Atlantis Highlights in Intelligent Systems (AHIS), volume 1
3rd International Conference on Control, Automation and Artificial Intelligence (CAAI 2018)

FPGA Software Design for the Missile-borne Computer with DSP and FPGA as Its Core

Tao Yang*, Jinpeng Yang, Shuaibing Wang and Ling Li
China Academy of Aerospace Aerodynamics, Beijing, China
*Corresponding author

***Abstract*—In order to meet the requirement of stability, real-time performance and miniaturization, a missile-borne computer with DSP and FPGA as its core processor is designed. FPGA communicates with DSP via EMIF for extension of various peripheral interfaces, including the serial port, AD converter and IO operation. This paper presents the FPGA software design in detail. The software structure is shown and its function modules are analyzed one by one. And also the registers for EMIF are defined and introduced. After several ground tests, the FPGA software works well in the missile-borne computer, indicating that it is feasible and reliable, fulfilling the design demand. The follow-up flight test in the future will verify its performance further.**

Keywords—missile-borne computer; FPGA; EMIF; software design

I. INTRODUCTION

Missile-borne computer is the core of the missile control system [1,2]. According to output of the IMU (Inertial Measurement Unit), it calculates the attitude and position parameters of the missile. Based on the relative position measured by the seeker, it computes the line-of-sight rate. The missile-borne computer sends corresponding control command to the actuator system [3]. In this way, it controls the missile attitude and conducts trajectory correction, leading it to the intended target. As a result, it is performance of the missile-borne computer that determines whether the missile can hit the target [4]. In recent years, with the diversity of fighting environment and warfare requirement, the demand for longer firing range and higher attacking precision is more and more urgent. In the missile-borne computer design, stability, real-time performance and miniaturization are three main focuses [5]. Stability means strong environmental adaptability, namely, counter high overloading ability and anti-electromagnetic interference ability. Real-time performance requires the fastest process and calculation of the received sensor data. Miniaturization calls for smaller size and lighter weight [6].

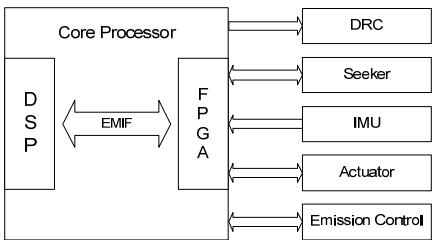

FIGURE I. THE MISSILE-BORN COMPUTER

Our research group develops a missile-borne computer with DSP (Digital Signal Processor) selected as the processor of the missile-borne computer to conduct core navigation algorithm, while FPGA (Field-Programmable Gate Array) extends the various peripheral interfaces [7]. Its block diagram is shown in Figure 1. This paper talks about the FPGA software design. It is divided into various function modules.

II. THE OVERALL FPGA SOFTWARE DESIGN SCHEME

The structure of the FPGA Software is shown in Figure 2. FPGA communicates with DSP via EMIF (External Memory Interface) with 16 bit data bus and 20 bit address bus. The EMIF module is divided into two parts, the transmission module EMIF_TX and the reception module EMIF_RX. The peripherals consist of an ADC, six RS422 serial port interfaces, one IO input and five IO outputs. The voltage of power supply is acquired by the ADC for power monitoring.

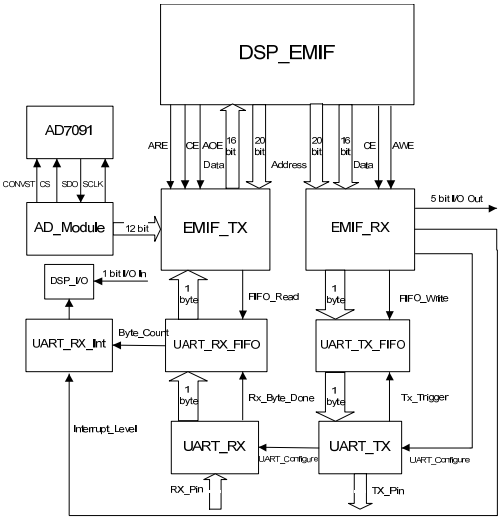

FIGURE II. STRUCTURE OF THE FPGA SOFTWARE

Only one serial interface is illustrated in Figure 1 and the others work under the same mechanism. Once inputted into PFGA, the serial data is transformed into one byte data according to certain data format via the serial reception module UART_RX. And then the pulse signal Rx_Byte_Done is generated. The data is written into the serial reception FIFO (First Input First Output) UART_RX_FIFO with a depth of 128 bytes. When DSP reads the serial data via EMIF, the EMIF_TX module generates the read pulse signal FIFO_Read.

It reads the serial data from UART_RX_FIFO and the data is available later in EMIF data bus. When DSP transmits data via EMIF, the EMIF_RX module generates the write pulse signal FIFO_Write, putting data from the EMIF data bus into the serial transmission FIFO UART_TX_FIFO. Query the status of the transmission FIFO. If there is data in UART_TX__FIFO, transmit it in real time via the serial transmission module UART_TX. What's more, the serial port interface can initiate data reception interrupt. Under the circumstances that the number of data bytes in UART_TX__FIFO matches the interrupt level, the interrupt signal is sent to DSP for interrupt process.

III. THE FUNCTION MODULES

A. EMIF Module

DSP communicates with FPGA via EMIF, the transmission module EMIF_TX and reception module EMIF_RX included. According to certain EMIF addresses, the EMIF transmission module puts data in the bidirectional EMIF data bus for DSP to read, while the EMIF reception module helps to send data from DSP to FPGA for process.

1) The EMIF transmission module

As shown in Figure 3, there are three control signals: CE, AOE and ARE. CE stands for memory space enable (active low), AOE stands for data output from FPGA enable (active low), ARE stands for data read enable (active low). During the transmission process, when the setup period starts, the address becomes valid. CE and AOE drive down from high level. At the beginning of the strobe period, ARE becomes valid. And then ARE drives down in the end of the strobe period. And Data is sampled on the clock rising edge, just prior to the ARE low-to-high transition. CE and AOE become inactive. The setup, strobe and hold time of the DSP EMIF needs to be set properly.

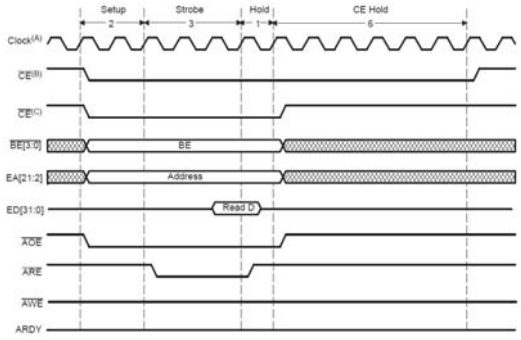

FIGURE III. THE EMIF TRANSMISSION PROCESS

In the EMIF transmission module, there is a driving clock. It takes several clock cycles to conduct data transmission. At first, monitor the level of ARE when every clock rising edge arrives. Once the falling edge of ARE detected, the transmission process is triggered. Judge whether ARE is high or not in the flowing clock rising edge. If ARE is high, the previous falling edge of ARE is interference pulse, stop the transmission process immediately. Otherwise, if ARE is low, the normal transmission process is underway. Wait for the next clock rising edge. When it arrives, the levels of ARE, AOE and CE are low. According to different EMIF addresses, AD data and the number of data bytes in serial reception FIFO are put into EMIF data bus, while generating the read enable signal of the serial reception FIFO, which is driven down in the next clock cycle. And then one byte data is read from serial reception FIFO and appears in EMIF data bus later, thus data transmission completed.

2) The EMIF Reception Module

As shown in Figure 4, there are two control signals, namely, CE and AWE. CE stands for memory space enable (active low) and AWE stands for data write enable (active low). During the reception process, the setup period starts, along with valid EMIF address and EMIF data. At the same time, CE turns low from high. AWE is active at the beginning of the strobe period and inactive at the beginning of the hold period. CE becomes inactive at the end of the hold period. Same as the transmission process, a successful reception process demands appropriate setup, strobe and hold time.

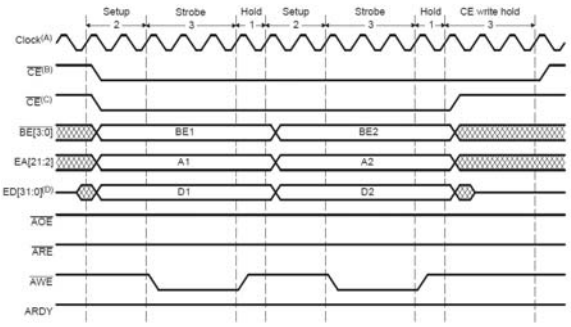

FIGURE IV. THE EMIF RECEPTION PROCESS

A driving clock exists in the EMIF reception module. It also takes several clock cycles to conduct data reception. The AWE level is of much concern in every clock rising edge. Its falling edge initiates the writing process. Whether AWE is still high or not in the following clock cycle determines how to process later. If AWE is high, the previous falling edge is disturbance. There is no need continuing the reception process. On the contrary, if AWE keeps low, it indicates a normal reception process. In the following clock cycle, the levels of AWE and CE are low. According to different EMIF addresses, update the serial transmission FIFO data, configure the serial port hardware, and set the levels of IO output. Later, generate the write enable signal of serial transmission FIFO, which sends the latest serial data into the serial transmission FIFO. And then the reception process is completed.

B. Serial Data FIFO Module

For the simple reason that the FPGA chip selected in the missile-borne computer is from Xilinx company, the serial data FIFO module utilizes its IP (Intellectual Property) core in Figure 5 directly. Set Native as the Interface Type, and choose Independent Clocks as the option of Read/Write Clock Domains, namely, different clocks for read or write. Read Mode is Standard FIFO. The Data Port Parameters is configured with 8 bits of Write and Read Width as well as 256 bytes of Write and Read Depth. Therefore, the FIFO is used in 8-bit width and the buffer is 256 bytes. For the optimization of

process speed, it is unnecessary to set the Reset signal and other flags. According to the FIFO depth, the Write Data Count and Read Data Count is 8 bit.

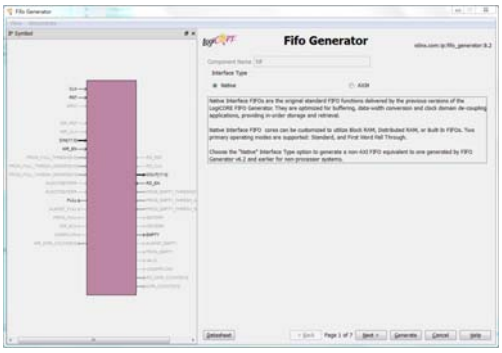

FIGURE V. THE IP CORE OF FIFO

1) The serial transmission FIFO

In the serial transmission FIFO UART_TX_FIFO, system clock is the write and read clock. After one byte data is received, the high-level enable pulse signal is generated for one clock cycle and puts data into UART_TX_FIFO. The rd_data_count increases with it. If the number is more than zero, data in FIFO UART_TX_FIFO is read and transmitted. Completion of transmission generates one high-level pulse signal, in the falling edge of which the rd_data_count is judged again. If it is not zero, the next transmission is initiated. However, if it is zero, the data transmission ends. In this way, data transmission is conducted in real time.

2) The serial reception FIFO

In the serial reception FIFO UART_RX_FIFO, the write and read clock is the system clock. When one byte valid data is received in UART_RX, the high-level pulse rx_byte_done is generated and it lasts for one clock cycle. It serves as the write enable signal of UART_RX_FIFO. As long as there is data input, it goes into UART_RX_FIFO immediately. The read enable signal is generated by the EMIF module and it is a high-level pulse of one clock period as well, which reads the data from UART_RX_FIFO and puts it into the EMIF data bus. Based on the number of data bytes in UART_RX_FIFO, the wr_data_coun and rd_data_count increase and decrease at the same time. rd_data_count is selected as the number indicator. When DSP requests the serial data from UART_RX_FIFO, first get the number of data bytes, and then read the data accordingly. In order to avoid FIFO overflow, the number threshold of data bytes is set as 128. If the number is more than 128, the UART_RX_FIFO moves data out automatically, thus making sure the serial reception FIFO is not overflow with updated data.

C. Serial Data Transmission and Reception Module

The serial data transmission and reception module is driven by the system clock. During initialization, the data character length, parity enable, parity odd/even selection, stop bits and baud rate can be configured via EMIF command.

1) The data transmission module

The count value of data transmission clock is set according to the system frequency and baud rate. Once the transmission enable pulse is detected, the counter begins. When the counter value equals the count value, one high-level pulse signal is generated as the flag to initiate transmission of data bit, parity bit and stop bit. Once the transmission finished, one high-level pulse signal generates for indication. The counters stops and restarts when the next data transmission process begins.

2) The data reception module

On the basis of the system frequency and baud rate, set the count value of data reception clock. Detection of the data start bit in serial data reception module starts the counter. When the counter value matches half of the count value, one high-level pulse signal is generated as the flag to trigger reception of every data bit according to data character length. Following the data bit, the parity bit and stop bit are received, thus accomplishing data reception of one valid byte. One high-level pulse signal indicates the end of data reception. The counter stops and returns to zero, which counts again until the next data reception process is conducted.

D. Other Modules

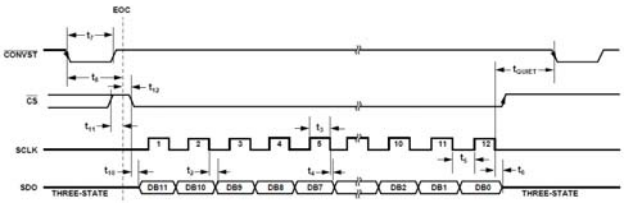

FIGURE VI. BLOCK DIAGRAM OF THE FPGA SOFTWARE

The AD7091 is a successive approximation register analog-to-digital converter, with SPI (Serial Peripheral Interface) as its communication interface, clock frequency of 2.5MHz and data acquisition interval of 100us. As shown in Figure 6, the conversion process and data acquisition are controlled via the CONVST signal. Its falling edge initiates the conversion. The 12 bit output coding of AD7091 is straight binary. The MSB is clocked out first and then the DB10 to DB0 are shifted out one bit by one bit. The AD conversion result appears in the EMIF data bus according to certain EMIF addresses.

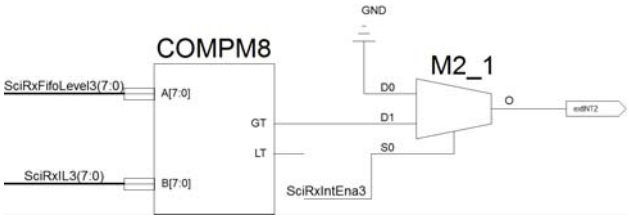

FIGURE VII. BLOCK DIAGRAM OF THE FPGA SOFTWARE

The serial port interfaces has a mechanism of reception interrupt. If enabled, the interrupt is generated according to the interrupt level set previously. As shown in Figure 7, there is a comparator to compare the number of data bytes in UART_RX_FIFO and the interrupt level. When the number is bigger than the interrupt level, the comparator output is high. In other cases, its output is low. There is a multiplexer, whose address is the interrupt enable signal. If the enable signal is high, the comparator output is the multiplexer output. On the contrary, if the enable signal is low, the multiplexer output is

just low level. The multiplexer output connects an external interrupt IO of DSP. Set the interrupt polarity on the rising edge and the serial data reception interrupt is handled.

As for the external IO input, the interrupt mechanism is adopted too. In normal operation, the IO input is high. The high-to-low transition triggers the DSP external interrupt, with the falling edge as the interrupt polarity. In the meanwhile, the IO output is set directly via EMIF by writing its value into one certain register.

IV. THE EMIF REGISTERS

DSP and FPGA communicate with each other via EMIF. In this design, the EMIF address bus is 20 bit and the EMIF data bus is 16 bit. There are six serial port interfaces, one AD converter and one IO input as well as several IO output. The corresponding registers are defined as shown in Table 1. For simplicity, there are registers of only one serial port interface. Descriptions of each register are made from Table 2 to Table 8.

TABLE I. EMIF REGISTERS

No.	Address	Name	Description
1	0x10001	SCICCR1	Serial control register1
2	0x10002	SCITXBUF1	Serial transmission register 1
3	0x10003	SCIRXBUF1	Serial reception register 1
4	0x10004	SCIRXFIFO1	Serial reception data number register 1
5	0x10005	SCIRXIL1	Serial reception interrupt register 1
6	0x70001	ADDATABUF	AD data register
7	0x80001	IOBUF	IO register

TABLE II. SCICCR REGISTER

15-10	9-6	5	4	3	2	1	0
RSRV	SCIBAUD	STOPBITS	PARITY	PARITYENA	SCICHAR		

SCICHAR: These bits select the character length from one to eight bits. PARITYENA: This bit enables or disables the parity function. PARITY: Odd/even parity selection. STOPBITS: Number of stop bits. SCIBAUD: Baud rate selection.

TABLE III. SCITXBUF REGISTER

15-8	7	6	5	4	3	2	1	0
RSRV	D7	D6	D5	D4	D3	D2	D1	D0

SCITXBUF: 8 bit data for serial data transmission.

TABLE IV. SCIRXBUF REGISTER

15-8	7	6	5	4	3	2	1	0
RSRV	D7	D6	D5	D4	D3	D2	D1	D0

SCITXBUF: 8 bit data of the serial reception data.

TABLE V. SCIRXFIL REGISTER

15-9	8	7-0
RSRV	RXIENA	RXIL

RXIL: These bits determine the serial reception interrupt level. RXIENA: This bit enables or disables the serial interrupt function.

TABLE VI. SCIRXFIFO REGISTER

15-8	7	6	5	4	3	2	1	0
RSRV	D7	D6	D5	D4	D3	D2	D1	D0

SCIRXFIFO: 8 bit data indicates the number of data bytes in serial reception FIFO.

TABLE VII. ADDATABUF REGISTER

15-12	11-0
RSRV	D11~D0

ADDATABUF: 12 bit data represents the AD conversion result.

TABLE VIII. IOBUF REGISTER

15-11	4	3	2	1	0
RSRV	Out5	Out4	Out3	Out2	Out1

Out5 ~ Out1: These bits decide the IO output level.

V. CONCLUSION

This paper introduces the FPGA software design for the missile-borne computer with DSP and FPGA as its core. All the details about the PFGA software can be found in this paper. The software helps FPGA to conduct communication between the missile-born computer and other key devices in the missile via the serial port. To test and verify the FPGA software, it has to cooperate with the DSP software. Ground tests prove its feasibility and reliability. It will be tested in missile flight missions later.

REFERENCES

[1] Zhang Wei and Zhou Ying, “Research on design method of missile-borne guidance computer,” in Aeronantical Computing Technique, vol. 39, pp. 123–127, 2009.

[2] Zhang Xiaoxi, Liu Yongqiang and Liu Shuo, “Integrated guided of missile-borne and design of control system,” in Aeronantical Computing Technique, vol. 45, pp. 121–124, 2015.

[3] Gao Min, Ren Hailong, Yang Fang and You Weihua. “Design of missile-borne computer based on DSP+FPGA,” in Computer Measurement and Control, vol. 22, pp. 3995–3997, 2014.

[4] Shan Yanhu, Xie Lu, Yang Yuhua and Wang Dawei, “Design of hardware platform for flight control system based on DSP and FPGA,” in Fire Control and Command Control, vol. 42, pp. 169–173, 2017.

[5] Li Yinhai, Wang Mingang and Sun Chuanxin, “The design of digital flight control computer based on DSP and FPGA,” in Electronic Design Engineering, vol. 22, pp. 21–24, 2014.

[6] Yu Shaolin, Han Bo and Li Ping, “Design of multi-serial communication for flight controlling computer based on FPGA,” in Computer Engineering, vol. 37, pp. 242–245, 2011.

[7] Chen Zhangzhe and Liu Yabin, “Design and implementation of time-sharing measurement and control system based on the FPGA,” in Electronic Design Engineering, vol. 26, pp. 75–78, 2018.