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

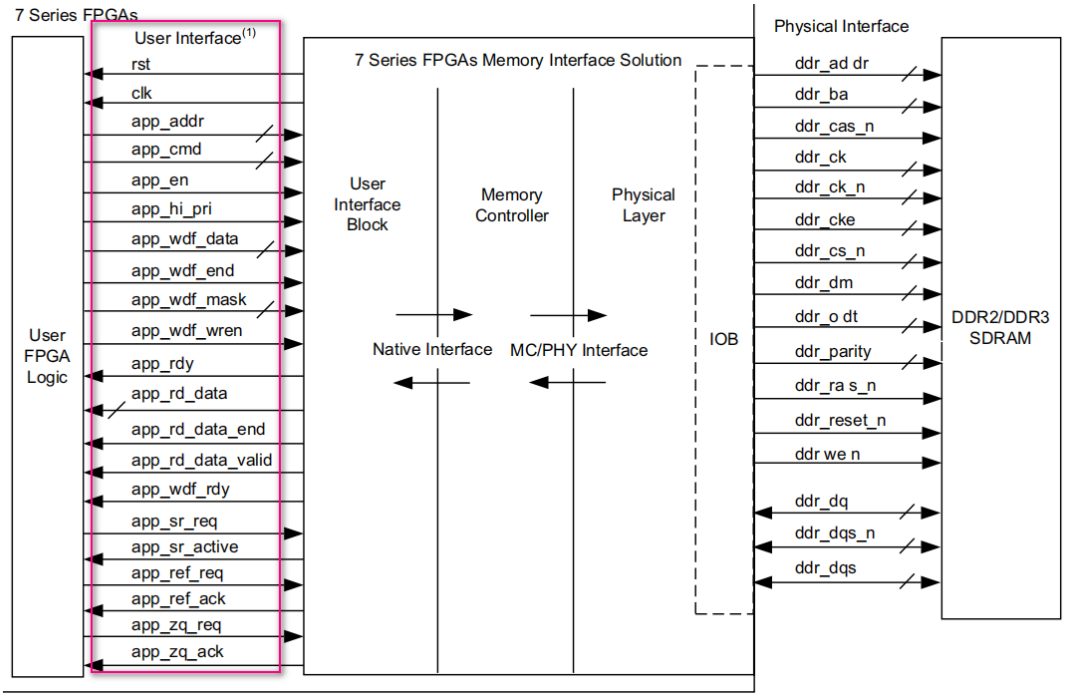
第三篇：mig IP 用户读写时序

1 mig 接口说明

Signal	Direction	Description
app_addr[ADDR_WIDTH – 1:0]	Input	This input indicates the address for the current request.
app_cmd[2:0]	Input	This input selects the command for the current request.
app_en	Input	This is the active-High strobe for the app_addr[], app_cmd[2:0], app_sz, and app_hi_pri inputs.
app_rdy	Output	This output indicates that the UI is ready to accept commands. If the signal is deasserted when app_en is enabled, the current app_cmd and app_addr must be retried until app_rdy is asserted.
app_hi_pri	Input	This active-High input elevates the priority of the current request.
app_rd_data [APP_DATA_WIDTH – 1:0]	Output	This provides the output data from read commands.
app_rd_data_end	Output	This active-High output indicates that the current clock cycle is the last cycle of output data on app_rd_data[]. This is valid only when app_rd_data_valid is active-High.
app_rd_data_valid	Output	This active-High output indicates that app_rd_data[] is valid.
app_sz	Input	This input is reserved and should be tied to 0.
app_wdf_data [APP_DATA_WIDTH – 1:0]	Input	This provides the data for write commands.
app_wdf_end	Input	This active-High input indicates that the current clock cycle is the last cycle of input data on app_wdf_data[].

app_rd_data_valid	Output	This active-High output indicates that app_rd_data[] is valid.
app_sz	Input	This input is reserved and should be tied to 0.
app_wdf_data [APP_DATA_WIDTH – 1:0]	Input	This provides the data for write commands.
app_wdf_end	Input	This active-High input indicates that the current clock cycle is the last cycle of input data on app_wdf_data[].
app_wdf_mask [APP_MASK_WIDTH – 1:0]	Input	This provides the mask for app_wdf_data[].
app_wdf_rdy	Output	This output indicates that the write data FIFO is ready to receive data. Write data is accepted when app_wdf_rdy = 1'b1 and app_wdf_wren = 1'b1.
app_wdf_wren	Input	This is the active-High strobe for app_wdf_data[].
app_correct_en_i	Input	When asserted, this active-High signal corrects single bit data errors. This input is valid only when ECC is enabled in the GUI. In the example design, this signal is always tied to 1.
app_sr_req	Input	This input is reserved and should be tied to 0.
app_sr_active	Output	This output is reserved.

Signal	Direction	Description
app_ref_req	Input	This active-High input requests that a refresh command be issued to the DRAM.
app_ref_ack	Output	This active-High output indicates that the Memory Controller has sent the requested refresh command to the PHY interface.
app_zq_req	Input	This active-High input requests that a ZQ calibration command be issued to the DRAM.
app_zq_ack	Output	This active-High output indicates that the Memory Controller has sent the requested ZQ calibration command to the PHY interface.
ui_clk	Output	This UI clock must be a half or quarter of the DRAM clock.
init_calib_complete	Output	PHY asserts init_calib_complete when calibration is finished.
app_ecc_multiple_err[7:0](1)	Output	This signal is applicable when ECC is enabled and is valid along with app_rd_data_valid. The app_ecc_multiple_err[3:0] signal is non-zero if the read data from the external memory has two bit errors per beat of the read burst. The SECDED algorithm does not correct the corresponding read data and puts a non-zero value on this signal to notify the corrupted read data at the UI.
ui_clk_sync_rst	Output	This is the active-High UI reset.
app_ecc_single_err[7:0]	Output	This signal is applicable when ECC is enabled and is valid along with app_rd_data_vali. The app_ecc_single_err signal is non-zero if the read data from the external memory has a single bit error per beat of the read burst.



对于 mig 与 DDR3/DDR2 SDRAM 的读写时序我们不需要了解太多，交给 mig 就可以了。我们需要做的是控制好 User Interface，写出正确的 User logic。想要写好 User logic，我们就必须清楚每一个用户控制接口的含义：

a. app_addr[ADDR_WIDTH - 1:0]：此输入指示当前提交给 UI 的请求的地址。 UI 聚合外部 SDRAM 的所有地址字段，并为您提供平面地址空间（Rank,bank,row,colum）。

b. app_cmd[2:0]：此输入指定当前提交给 UI 的请求的命令。

Operation	app_cmd[2:0] Code
Read	001
Write	000

c. app_en：此输入在请求中变化。 您必须将所需的值应用于 app_addr []和 app_cmd [2：0]，然后断言 app_en 以将请求提交给 UI。 这会通过断言 app_rdy 来启动 UI 确认的握手。

d. app_wdf_data[APP_DATA_WIDTH – 1:0]：该总线提供当前正在写入外部存储器的数据。

e. app_wdf_end：此输入指示当前周期中 app_wdf_data []总线上的数据是当前请求的最后一个数据。

f. app_wdf_wren：此输入表示 app_wdf_data []总线上的数据有效。

g. app_wdf_rdy：此输出表示写数据 FIFO 已准备好接收数据。 当 app_wdf_rdy 和 app_wdf_wren 都被声明时，接受写入数据。

h. app_wdf_mask[APP_MASK_WIDTH – 1:0]：该总线指示 app_wdf_data []的哪些字节写入外部存储器以及哪些字节保持其当前状态。 通过将值 1 设置为 app_wdf_mask 中的相应位来屏蔽字节。 例如，如果应用程序数据宽度为 256，则掩码宽度取值为 32. app_wdf_data 的最低有效字节[7：0]使用 app_wdf_mask 的 Bit [0]屏蔽，app_wdf_data 的最高有效字节[255：248]使用 app_wdf_mask 的 Bit [31]屏蔽。 因此，如果必须屏蔽最后一个 DWORD，即 app_wdf_data 的字节 0,1,2 和 3，则 app_wdf_mask 应设置为 32'h0000_000F。

i. app_rdy:此输出指示您是否接受当前提交给 UI 的请求。 如果在确认 app_en 后 UI 未声明此信号，则必须重试当前请求。 如果出现以下情况，则不会声明 app_rdy 输出：
1>PHY /内存初始化尚未完成;
2>所有 bank 都被占用（可以看作命令缓冲区已满）;
3>请求读取并且读取缓冲区已满;
4>请求写入，没有可用的写缓冲区指针;
5>正在插入定期读取。

j. app_rd_data[APP_DATA_WIDTH – 1:0]：此输出包含从外部存储器读取的数据。

k. app_rd_data_end：此输出表示当前周期中 app_rd_data []总线上的数据是当前请求的最后一个数据。

l. app_rd_data_valid：此输出表明 app_rd_data []总线上的数据有效。

m. ui_clk_sync_rst：reset 信号来自于 UI，与 ui_clk 同步。

n. ui_clk：这是 UI 的输出时钟。 它必须是输出到外部 SDRAM 的时钟频率的一半或四分之一，这取决于在 GUI 中选择的 2：1 或 4：1 模式。

o. init_calib_complete：校准完成后，PHY 将 init_calib_complete 置‘1’。 在将命令发送到内存控制器之前，应用程序无需等待 init_calib_complete。

p. app_ref_req:置位时，此高电平有效输入请求内存控制器向 DRAM 发送刷新命令。 它必须在一个周期内进行脉冲以发出请求,直到 app_ref_ack 信号被置位以确认请求并指示它已被发送,然后置为无效。

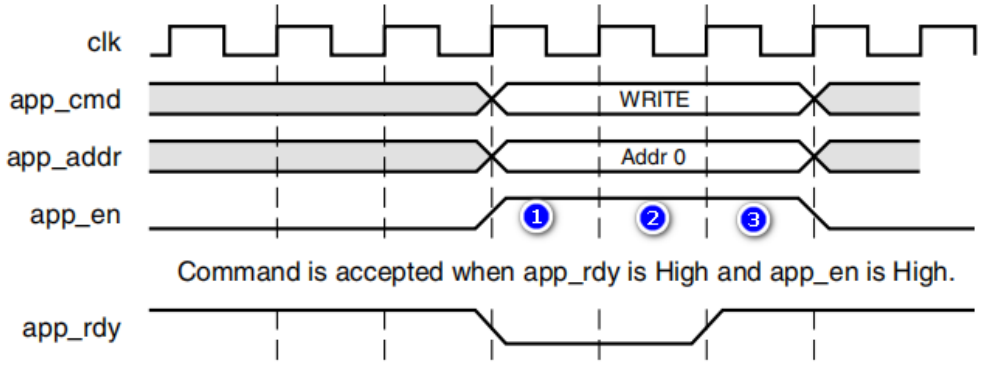
q. app_ref_ack:置位时，此高电平有效输入确认刷新请求，并指示该命令已从存储器控制器发送到 PHY。

r. app_zq_req:置位时,此高电平有效输入请求存储器控制器向DRAM发送ZQ校准命令。它必须在一个周期内进行脉冲以发出请求，直到 app_zq_ack 信号被置位以确认请求并指示它已被发送,然后取消置位。

s. app_zq_ack:置位时，此高电平有效输入确认 ZQ 校准请求，并指示该命令已从存储器控制器发送到 PHY。

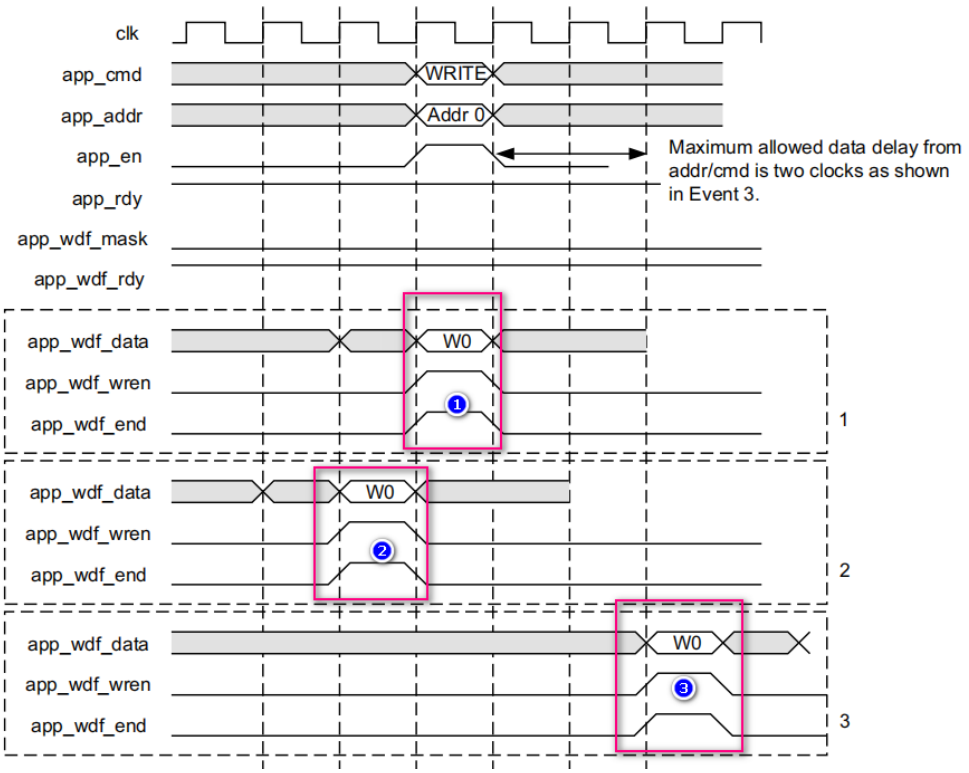
了解了每个信号的作用我们来讲解 mig 用户逻辑的读写时序。

2 命令与地址



如上图所示①,②,③情况，只有在③时刻 app_en 和 app_rdy 同时为高电平 app_cmd（命令）和（app_addr）地址才有效，所以当需要 app_cmd,app_addr 有效时 app_en 必须保持到 app_rdy 为高电平才有效。

3 写时序



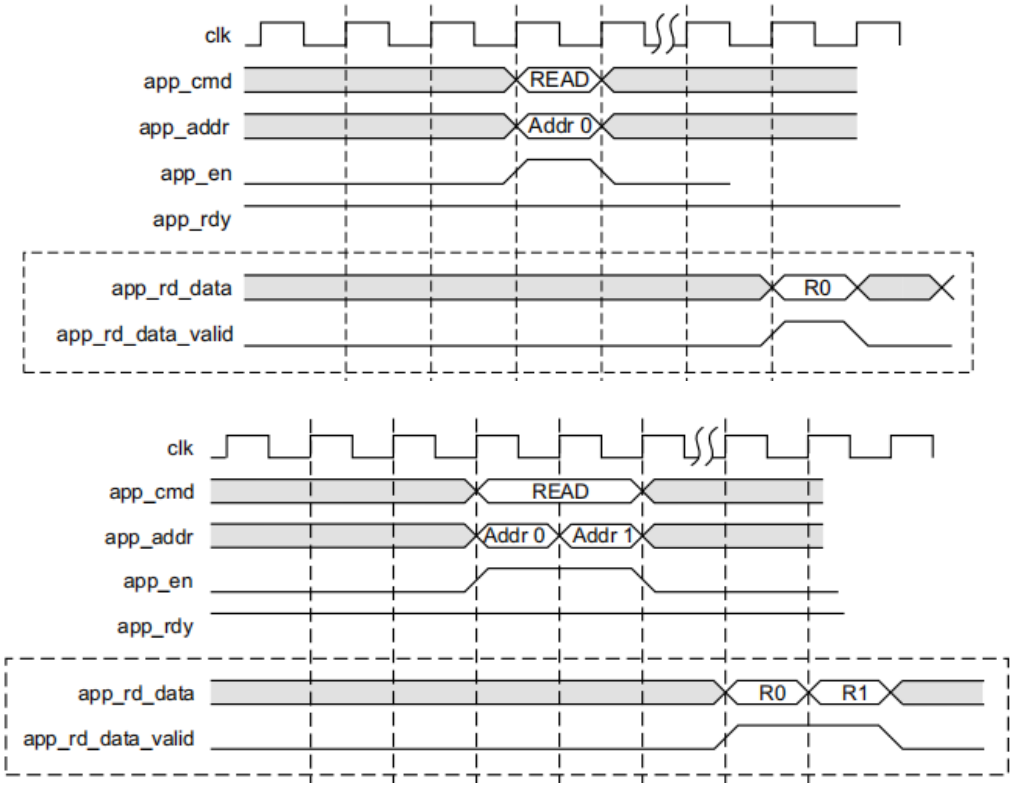
如上图所示①，②，③种情况，写命令和写数据直接存在三种逻辑关系。

1，①表示写命令（app_cmd），写当前地址(app_addr)和写数据(app_wdf_data)以及写控制信号(app_en,app_rdy,app_wdf_rdy,app_wdf_wren,app_wdf_end)同时有效。

2，②表示写数据（app_wdf_data）和写控制信号（app_wdf_wren,app_wdf_end）先于写命令（app_cmd）和写当前地址（app_addr）以及其他写控制信号（app_en,app_rdy,app_wdf_rdy）一个用户时钟（ui_clk）。

3，③表示写数据（app_wdf_data）和写控制信号（app_wdf_wren,app_wdf_end）迟于写命令（app_cmd）和写当前地址（app_addr）以及其他写控制信号（app_en,app_rdy,app_wdf_rdy）。最多两个用户时钟（ui_clk）。

4 读时序



如上图所示，当读命令(app_cmd)和当前读地址(app_addr)以及读控制信号(app_en，app_rdy）同时有效时，等待读数据有效信号（app_rd_data_valid）有效时读数据（app_rd_data）有效。

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