+8;
  app_wdf_data <= app_wdf_data +1;
 end
 else begin
  addr <= addr;
  app_wdf_data <= app_wdf_data;
 end
end
WR8:begin
 app_cmd <= `CMD_WR;
 bank <= 3'b111;
 if(wr8_done == 1'b1) begin
```

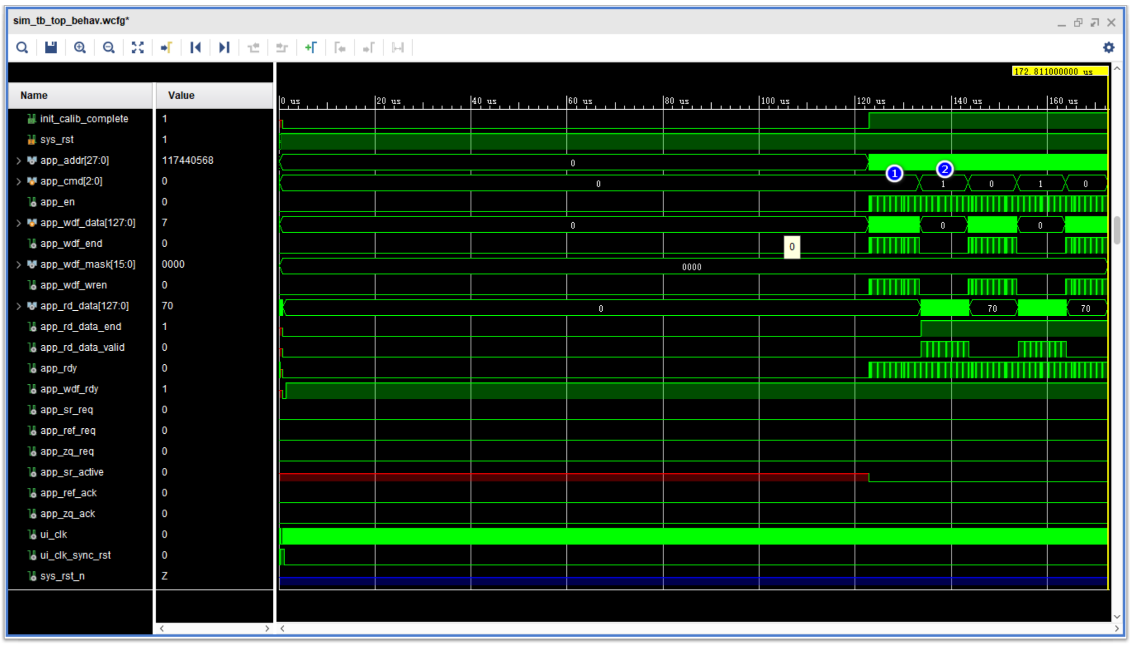
```
     addr <=24'b0;
     app_wdf_data <= 128'b0;
    end
    else if(app_wdf_rdy== 1'b1&&app_rdy== 1'b1) begin
     addr <= addr+8;
     app_wdf_data <= app_wdf_data +1;
    end
    else begin
     addr <= addr;
     app_wdf_data <= app_wdf_data;
    end
   end
   RD1:begin
    app_cmd <= `CMD_RD;
    bank <= 3'b000;
    if(rd1_done == 1'b1) begin
     addr <= 24'b0;
    end
    else if(app_rdy== 1'b1)begin
      addr <= addr+8;
    end
    else begin
    addr<=addr;
   end
  end
  RD2:begin
   app_cmd<=`CMD_RD;
   bank<=3'b001;
   if(rd2_done==1'b1) begin
    addr<=24'b0;
   end
   else if(app_rdy==1'b1)begin
     addr<=addr+8;
   end
   else begin
    addr<=addr;
   end
  end
  RD3:begin
   app_cmd<=`CMD_RD;
   bank<=3'b010;
   if(rd3_done==1'b1) begin
    addr<=24'b0;
   end
```

```
    else if(app_rdy==1'b1)begin
      addr<=addr+8;
    end
    else begin
     addr<=addr;
    end
  end
  RD4:begin
    app_cmd<=`CMD_RD;
    bank<=3'b011;
    if(rd4_done==1'b1) begin
     addr<=24'b0;
    end
    else if(app_rdy==1'b1)begin
      addr<=addr+8;
    end
    else begin
     addr<=addr;
    end
  end
  RD5:begin
    app_cmd<=`CMD_RD;
    bank<=3'b100;
    if(rd5_done==1'b1) begin
     addr<=24'b0;
    end
    else if(app_rdy==1'b1)begin
      addr<=addr+8;
    end
    else begin
     addr<=addr;
    end
   end
   RD6:begin
    app_cmd<=`CMD_RD;
    bank<=3'b101;
    if(rd6_done==1'b1) begin
     addr<=24'b0;
    end
    else if(app_rdy==1'b1)begin
      addr<=addr+8;
    end
    else begin
     addr<=addr;
```

```
  end
 end
 RD7:begin
  app_cmd <= `CMD_RD;
  bank <= 3'b110;
  if(rd7_done == 1'b1) begin
   addr <= 24'b0;
  end
  else if(app_rdy == 1'b1)begin
   addr <= addr+8;
  end
  else begin
   addr <= addr;
  end
 end
 RD8:begin
  app_cmd <= `CMD_RD;
  bank <= 3'b111;
  if(rd8_done == 1'b1) begin
   addr <= 24'b0;
  end
  else if(app_rdy == 1'b1)begin
```

```
         addr <= addr+8;
       end
       else begin
         addr <= addr;
       end
     end
     DONE:begin
       bank <= 3'b000;
       addr<=24'b0;
       app_cmd <= `CMD_WR;
       app_wdf_data <= 128'b0;
     end
   endcase
end



//****************************************************************************


// Start of User Design top instance
```

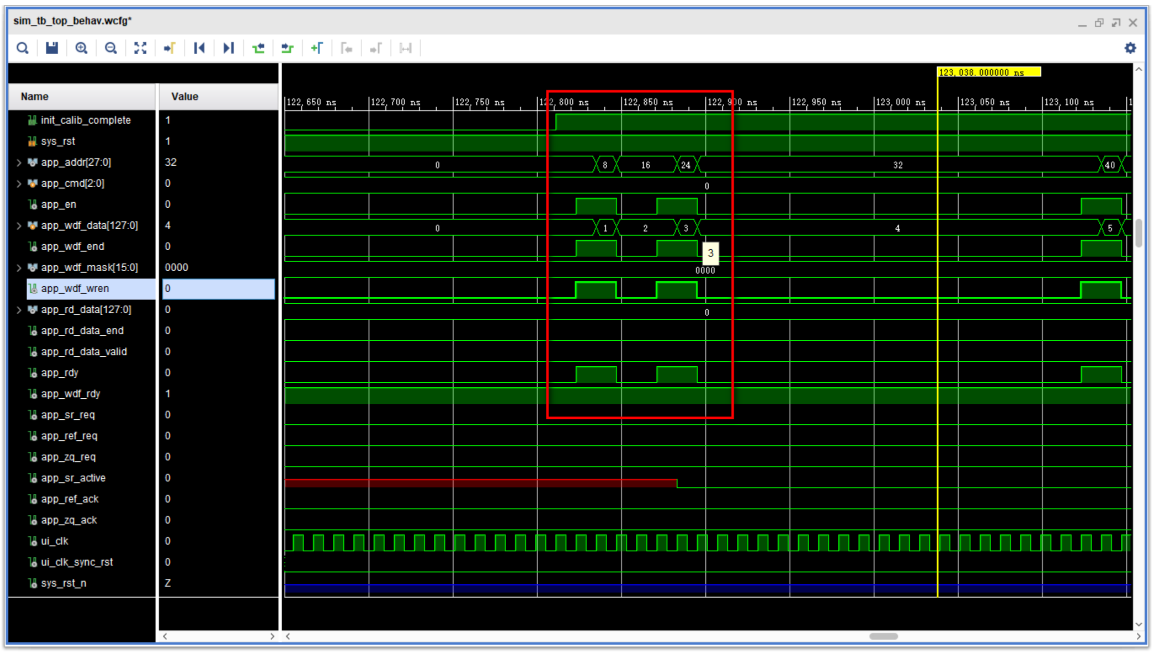
```
//*************************************************************************
// The User design is instantiated below. The memory interface ports are
// connected to the top-level and the application interface ports are
// connected to the traffic generator module. This provides a reference
// for connecting the memory controller to system.
//*************************************************************************

 ddr3_ip u_ddr3_ip
   (


// Memory interface ports
    .ddr3_addr                (ddr3_addr),
    .ddr3_ba                  (ddr3_ba),
    .ddr3_cas_n               (ddr3_cas_n),
    .ddr3_ck_n                (ddr3_ck_n),
    .ddr3_ck_p                (ddr3_ck_p),
    .ddr3_cke                 (ddr3_cke),
    .ddr3_ras_n               (ddr3_ras_n),
    .ddr3_we_n                (ddr3_we_n),
.ddr3_dq              (ddr3_dq),
.ddr3_dqs_n           (ddr3_dqs_n),
.ddr3_dqs_p           (ddr3_dqs_p),
.ddr3_reset_n         (ddr3_reset_n),
.init_calib_complete  (init_calib_complete),

.ddr3_cs_n            (ddr3_cs_n),
.ddr3_dm              (ddr3_dm),
.ddr3_odt             (ddr3_odt),
// Application interface ports
.app_addr             (app_addr),
.app_cmd              (app_cmd),
.app_en               (app_en),
.app_wdf_data         (app_wdf_data),
.app_wdf_end          (app_wdf_end),
.app_wdf_wren         (app_wdf_wren),
.app_rd_data          (app_rd_data),
.app_rd_data_end      (app_rd_data_end),
.app_rd_data_valid    (app_rd_data_valid),
.app_rdy              (app_rdy),
.app_wdf_rdy          (app_wdf_rdy),
.app_sr_req           (app_sr_req),
    .app_ref_req              (app_ref_req),
    .app_zq_req               (app_zq_req),
    .app_sr_active            (app_sr_active),
    .app_ref_ack              (app_ref_ack),
    .app_zq_ack               (app_zq_ack),
    .ui_clk                   (ui_clk),
    .ui_clk_sync_rst          (ui_clk_sync_rst),

    .app_wdf_mask             (app_wdf_mask),


// System Clock Ports
    .sys_clk_i                (sys_clk_i),
// Reference Clock Ports
    .clk_ref_i                (clk_ref_i),

    .sys_rst                  (sys_rst)
    );
// End of User Design top instance


endmodule
```



sim_tb_top_behav.wcfg*

| Name | Value |
|---|---|
| init_calib_complete | 1 |
| sys_rst | 1 |
| app_addr[27:0] | 117440568 |
| app_cmd[2:0] | 0 |
| app_en | 0 |
| app_wdf_data[127:0] | 7 |
| app_wdf_end | 0 |
| app_wdf_mask[15:0] | 0000 |
| app_wdf_wren | 0 |
| app_rd_data[127:0] | 70 |
| app_rd_data_end | 1 |
| app_rd_data_valid | 0 |
| app_rdy | 0 |
| app_wdf_rdy | 1 |
| app_sr_req | 0 |
| app_ref_req | 0 |
| app_zq_req | 0 |
| app_sr_active | 0 |
| app_ref_ack | 0 |
| app_zq_ack | 0 |
| ui_clk | 0 |
| ui_clk_sync_rst | 0 |
| sys_rst_n | Z |

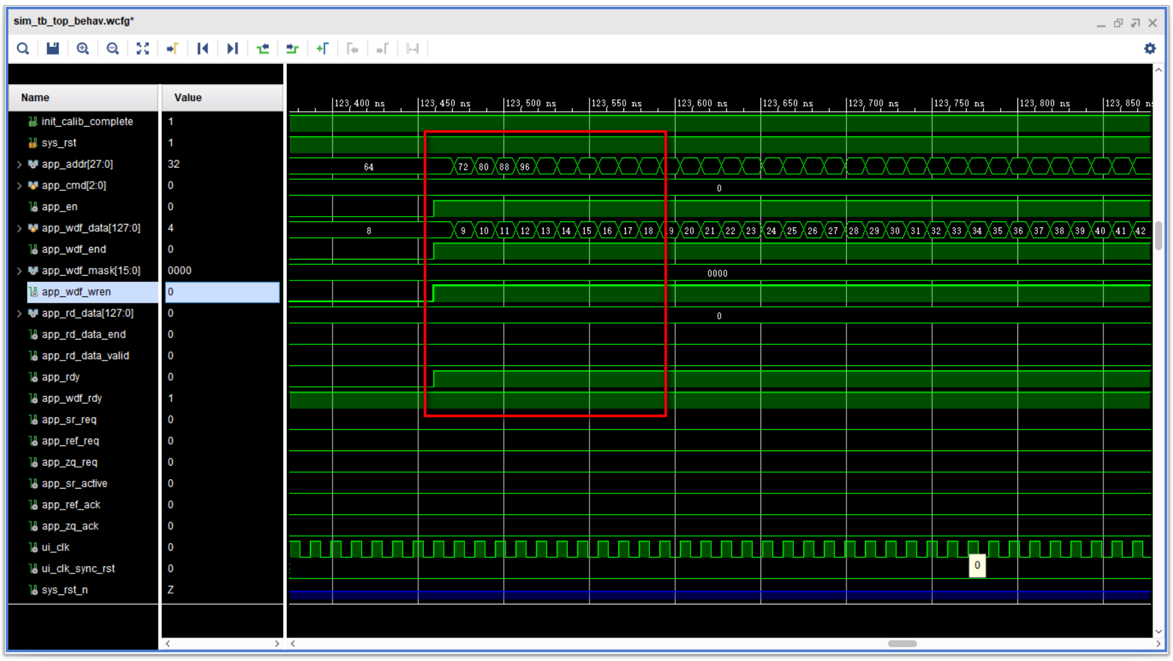
如上图所示，**app_cmd** 信号在①处为写 **ddr3** 命令，从 **bank0** 写到 **bank7**。②处为 **ddr3** 读命令，从 **bank0** 读到 **bank7**。



sim_tb_top_behav.wcfg*

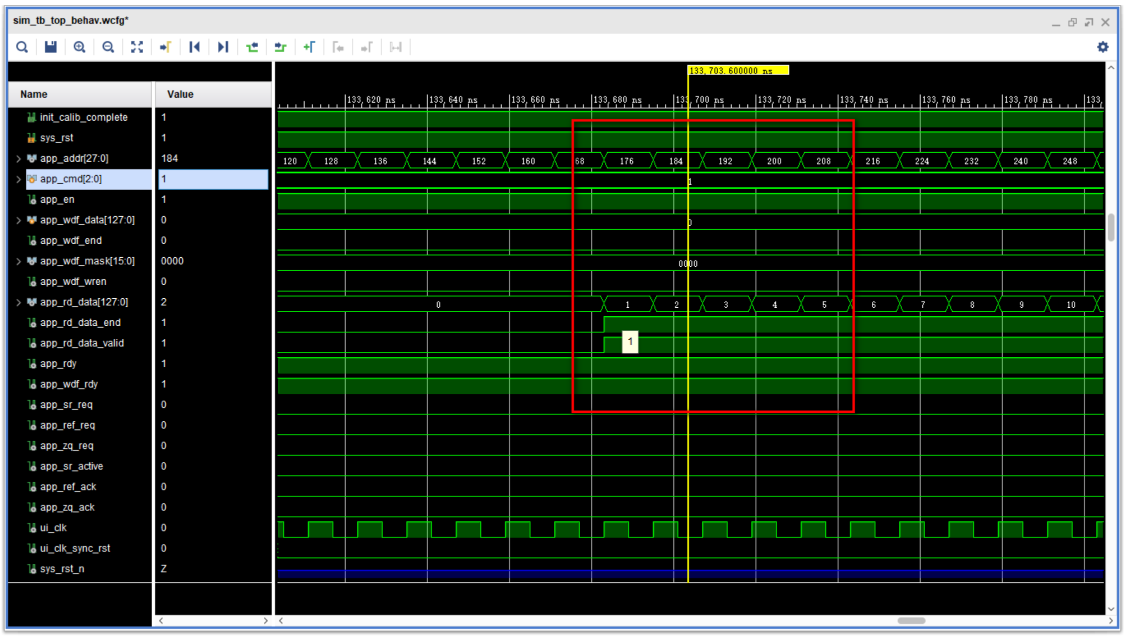
| Name | Value |
|---|---|
| init_calib_complete | 1 |
| sys_rst | 1 |
| app_addr[27:0] | 32 |
| app_cmd[2:0] | 0 |
| app_en | 0 |
| app_wdf_data[127:0] | 4 |
| app_wdf_end | 0 |
| app_wdf_mask[15:0] | 0000 |
| app_wdf_wren | 0 |
| app_rd_data[127:0] | 0 |
| app_rd_data_end | 0 |
| app_rd_data_valid | 0 |
| app_rdy | 0 |
| app_wdf_rdy | 1 |
| app_sr_req | 0 |
| app_ref_req | 0 |
| app_zq_req | 0 |
| app_sr_active | 0 |
| app_ref_ack | 0 |
| app_zq_ack | 0 |
| ui_clk | 0 |
| ui_clk_sync_rst | 0 |
| sys_rst_n | Z |

如图红框所示，我们采用写时序第一种情况（具体参看《第三篇：**mig IP** 用户读写时序》）。地址每次加 **8**，数据每次加 **1**。



| Name | Value |
|---|---|
| init_calib_complete | 1 |
| sys_rst | 1 |
| app_addr[27:0] | 32 |
| app_cmd[2:0] | 0 |
| app_en | 0 |
| app_wdf_data[127:0] | 4 |
| app_wdf_end | 0 |
| app_wdf_mask[15:0] | 0000 |
| app_wdf_wren | 0 |
| app_rd_data[127:0] | 0 |
| app_rd_data_end | 0 |
| app_rd_data_valid | 0 |
| app_rdy | 0 |
| app_wdf_rdy | 1 |
| app_sr_req | 0 |
| app_ref_req | 0 |
| app_zq_req | 0 |
| app_sr_active | 0 |
| app_ref_ack | 0 |
| app_zq_ack | 0 |
| ui_clk | 0 |
| ui_clk_sync_rst | 0 |
| sys_rst_n | Z |

如上图红框所示，连续写数据。



| Name | Value |
|---|---|
| init_calib_complete | 1 |
| sys_rst | 1 |
| app_addr[27:0] | 184 |
| app_cmd[2:0] | 1 |
| app_en | 1 |
| app_wdf_data[127:0] | 0 |
| app_wdf_end | 0 |
| app_wdf_mask[15:0] | 0000 |
| app_wdf_wren | 0 |
| app_rd_data[127:0] | 2 |
| app_rd_data_end | 1 |
| app_rd_data_valid | 1 |
| app_rdy | 1 |
| app_wdf_rdy | 1 |
| app_sr_req | 0 |
| app_ref_req | 0 |
| app_zq_req | 0 |
| app_sr_active | 0 |
| app_ref_ack | 0 |
| app_zq_ack | 0 |
| ui_clk | 0 |
| ui_clk_sync_rst | 0 |
| sys_rst_n | Z |

如上图红框所示，**app_rd_data_valid** 信号有效，读出数据和写入数据一致，仿真验证完成。

FPGA 开源工作室为了大家更好更快的学习 FPGA 目前开通了知识星球(FPGA 自习学院)。FPGA 自习学院将不断更新和总结 FPGA 相关的学习资料，欢迎大家加入，一起学习一起成长。

知识星球
FPGA自习学院
星主: OpenS_Lee
长按扫码预览社群内容
和星主关系更近一步