Design of SDRAM Controller IP Core Based on FPGA*

Yonggang Chen[1,a], Yawen Li[2,b], Caizhen Zhang[2,a]

[1]School of Automation & Electrical Engineering, Lanzhou Jiao Tong University, Lanzhou, 730070, Gansu, CHN

[2]School of Electronic and Information Engineering, Lanzhou Jiao tong University, Lanzhou , 730070 ,Gansu , CHN

[a]email: zhangcaizhen@mail.lzjtu.cn, [b]emal: wydxxzj@163.com

[*]Corresponding Author：Caizhen Zhang.

Keywords: FPGA, SDRAM Controller, IP core, Timing Analysis.

Abstract. Although having advantages of high integration, Low power consumption and strong processing capability, etc., it is not easy for SDRAM to be developed and applied because of its timing complexity. To reduce costs, and shorten the development period, combining with features of strong reconfiguration and portability for the design based on FPGA, particular timing constraints for the read and write processes of SDRAM controller were made and SDRAM controller IP soft core was designed according to SDRAM control norms, using EDA top-down design method. The IP core was verified with Altera's FPGA-EP2C35F484C8 devices. Timing simulation and SignalTapII logic analyzer sampling results show that the designed IP core is in line with SDRAM timing requirements and can operate reliably and continuously. The design has high reliability and universal applicability.

1. Introduction

With advantages of high execution speed, large scale, etc., in recent years, LUT-based FPGA obtains a wide range of application in the fields of image processing, control systems and so on. In an application system, an external memory is usually used to save RAM resources of FPGA chip[1-2]. Having advantages of high speed and low-cost, SDRAM (Synchronous Random Access Memory) is widely used as an external memory, while it is not easy for SDRAM to be controlled and developed[3] because of its complex timing. Although adopting of IP hard core[4] (Intellectual property core, intellectual property cores) used to replace the user's design logic can shorten the development period and improve development efficiency, the entire application systems are expensive.

Yang Haitao et al[5] completed the design and implementation of a SDRAM controller based on FPGA, but the controller is for a specific SDRAM chips, and the speed is only 50MHz, it is necessary to improve its versatility and portability. In this paper, combining the structure of FPGA and requirements of SDRAM controller, logic functions and timing were optimized, especially, particular timing constraints for the read and write processes of SDRAM controller were made and a portable IP soft core of SDRAM controller was designed.

2. SDRAM Timing Analysis

Basic operations of SDRAM include internal chip initialization, Line efficiency, the column read and write, data output (read operation), data input (write operation), burst length, precharge, refresh, data mask., etc.

A specific storage unit is fixed after the column address is selected, then data output to memory buses is completed via data I / O channels. SDRAM read timing is shown in Fig.1, a CLK equaling 100MHz is obtained at the FPGA_CLK port by doubling frequency. It can be seen that provided the signal CAS is given, data output to SDRAM output ports will be finished after a Tac time, then data will be transmitted to the FPGA data ports and retained a Toh time by means of the following SDRAM clock source, and FPGA sampling data is carried out at the same time[6].

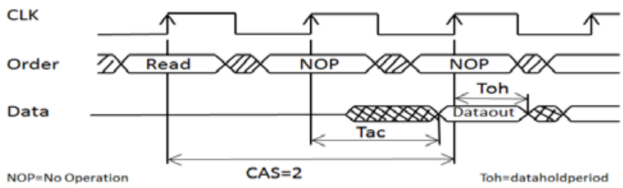


Fig.1 SDRAM read timing

Data write operation is performed after a fixed machine period delay. During column addressing, the signal WE is valid, a corresponding L-BANK address is locked. Fig. 2 shows SDRAM write timing, it can be seen that since the data signal is sent out by the control terminal, no adjustment is needed to be done during data input process, the thing needed to be done is to send data to ports Dinn (data buses) sequentially and transmit them to data input registers directly, then the storage capacitor is charged by the write driver, so data can be sent simultaneously with the signal CAS, in other words, the write delay is 0. To ensure written data reliable, enough writing/correction time should be given[7], the time occupies one or more clock period at least.

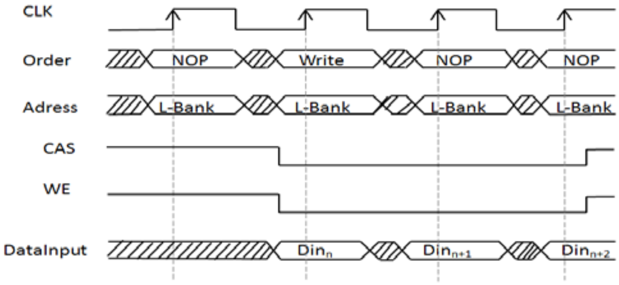


Fig.2 SDRAM write timing

Because the clock of SDRAM is from FPGA during data output, SDRAM controller belongs to a communication structure of variant common clock system. According to the SDRAM control requirements[8-11], when data are written from FPGA to a SDRAM, the minimum holding time of SDRAM input is set as 3 ns, the clock period of the SDRAM Tclk is 10ns, the maximum delay time Taccesss_time_max is 6ns; while data is read from SDRAM to FPGA, the SDRAM internal delay time is 6ns, the minimum data input setup time of FPGA reading SDRAM setup_out_min is 2ns, minimum holding time Thold_out_min is 3ns. Then the minimum data output setup time of FPGA writing SDRAM is described as:

Tsetup_in_min=Tclk-Taccess_time_max=10ns-6ns=4ns (1)

Since internal clock buffers are contained in FPGA, the filter time Tflt can be ignored, so as to the clock jitter, the timing constraint Eq.1[9] may be reduced to:

Tsetup_margin _read = Tclk- Tflt_clk- Tflt_data- Tco_data – Tsetup
= Tsetup_SDRAMout_max - Tclk_delay – Tdata_delay–Tsetup (2)
Thold_margin _read = Tco_data + Tdata_delay + Tclk_delay – Thold

= Thold_SDRAMout_min + Tdata_delay + Tclk_delay – Thold (3)

Substituting data in the Eq.2, 3, the read timing constraints are:

Tsetup_margin _read = Tsetup_max - Tclk_delay – Tdata_delay –Tsetup
= 8ns - Tclk_delay – Tdata_delay > 0ns
Tdata_delay + Tclk_delay < 8ns (4)
Thold_margin_read = Thold_out_min + Tdata_delay + Tclk_delay –Thold_in_min

$= 3\text{ns} + \text{Tdata\_delay} + \text{Tclk\_delay} - 3\text{ns} > 0\text{ns}$

$\text{Tdata\_delay} + \text{Tclk\_delay} > 0\text{ns}$ (5)

Similarly, write timing constraints are:

$\text{Tsetup\_margin\_write} = \text{Tclk\_delay} + \text{Tsetup\_out} - \text{Tdata\_delay} - \text{Tsetup\_in}$

$= \text{Tclk\_delay} + 4\text{ns} - \text{Tdata\_delay} - 3\text{ns} > 0$

$\text{Tclk\_delay} - \text{Tdata\_delay} > 1\text{ns}$ (6)

$\text{Thold\_margin \_write} = \text{Tdata\_delay} + \text{Thold\_out} - \text{Tclk\_delay} - \text{Thold\_in}$

$\text{Tclk\_delay} - \text{Tdata\_delay} < 4\text{ns}$ (7)

Considering differences between SDRAMs with various brands, more strict constraints is considered in Eq.7 to improve design compatibility[10]

$1\text{ns} < \text{Tclk\_delay} - \text{Tdata\_delay} < 3\text{ns}$ (8)

During an actual routing, the length of a clock line is usually longer or the same as that of the data and address lines, so, Eq.8 can be changed as[11]:.

$0\text{ns} < \text{Tclk\_delay} - \text{Tdata\_delay} < 3\text{ns}$ (9)

$1\text{ns} < \text{Tdata\_delay} + \text{Tclk\_delay} < 3\text{ns}$ (10)

3. SDRAM Controller Design

The designed SDRAM controller consists of the following six modules: serial control module; read FIFO module; write FIFO module; PLL clock module; data and address generating module along with SDRAM control module. The whole logical relationship is shown in Fig. 3.

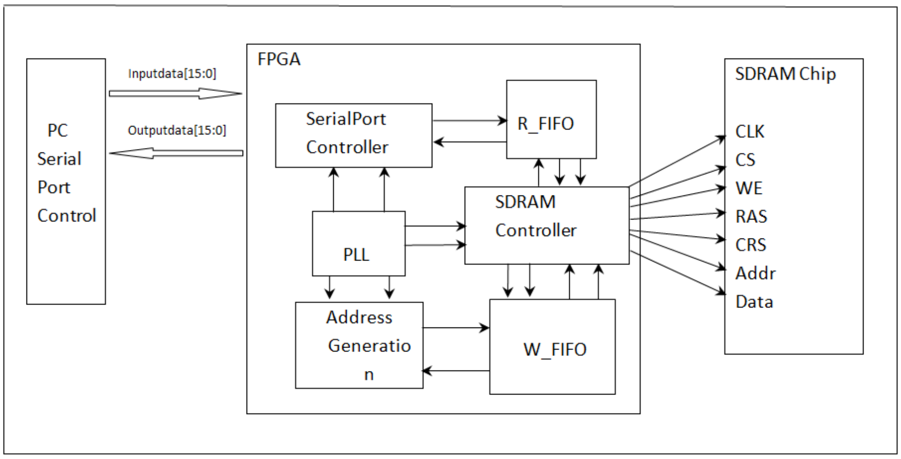


Fig.3 System composition block diagram

3.1 Main Modules Design of SDRAM

1) Read / Write FIFO Module

As a data buffer, asynchronous FIFO is widely used as a high-speed real-time data acquisition, high-performance data transfer between different clock domains and multi-machine processing system. The designed FIFO controller is to control data communications between the two pieces of on-chip FIFO and SDRAM. When the write enable signal WR_EN is effective (high level), the input data (16bit wide) Din [N: 0] triggered by writing clock signal WR_CLK rising edge is transferred through data width conversion module 1 and then written in FIFO1 with 16 bit wide and 2K depth. When the SDRAM controller detects FIFO1 is almost full, it begins to read the data, and then write them in the SDRAM according to increasing addresses by SDRAM controller until FIFO1 is empty. When the FIFO controller detects FIFO2 is going to be empty, it starts to read the SDRAM data sequentially according to increasing addresses and then write in FIFO2 by SDRAM controller until FIFO2 is filled[12].

2) Address Generating Module

This module receives a command request and refresh request firstly, and then responds a request, generates a corresponding operation signal as well as sends it to SDRAM according to the principle of the arbiter that refresh request takes precedence over the command request, at the same time, the module generates a enable signal OE and sends it to data path module to control the data flow. Since the SDRAM address signal is time-multiplexed, address signal ADDR of controller are block address, row address, and column address in a decreasing order. In this module, block, row and column addresses included in ADDR are written in the response register, block and row addresses are sent to SDRAM when the row command is active, then column address is sent to SDRAM when read or write command is issued[13].

3) SDRAM Control Module

According to operating principles of SDRAM, states of the model include power-on state (poweron), idle state (idle), self-refresh state (Auto_Refresh), pre-charge state (Precharge), active state (Active), read state (R_Operation), write state (W_Operation) and set mode register state (Load_MRS), etc. Fig.4 is the state transition diagram. The process includes: waiting a fixed time after the power-on reset and entering the pre-charge state, finishing mode register settings after eight times self-refresh operations at the arrival of the reference clock period. Internal states mainly switch among states of self-refresh(Auto_Refresh), read/write operations(W/ R_Operation), active state (Active), read/write burst length (WR / RD_BT) and precharge (Precharge), ensuring reliable working of the entire module.

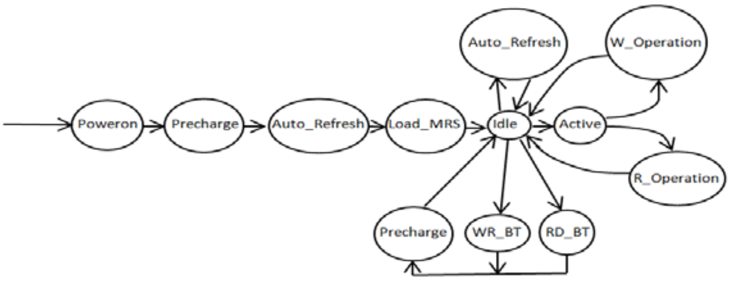


Fig.4 SDRAM state transition diagram

4. Test Analysis of SDRAM

SDRAM write data simulation waves were completed in ModelSim-Altera 10.1d. As shown in Fig.5, signal sys_data_in is the data sent to SDRAM control module by FIFO, signal sdram_data is the SDRAM data bus, there is one clock period delay between the two signals. It can be seen that,at first, a row effective command(10011) was sent by SDRAM controller using a group of control signals (sdram_CKE_n / sdram_CS_n / sdram_AS_n / sdram_RAS_n / sdram_CAS_n / sdram_WE_n), selecting the first line 0 among the first logic block 0, then after two clock period, a write command (10100) is sent, that is to say, the column address 0 is selected, simultaneously, the first data 0x0000 is sent to the data bus without delay, eight 16-bit data are written when the command is executed once.

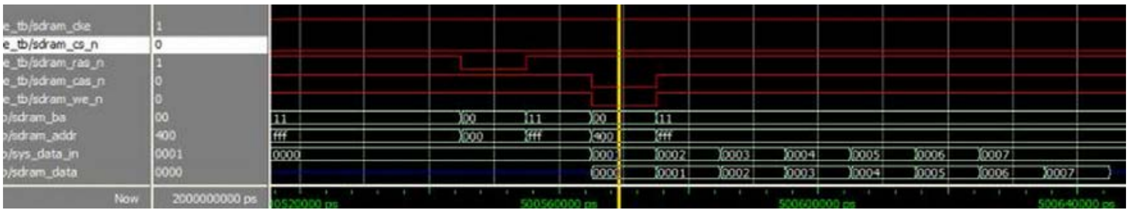
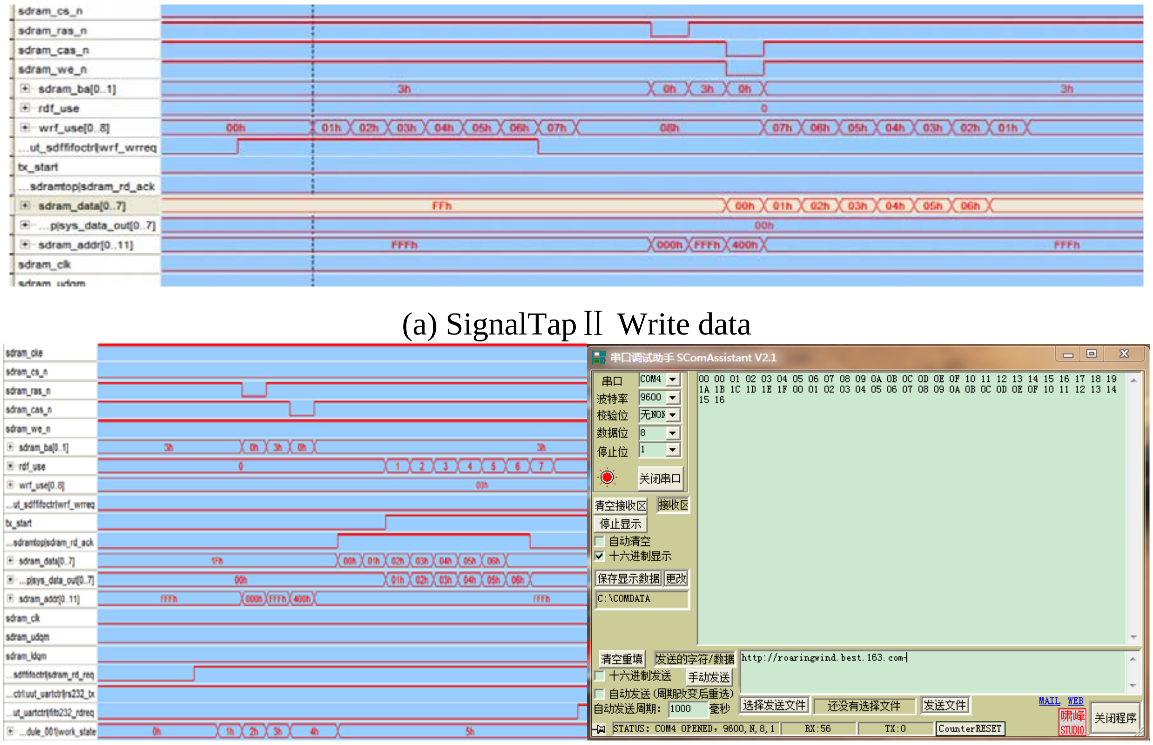


Fig.5 Write data simulation

The design was adapted and downloaded to Altera-EP2C35F484C8 chip to finish the hardware testing. Fig.6 shows read and write data sampling results by the embedded logic analyzer SignalTapII in FPGA.



(a) SignalTapⅡ Write data

串口 COM4
波特率 9600
校验位 无NON
数据位 8
停止位 1

```
00 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19
1A 1B 1C 1D 1E 1F 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14
15 16
```

□ 自动清空
☑ 十六进制显示
C:\COMDATA
http://roaringwind.best.163.com
□ 十六进制发送
□ 自动发送(周期改变后重选)
自动发送周期: 1000 毫秒
STATUS: COM4 OPENED, 9600, N, 8, 1 RX:56 TX:0

(b) SignalTapⅡ Read data (c) COM Debug Assistant Read data

Fig.6 Sampling results in SignalTap II

Fig.6 (a) shows that, when data is written, the corresponding address sdram_addr [0:11] is 0x400, which indicates data written in address sdram_data [0: 7] from 0x0000, since the burst length is 8, the data is written 8 times successively. Data read operation is shown in the left of Fig.6 (b), the first read operation is started at address 0. After data is read out, they are sent to “Read FIFO”, and then sent out through the serial port. Since signal RAS is set as 2, after two clock period of the read command (signal cas_n is 0), low 8-bit data bus sdram_data [0: 7] changes from 1FH to 00H. Then signal rdf_use increase, which indicates the data is written in "read FIFO". When tx_start signal is 1, it indicates the serial port starts, the baud rate is set to 9600, through the serial debugging assistant, it can be seen that the entire data can be read as shown in right of Fig.6 (b).

The above results indicate the designed SDRAM controller could accurately complete read and write operations.

5 Conclusions

An RTL-level SDRAM controller IP soft core was designed with the top-down design method, the design was verified by means of Altera's FPGA-EP2C35F484C8 device. Modelsim simulation and hardware test results by SignalTapII show that the design meet SDRAM control specification, it can accurately complete read and write operations, and meet the timing requirements of the design. A special timing constraints was added in the design to ensure the reliability of SDRAM at high working speed and enhance portability of SDRAM controller.

This work was supported by the Basic Scientific Research Professional Foundation (Grant no. 620027).

References

[1]ZHANG Xiu-Yan,ZHAO Feng-Jun. Design of DDR SDRAM based on FPGA IP Core[C]. Institute of Electronics,Chinese Academy of Sciences,Beijing 100080,China, (2006), pp:261-263.

[2] Shivani Madhok, Navdeep Singh, Jasleen Kaur, etc. HSTL IO Standard Based Energy Efficient FIR Filter Design on 28nm FPGA[J]. International Journal of Control and Automation. Vol. 8, No. 7 (2015), pp. 47-54.

[3] Wu Zhen, Lang Weiyan. Design and Implementation of the SDRAM Controller based on FPGA[J]. Journal of North China Institute of Aerospace Engineering, Vol.22, No.5, (2012), pp:10-13.

[4] JIANG Hao, XIAO Tie-jun. IP core design of PCI-slave interface based on FPGA[J]. Computer Engineerig and Design. Vol.29, No.11, (2008), pp:2785-2788.

[5] Yang Haitao, Su Tao, Wu Men. The Design and Implementation of the SDRAM Controller Based on the FPGA[J]. Electronic Science and Technology, No.01, (2007), pp:8-12.

[6] Wang, Jianmin;Zhang, Yanqin;Zhou, Jinhu, Jia Peng, He, Xunjun. Realization of Multi-port SDRAM Controller in LXI Data Acquisition System[C]. 2013 International Conference on Information Science and Cloud Computing Companion, 2014 IEEE, Guangzhou, China. (2013), pp:515-520.

[7] Yang Yinghui. Design of SDRAM Controller and Its Application on FPGA[D]. Lanzhou, Lanzhou University, (2007), pp:13-20.

[8] WU Jie-yi, PANG Rui, CHENG An-yu, LUO Hong-ping. A solution of SDRAM interface timing compatibility[J]. Digital Communication.Vol.5, (2009), pp:81-83.

[9] ZHANG Hua-gao, CHEN Lan. Design and constraint method of DDR source synchronous interface[J]. Computer Engineering and Design. Vol.29, No.7, (2008), pp:1600-1605.

[10] Qu Fuyang. DDR SDRAM Interface Design and Static Timing Analysis[D]. Xi’an, XIDIAN UNIVERSITY, (2012), PP:30-40.

[11] Gao Yuan, Song Jie. The analysis of SDRAM interface timing and PCB layout line length[J]. Electronic Engineering & Product World. Vol.9, (2011), pp:47-50.

[12] Gao Qunfu, Chen Xing, Cheng Yue. Design and implementation of DDR SDRAM controller based on FPGA[J]. ELECTRONIC MEASUREMENT TECHNOLOGY. Vol. 34, No. 8, (2011), pp:56-59.

[13] FAN Ji,WU Xiao-yi,WANG Bin-yi,HU Jiang-feng. FPGA Realization of SDRAM Controller Design in Digital System[J]. JOURNAL OF GUN LAUNCH&CONTROL. No.4, (2009), pp:38-41.