Evaluation of the μController Networks Communication for EtherCAT Process Data Interface

Li Jiang[1], Mingxin Hou[1], Fanshao Wei*,[1], Minghe Jin[1], Hong Liu[1]
Harbin Institute of Technology, Harbin, China
Email: jamesking666666@gmail.com; fansw@hit.edu.cn

Zhaopeng Chen[2]
German Aerospace Center, Munich, Germany
Email: czp_unique@hotmail.com

Abstract: - According to the EtherCAT process data interface networks system designs, we determined that the Synchronous and Asynchronous μController scheme evaluations based on the FPGA state machine algorithm. The present study focuses on the Schematic of μController interconnection between FPGA and ESC based on the VHDL state machine algorithm, which includes state machine programs. In addition, the performance of the Synchronous and Asynchronous μController was verified through numerical simulations. Furthermore, we found the simulations gave further evidence that the 16 bits synchronous and Asynchronous μController signals can satisfy the two write access and a read access requirements for the EtherCAT networks communication. Finally, the results include the positive and negative aspects of the Synchronous and Asynchronous μController networks communication, and the comparison the results of the Synchronous, Asynchronous and SPI communication evaluations done by the researchers.

Key-Words: - Synchronous network communication; Asynchronous network communication; EtherCAT; FPGA State Machine; μController

1 Introduction

We are currently witnessing an exponential growth in Ethernet for Control Automation Technology (EtherCAT) use and a corresponding increase in research to provide better utilization of EtherCAT. An EtherCAT Slave Controller (ESC) takes care of the EtherCAT communication as an interface between the EtherCAT master fieldbus and the slave application [1, 2]. Typical application fields for the EtherCAT are control of machines, such as motion control [3, 4], high precision motion system [5], and modular robots [6, 7]. Gianluca and Ivan Cibrario [8] evaluated the performance of the synchronization technology of the EtherCAT, namely the Distributed Clock. And [9] proved EtherCAT enables high-performance machine control to be realized and is capable of exchanging distributed signals with cycle times significantly below 100us. However, these applications generally treat the EtherCAT master or slave structure, which may not be an accurate reflection of the ESC and the μController networks design.

The performance of EtherCAT Process Data Interface (PDI) is mainly determined by the μController design by which the memory of an ESC networks can be used for exchanging data between the EtherCAT master and a local application (on a μController attached to the PDI) without any restrictions [11-13]. This work was inspired from the concept of ESC μController applications in order to realize the write access and read access from the μController to the ESC (ET1100) networks, especially to the FPGA as a μController. By creating the corresponding Synchronous and Asynchronous networks signals based on FPGA State Machine algorithm, we could get the precise signals, which can satisfy the requirements between the FPGA μController and the ESC. The motivation here is to obtain a 16 bit FPGA Synchronous and Asynchronous μController, which shall provide the ESC hardware access layer (hardware sequence circuit). However, much remains to be done to understand and exploit the design of the ESC μController. For instance, the structure of the μController should be explored and the methods of the sequence of the write access and read access should be analyzed. To evaluate the basic Synchronous and Asynchronous networks design principles of the ESC μController, we adopt a FPGA State Machine-based solution that will be

Table 1. Controller Signals between FPGA and ESC

Signal	*Direction*	*Polarity*	*Description*
CPU_CLK_IN	FPGA to ESC	-	µController interface clock
EEPROM LOAD	ESC to FPGA	Act high	PDI is active, EEPROM is loaded
CS	FPGA to ESC	Act low	Chip Select
ADR[15:0]	FPGA to ESC	Act high	Address Bus
BHE	FPGA to ESC	Act low	Byte High Enable
DATA[15:0]	Bidirectional Data Bus	Act high	Data bus for 16 Bit Controller interface
RD	FPGA to ESC	Act low	Read command
WR	FPGA to ESC	Act low	Write command

2.2 Write access

In order to operate the 16 bit Synchronous and Asynchronous µController, the EEPROM LOADED signal must act high signal to the FPGA, which means the EEPROM of the ESC (ET1100) has been loaded as shown in Fig. 2. For the Synchronous µController, figure 2 (a), a write access starts with a Transfer Start (TS) and the Chip Select (CS) is together with TS. The CPU CLK IN edge at which CS is sampled can be configured by FPGA, where the maximum frequency of the CPU CLK IN is no more than 40 MHz. The ADR, BHE and RD/WR are valid together with TS. Once the EtherCAT device has finished the access, Transfer Acknowledge is asserted for one clock cycle. For the Asynchronous µController, figure 2 (b), a write access starts with assertion of Chip Select (CS), if it is not permanently asserted. ADR, Byte High Enable and Write Data are asserted with the falling edge of WR (active low). Shortly after the rising edge of WR, the access can be finished by deserting ADR, BHE and DATA. Internally, the write access is performed after the rising edge of WR, this allows for fast write accesses. Nevertheless, an access following immediately will be delayed by the preceding write access.

2.3 Read access

After the high signal from the ESC EEPROM LOAD, a read Synchronous access starts with a Transfer Start (TS) and the Chip Select (CS) is together with TS (it doesn't need to be configured), as depicted in Fig.3(a). The same as the frequency of the write access, where the maximum frequency of the read access clock signal CPU CLK is also no more than 40 MHz. Once the EtherCAT device has finished the access, Transfer Acknowledge is asserted for one clock cycle together with the read data. Asynchronous read access starts with assertion of Chip Select (CS), if it is not permanently asserted, as depicted in Fig.3 (b). Address and BHE have to be valid before the falling edge of RD, which signals the start of the access. The read data will remain valid until ADR, BHE, RD or CS change. The data bus will be driven while CS and RD are asserted. We expect that the read access to the µController always to be a 16 bit read access, regardless of the Byte Select signal. For this reason, it is configurable that the Byte Select signal is ignored and a read access is always a 16 bit access.

3 Conclusion ESC State Machine Algorithm

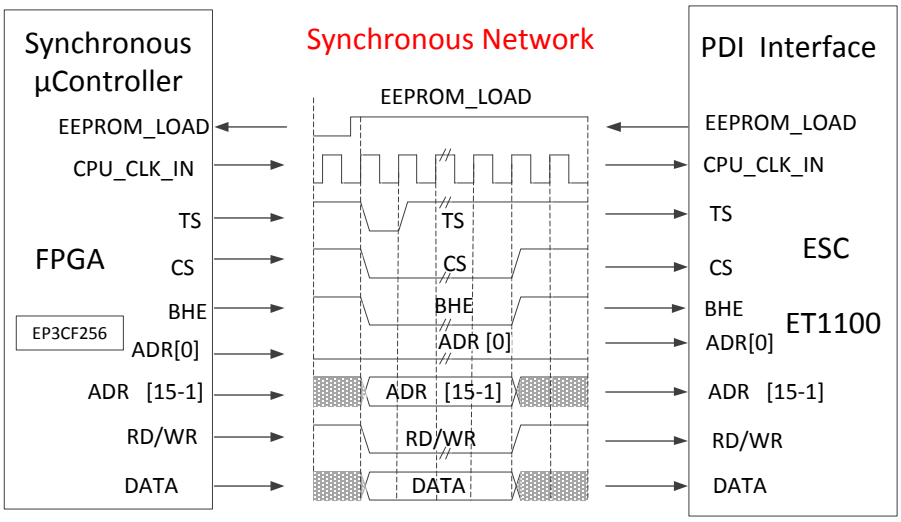
3.1 Sequence of State Machine

Figure 4 shows the Synchronous and Asynchronous networks sequence of two write accesses and a read access. The states algorithm diagrams from Idle state to Read-Data state are shown in Figure 5.

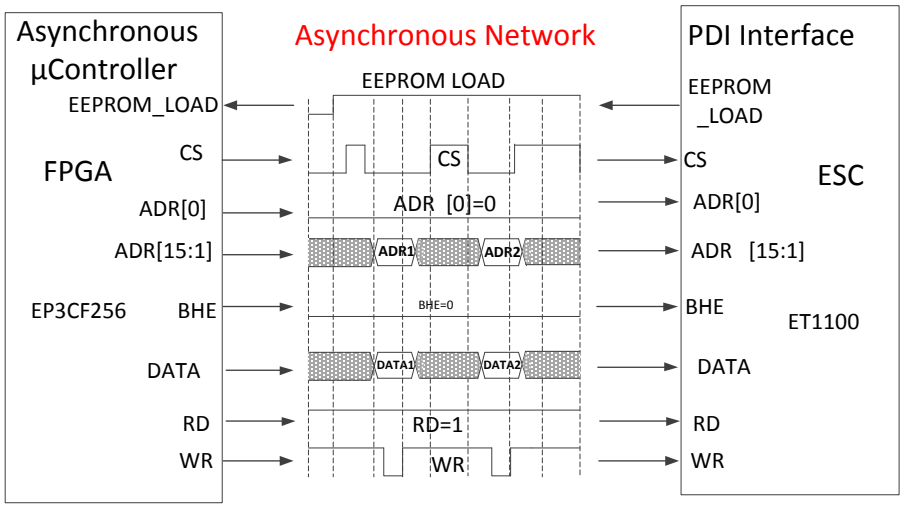
3.2 ESC State Machine Algorithm

In this state machine algorithm case, registers are necessary, so they will be inferred by the Quartus II compiler. To satisfy the Synchronous and Asynchronous networks state diagrams criteria above, as shown in figure 5, the following rules should be observed:

- Rule 1: Make sure that all the Synchronous and Asynchronous µController ESC signals used in the PROCESS will be appeared in the Table 1.
- Rule 2: Make sure that all the combination of the FPGA input or output signals are included in the code; that is, make sure that, by looking at the code, the sequence of the two write accesses and a read access as well as the concurrent code can be obtained.

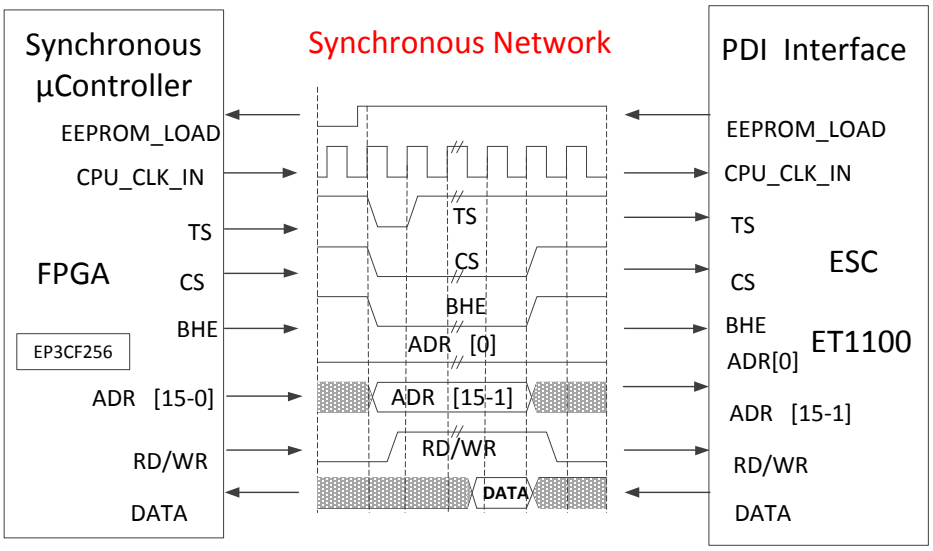


(a) Synchronous Communication

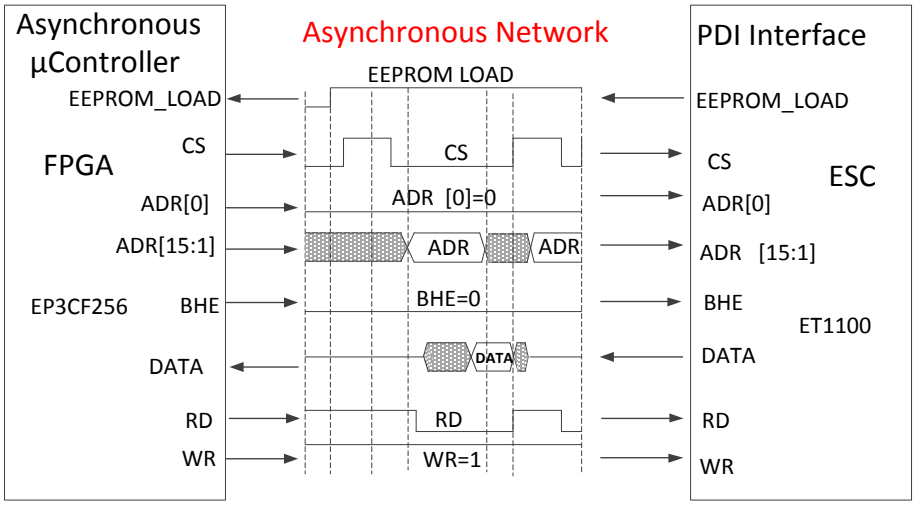


(b) Asynchronous Communication

Figure 2. Write access from FPGA to ESC



(a) Synchronous Communication

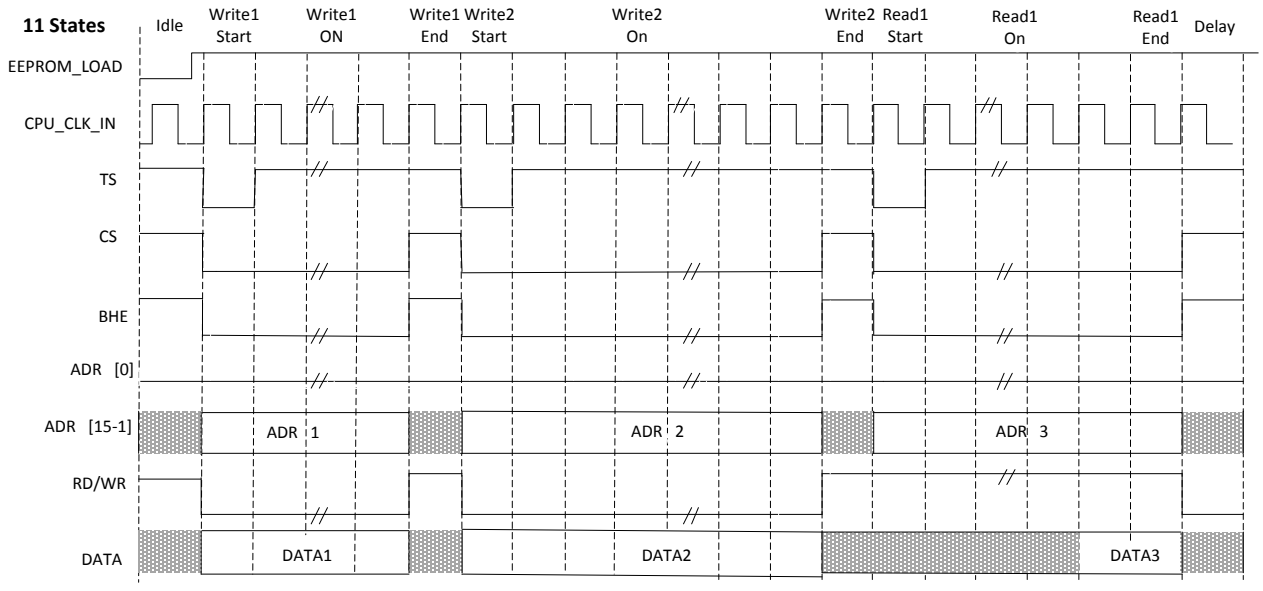


(b) Asynchronous Communication

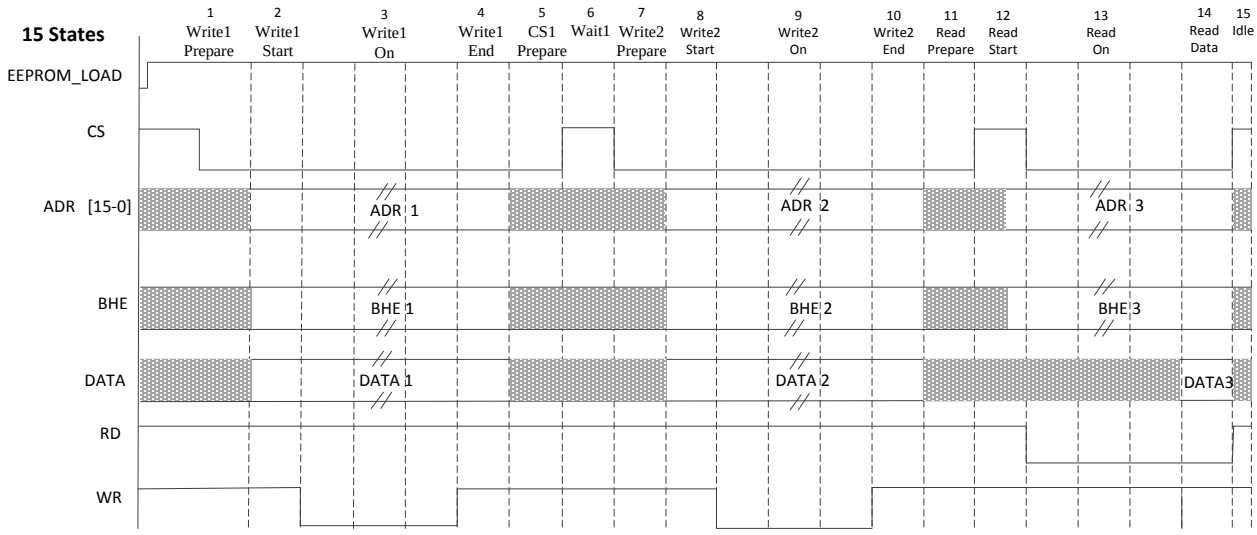
Figure 3. Read access from ESC to FPGA

With respect to the two rules above, typical states VHDL Synchronous network algorithm code (Fig.4 a) for the ESC µController is following:

```
STATES: PROCESS (Current state)
BEGIN
 CASE Current state IS
    WHEN Idle      =>
 TS <= '1'; CS <= '1'; RD_WR <= '1'; BHE <= '1';
next state <= Write1 Start; timer <= 1;
 ……
  WHEN Delay    =>
TS <= '1'; CS <= '1'; RD_WR <= '0'; BHE<= '1';
next state <= Write1 Start; timer <= 1;
END CASE;
END PROCESS STATES;
```



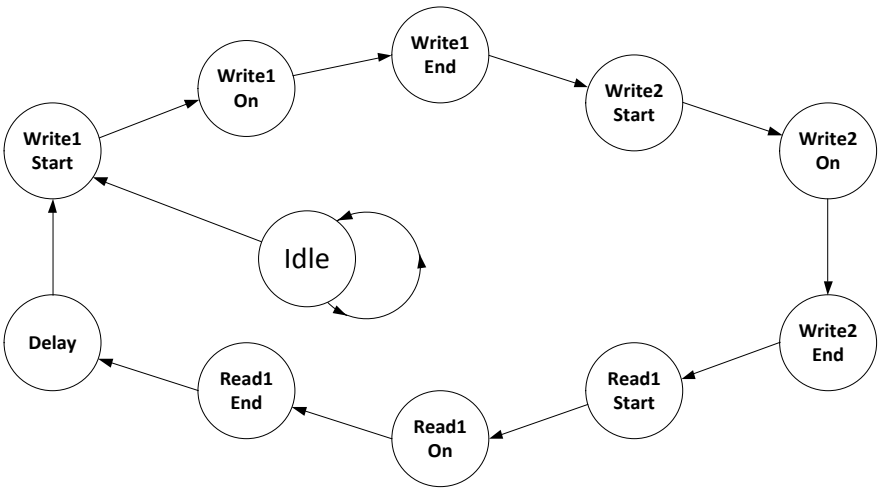
(a) Synchronous network Communication



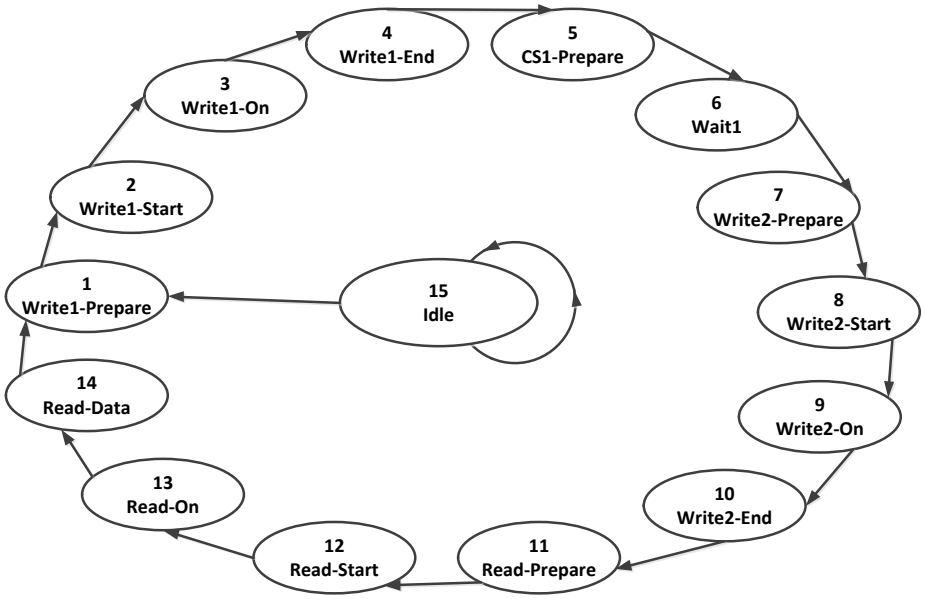
(b) Asynchronous network Communication

Figure 4. FPGA 15 states for the networks sequence of two write accesses and a read access

The state machine code based on VHDL shown above is an effective form of shorthand, which determines the initial state of the system (Idle state), followed by Write1 Start state, Write1 On state… Delay state. The next state will be produced at the end of each state, such as the Idle state: next state <= Write1 Start. The advantage of this design style is that the number of registers is the minimum. The Asynchronous algorithm code is similar to the Synchronous, which does two things: (a) it assigns the output signals and (b) it establishes the next state for each current state. Moreover, it complies with rule 1 and 2, relative to the design of combinational circuits using sequential statements, for all the Synchronous or Asynchronous networks μController ESC signals have been presented and the sequences of the write access and read access have been specified.



(a) Synchronous network Communication algorithm



(b) Asynchronous network Communication algorithm

Figure 5. State machine of FPGA for two write accesses and a read access networks

4 Result

The Synchronous network Communication Simulations of two write accesses and a read access are shown in Fig. 6, Fig. 7, Fig. 8, where Fig.6 (a) corresponds precisely to the Fig. 4 (a) Idle, Write1 Start, Write1 On, Write1 End states; Fig. 7 (a) relates closely to the Fig. 4 (a) Write2 Start, Write2 On, Write2 End states; Fig. 8 (a) matches exactly to the Fig. 4 (a) Read1 Start, Read1 On, Read1 End and Delay states. For the FPGA NIOS project, a 100M clock is provided by PLL, which means a frequency divider must be built for ESC clock. We have implemented a VHDL circuit that the 100M clk was divided by 4 are shown in Fig. 7 (a) and got the 25M CPU CLK IN signal to the Synchronous network.

The Asynchronous network Communication Simulations of two write accesses and a read access are shown in figure 6, figure 7 and figure 8, where figure 6(b) is similar to the figure 4(b)states: 1 Write1-Prepare, 2 Write1-Start, 3Write1-On, 4 Write1-End, 5 CS1-Prepare; Figure 7(b) is equivalent to the figure 4(b) states: 6 Wait1, 7 Write2-Prepare, 8 Write2-Start, 9 Write2-On, 10 Write2-End; Figure 8(b) is almost identical to the figure 4 (b)states : 11 Read-Prepare, 12 Read-Start, 13 Read-On, 14 Read-Data, 15 Idle.

Therefore, two write accesses and a read access between the FPGA Synchronous or Asynchronous network μController and the ESC were realized.

5 Discussion

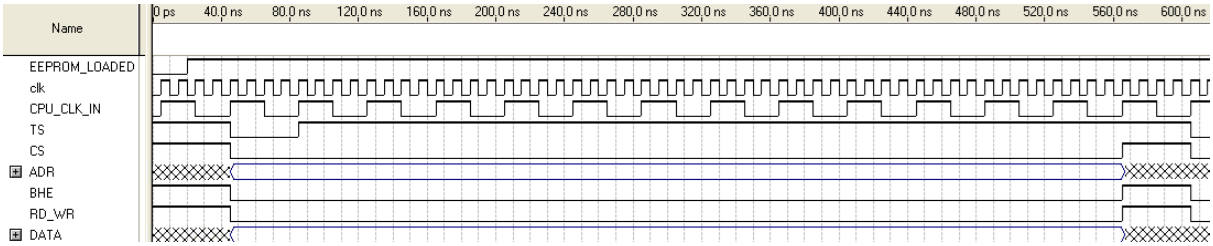
This study investigates the μController Networks Communication for EtherCAT Process Data Interface. The first objective is to describe the structure of the Synchronous and Asynchronous, independent of networks signals, thus making them applicable to parametric architecture. The second objective is to empirically compare our method with general existing SPI IP core communication. The third objective is to experimentally compare the Synchronous and Asynchronous networks sequence, thus confirming that the Synchronous network is suitable for synchronizing by an external clock, while asynchronous network transmission is synchronized by special signals (CS) along the transmission sequence.

Compared with SPI communication, the Synchronous and Asynchronous methods achieve higher speed rate than traditional SPI transmission. Faster speed rate implies that the efficiency of this study, which will be running in a short time. The accuracy and real-time characteristics of networks, which are composed of Synchronous or Asynchronous structure, have the right qualities for the network parameters requirements.

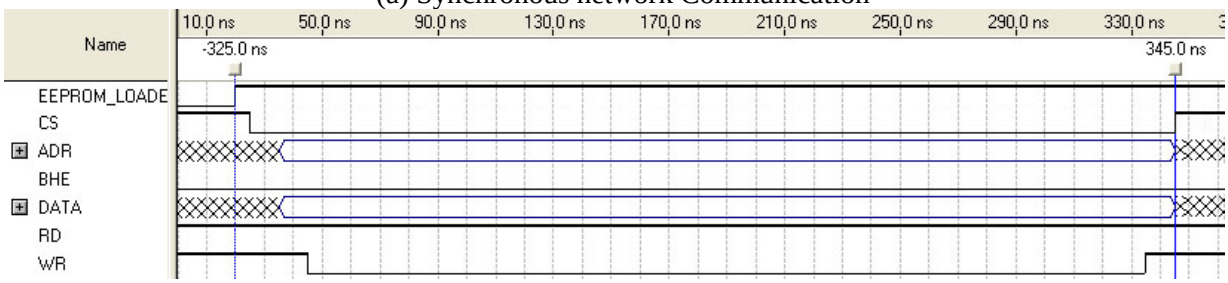
The relation between the Synchronous and Asynchronous networks design, actuation signals and the performance is visualized for each design step by Figs. 2, 3 and 4. However, some general SPI design guidelines for EtherCAT network can be generated with only the SPI 5 signals: SPI_CLK,

SPI_DI (MOSI), SPI_DO (MISO), SPI_SEL and SPI_IRQ. Therefore, the number of the SPI design pins must be fewer than Synchronous and Asynchronous networks.

The main contribution of the proposed design cases is the compared the Synchronous and Asynchronous EtherCAT PDI networks in relation to the write and read performance and design requirements. The visualization of these network relations immediately shows the designer the signals of the network parameters to obtain the desired performance.

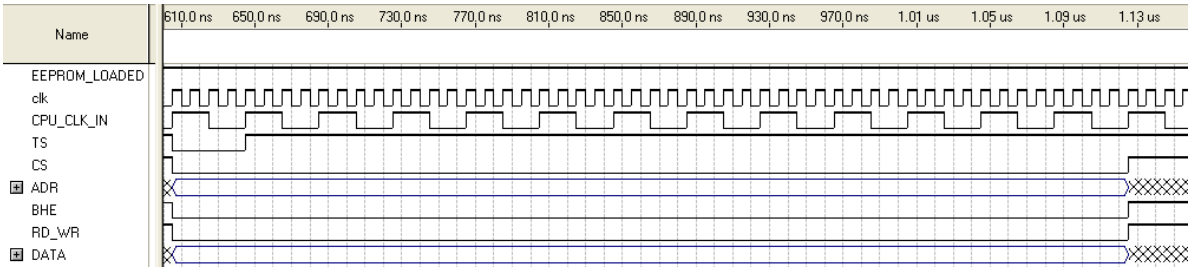


(a) Synchronous network Communication

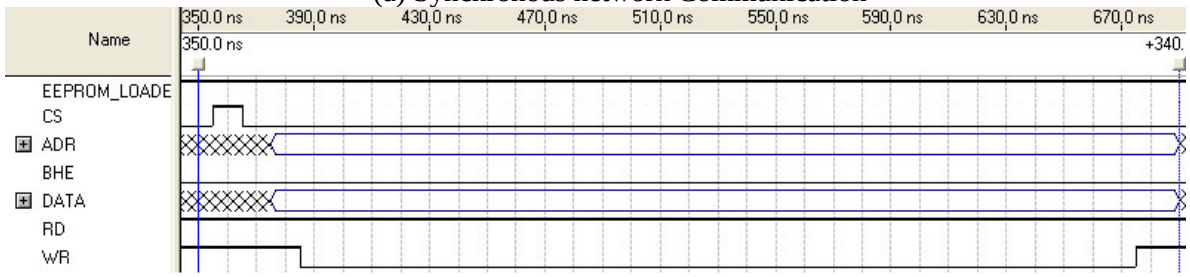


(b) Asynchronous network Communication

Figure 6. Simulations of states machine for the networks sequence of the first write access

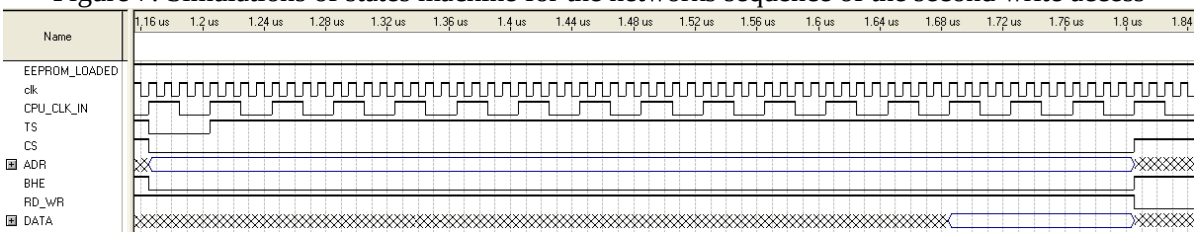


(a) Synchronous network Communication

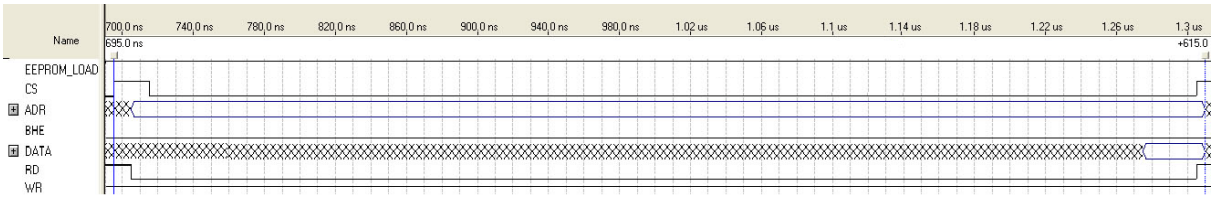


(b) Asynchronous network Communication

Figure 7. Simulations of states machine for the networks sequence of the second write access



(a) Synchronous network Communication



(b) Asynchronous network Communication

Figure 8. Simulations of states machine for the networks sequence of the following Read access

6 Conclusions and Future Work

In this research, we dealt with an analysis Schematic of Synchronous and Asynchronous network µControllers interconnection between FPGA and ESC based on FPGA state machine algorithm. The write access and read access from the FPGA to the ESC was firstly analyzed. The 16 bits Synchronous and Asynchronous network Communication µControllers with the state machine algorithm which are composed of states was secondly proposed. Finally, the performance of the 16 bits Synchronous and Asynchronous network µController signals were verified by means of Simulations.

Future works are mentioned here. The center of the EtherCAT slave application is restricted by process data interface, which relies on Asynchronous, Synchronous µController or SPI communication. It is necessary to consider the real application and compare the Synchronous or Asynchronous µController with the SPI stability in practical networks operation.

Acknowledgment

This work is supported by National Program on Key Basic Research Project of China (973 Program) (No.2011CB013306).

Sincere thanks go to the editor and the reviewers for their careful review and many helpful comments.

References:

[1] Macro Cereia, Ivan Cibrario Bertolotti and Stefano Scanzio, Performance of a real-time EtherCAT master under Linux, *IEEE Transactions on Industrial Informatics*, Vol.7, No.7, 2011, pp.679-686.

[2] Peter G. Whiteisa and Melinda J. Melloa, Development of an EtherCAT enabled digital servo controller for the Green Bank Telescope, *Ground-Based and Airborne Telescopes Iv*, Vol.8444, 2012, pp.84445N1-N9.

[3] S.Vitturi, L. Peretti, L. Seno, et al, Real-time Ethernet networks for motion control, *Computer Standards and Interfaces*, Vol.33, No.5, 2011, pp. 465-476.

[4] Lei Wang, Muguo Li, Jing WANG, Design and performance evaluation for a class of networked motion control system, *International Journal of Advancements in Computing Technology*, Vol.3, No.10, 2011, pp.46-53.

[5] Sung M., Kim K., Jin H. W., et al, An EtherCAT-based motor drive for high precision motion systems, *IEEE International Conference on Industrial Informatics*, Beijing, China, 2010, pp. 163-168.

[6] Yong-Jin Lee, Min-Kyu Park and Seok-Jo Go, Development of smart actuator for leight-weight modular robot, *International Conference on Control, Automation and Systems*, Gyeonggi-do, South Korea, 2010, pp. 674-677.

[7] Sven Gestegard Robertz, Klas Nilsson and Roger Henriksson, Industrial robot motion control with real-time Java and EtherCAT, *12th IEEE International Conference on Emerging Technologies and Factory Automation*, Cracow, Poland, 2007, pp. 1453-1456.

[8] Gianluca Cena, Ivan Cibrario Bertolotti, Stefano Scanzio, et al, Evaluation of EtherCAT distributed clock performance, *IEEE Transactions on Industrial Informatics*, Vol. 8, No.1, 2012, pp.20-29.

[9] Dirk, Holger Buttner, Real-time Ethernet - the EtherCAT solution, *IEE Computing and Control Engineering*, Vol.15, No.1, 2004, pp.16-21.

[10] Zhao Jie, Yu Fang, Tang Minfeng, et al, Design and implementation of EtherCAT-based distributed cotton bale inspection and management system, *Transactions of the Chinese Society of Agricultural Engineering* , Vol.29, No. 2013, pp. 207-216.

[11] Wang Xiangming, Wang Jinchao and Sun Donghua, Application of the real-time EtherCAT technology in wind power control system, *2012 International Mechanical Properties and Structural Materials Conference*, Shenyang, Liaoning, China, August, 2012, pp. 155-159.

[12] Maruyama Tatsuya and Yamada, Tsutomu, Real-time Ethernet acceleration technology and applying to EtherCAT master controller, *SICE 2012*, Japan, pp. 943-947.
[13] EtherCAT Technology Group, Hardware data Sheet ET1100 EtherCAT slave controller, *Beckhoff ETG*, Germany, 2010, pp.61-65.
[14] EtherCAT Technology Group, Application layer protocol specification EtherCAT specification part 6, *Beckhoff ETG*, Germany, 2010, pp.12-17.