Functional Verification Methodology of Complex Electronics System Based Modeling and Simulation

Jianwu Wu
School of Politics & Law and Public Administration
Hubei University,Wuhan,China
wujianwu@hubu.edu.cn

Abstract—**Nowadays electronics system becomes more and more complex, which causes difficulty to debug. The paper presents a new method of system-level simulation and functional verification to complex electronics system by modeling. A high-speed image real-time storage system is used as an example to explain the application of modeling method in actual system, and each functional module is modeled by Verilog language and simulated in ModelSim platform and verified by Matlab platform. Practice has proved that the above-mentioned method has a good benefit to debugging and functional verifying of complex electronics system.**

Index Terms—**simulation，system-level modeling，storage system， functional verification**

I. Introduction

Nowadays electronics hardware system becomes more and more complex, there are often several microprocessors and programmable logic devices, there is parallel operation and communication mechanism between them, and they can be regarded as a unity of program and time sequence. The gradually progressive debugging strategy is usually adopted for the complex system[1]. The strategy can be used for system testing，but there are also certain limitations，which is embodied in redesigning only the circuit if it is found that the hardware is unable to meet requirement，and thus it will bring the waste of time, effort and financial resources. Based on this point , this paper presents a new system modeling method. The method simulates hardware through system-level modeling，Verilog language is used to model complex hardware system，the stability and the complete function of the system can be verified by Matlab and Modelsim platforms.

II. Modeling Method

The system-level modeling is a common method used in the field of chip design， its purpose is to divide and describe the system function[2]. The debugging method of actual complex hardware system is a subject full of skills. In this paper, the system-level modeling method in the field of chip design is reasonably extended to the debugging of complex hardware system. By adopting system-level modeling and simulation and functional verification at the primary stage of the system design, it can improve the hardware system design to achieve the debugging purpose to prevent from redesigning the circuit.

A. Key Points of Modeling

All key elements of an electronics system may be summarized as follows: data, time, space, function, program and time sequence, etc[3]. Among them, the data are the main part of the system, time and space are the operating area of the data， the function is the representation of data processing in time and space. The program is the executed code driven by clock, and time sequence is the time representation driven by the program . In practice, the program and the time sequence are often driven each other. Essentially, the electronics system is a unity of the above-mentioned key elements，the electronics system modeling is needed to base on these key elements.

B. Implementation Prototype of the Verification Method

The verification method of this paper is achieved through two platforms. Among them，Matlab platform is used as high-level verification platform，ModelSim platform is used as implementation platform for function simulation. The flowchart of the Matlab platform is shown in figure 1. The function of the Matlab platform is to generate the ModelSim's input excitation , simulate data processing and compare with output data. The ModelSim platform processes the ModelSim's input excitation，and stores the output data after simulation. Behavioral modeling of the system is achieved by adopting Verilog language under the Matlab platform[4]. Exchanging information between Modelsim platform and Matlab platform is implemented by file. ModelSim reads the excitation file of Matlab, and writes the result into a file in the simulation process. Matlab reads the simulation result of ModelSim，and compares with its own result，and then judges whether Modelsim has completed the function of the system[5].

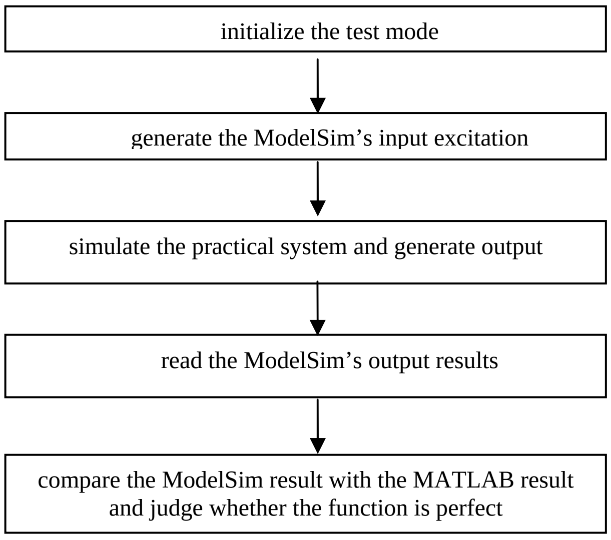
initialize the test mode
generate the ModelSim's input excitation
simulate the practical system and generate output
read the ModelSim's output results
compare the ModelSim result with the MATLAB result and judge whether the function is perfect

Figure 1. The flowchart of Matlab platform

III. Actual System Framework

A. Actual System Model

A high-speed image real-time storage system is used as an example to explain the application of modeling method in actual system. The real-time image storage system often adopts multiplex mode，it is because the image output of high-speed charge coupled devices (CCD) has been divided into multiplex signal[6]. To simplify the model here, one signal of them is surveyed. The single-channel storage flowchart is shown in figure 2. The inside of the dashed border is the storage system and the core, which is called memory card. Among them，the camera controller is the main control unit of the whole system，controlling image storage system throught communication interface. The CCD simulator simulates the actual output of the CCD, which is data source. The first in first out (FIFO) buffer completes the image data cache, enabling the parts of the system to operate in parallel. The microprocessor module controls the addition of the FIFO information through the field programmable gate array (FPGA) module, and receives the reference information and the control signal of the camera controller and returns the system state throught the RS-422 serial port. The FPGA module accomplishs the read/write operation of the FIFO buffer and the operation of direct memory access (DMA). The CPU embedded in the small computer system interface(SCSI*)* protocol controller accomplishs the SCSI bus operation. The SCSI protocol controller contains three interfaces, it transfers image data throught the DMA interface, receives the microprocessor commands throught the microprocessor interface , and interacts with the SCSI hard disk throught the SCSI interface. The SCSI hard disk used as storage medium stores the image data and the reference information.

The system is a high-speed real-time storage system, containing three microprocessors and programmable logic devices, and there are data transmission and asynchronous communication between them. The system debugging is complex, it needs to achieve the SCSI protocol and the data transmission，the key elements of the system and the functional verification can be analyzed by modeling.

B. Modeling Analysis

Combining the modeling method and the actual system flowchart , the modeling for the image storage systems is mainly focused on data flow direction[7], examines data transmission between each module, analyses real-time cost of data transfer, and examines the system reliability during actual operation. Thus， the modeling of the system belongs to data flow oriented system.

From the perspective of system function，the system function module may be divided into internal module and external module with the memory card at the core. The FIFO buffer , the FPGA module, the microprocessor module and the protocol chip inside the memory card belong to internal module, The CCD simulator and the camera controller of the front-end of the memory card and the SCSI hard disk of the back-end of the memory card belong to external modules.

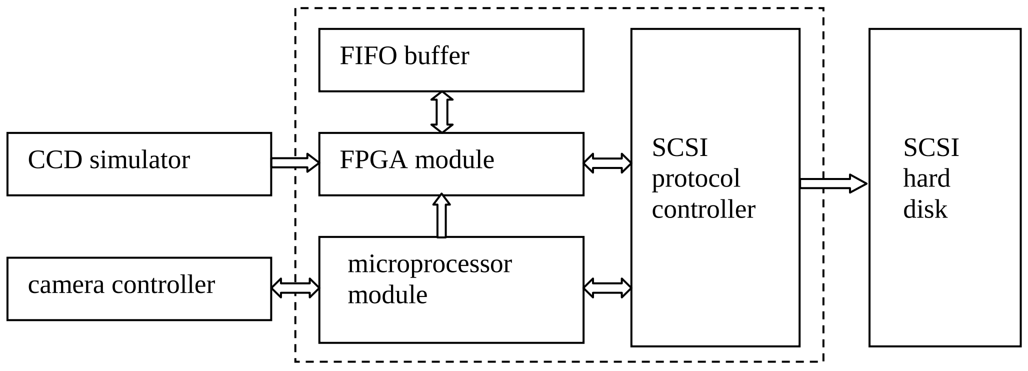

Figure 2. Hardware flowchart of a high-speed image real-time storage system

C. Implementation of the Parts Modeling

The following sections in turn describes the implementation of the parts modeling according to the data flow direction. The front-end of the memory card is the CCD simulator and the camera controller, the back-end of the memory card is the SCSI hard disk. In its middle is the memory card, according to the data flow direction, which are the FIFO buffer, the programmable logical device (PLD), and the microprocessor chip , respectively.

(1) The CCD Simulator Modeling. The CCD simulator modeling is to simulate the output of the CCD , it belongs to behavioral simulation[8], its corresponding file is named "*ccd_sim*". Here , we take DALSA Corporation 's CT-E4 as a example to explain，The module's output ports have pixel clock signal *strobe*, line effective signal *lval*, image data *cdat [7:0]*.

(2) The Camera Controller Modeling. The camera controller is the main control unit of the whole system, its corresponding file is named "*cc_sim*". The module communicates with the memory card through the RS-422 serial port in differential signal way. The camera simulator is modeled because the communication mode between the camera controller and the microprocessor of the memory card is asynchronous mode, so it is needed to examine the reliability of the communication between them. The port definition of the module is as follows.

```
module cc_sim(rst_n, txen, txdat, txclk, ccdok_n, huafu_n );
```

The following subsection divides the system state. The system is general divided into four states: *reset, init, tx,* and rx according to software design method of the camera controller and combining with the purpose of the system modeling.

To simplify the work , the RS-422 serial port is converted into parallel port when modeling，adopting the parallel port similar to serial peripheral interface (SPI) is to facilitate the communication between the simulation part of the microprocessor and it. The following section gives the design method of the parallel port. The data width is 8 bits *txdat[7:0]*，using *txclk* as transmission clock and *txen* as transmission-enabling signal. Among them, the rising edge of *txclk* is in the middle of *txdat*; and *txen* remains in effect during data transfer. Considering that serial baud rate is 187.5 Kb/S and the number of serial digit is 11 bits，so the corresponding parallel baud rate is 17 Kb/S kbps. Considering the overhead of serial interrupt handling, etc，the parallel baud rate is designed to be 15.625 Kb/S.

(3) The FIFO Buffer Modeling. The FIFO buffer modeling also belongs to behavioral modeling， its corresponding file is named *"fifo_sim*". The module mainly accomplishs read-write operation, so we may ignore the other settings and flag bits. The modeling includes the read-write operation , the storage of the FIFO buffer modeling and the operation of merging byte into word. The port definition of the module is as follows

```
module fifo_sim(rst_n, wclk, wen_n, wdat, ren_n, rclk, rdat);
```

The following section explains the modeling method of the above-mentioned three parts , respectively.

i. The Read-Write Operation. The read-write operation of the FIFO buffer adopts synchronous mode and sets enabling bit. Take the read operation as an example , the realization of the read pointer is explained，the read pointer is setted up through the counter, and using asynchronous reset signal and synchronous read enable signal.

```
    //read process
      always@(posedge rclk or negedge fiforst_n_now or
negedge rst_n)
      begin
         if(!fiforst_n_now|(!rst_n))
            rpt <=18'h3ffff; //-1
         else
            begin
            if(!ren_n)
            rpt  <= rpt +1;
            else
            rpt  <=rpt;
         end
       end
```

ii. The Storage of the FIFO Buffer modeling .

According to the explanation of IDT7123，the FIFO depth is 256 K words(1 word=16 bits). The definition of the storage of the FIFO buffer is as follows.

```
reg [15:0] FifoMem[0:262143]
```

iii. The Operation of Merging Byte Into Word. The FIFO’s input signal is 8-bit output data of the camera controller, and its output signal is 16-bit data, so it is needed to implement the operation of merging byte into word. Its implementing method is to use the lowest bit of the write counter to distinguish parity byte and latch the corresponding parity byte, separately, and then merge them by the write clock. Its implementing code is as follows.

```
always@(posedge wclk_drv)//get byte
      begin
        if (wpt[0]) begin wbuf1 <=wdat; end
          else  begin wbuf0 <=wdat; end
      end
      always@(negedge wclk) //byte ->word, just for debug
      begin
        if (wpt[0]) wbuf <={wbuf1, wbuf0};
      end
```

(4) The FPGA Modeling. The FPGA is a synthesizable module in the system, its corresponding file is named *"pld-syn* " . The FPGA modeling includes the direct memory access(DMA) controller, interrupt signal generation and FIFO signal processing, etc. Among them, the synthesizable module of the protocol chip is simplified , so it is not taken as part of the synthesizable module(refer to the section of the microprocessor modeling). The port definition of the module is as follows.

```
    module pld_syn(rst_n, dreq, dwr_n, dmaclk, dack_n,
dmadb,
          lval, strobe, cdat, frame,
          fifoen, fifowrclk, fifodat,
          wclk, wen_n, wdat, ren_n, rclk, rdat);
```

*(5) The Microprocessor Module Modeling.*The microprocessor module modeling is behavioral simulation to the microprocessor，its corresponding file is named "mcu_sim". Its main function includes serial communication function(parallelizing), frame interrupt

signal processing and protocol chip operation, etc[9]. The microprocessor's model is state machine model, it transforms state based on the program flow chart, and the synchronization state machine is used as the main method of the system modeling. According to the division of the system function , we design the following states, the definition of the state machine is as follows:

```
reg [3:0] cs, ns; //define state machine
parameter [3:0]
reset  =4'b0000,
init   =4'b0001,   //init system
wait1  =4'b0011,   //wait for CCCOK
faswr  =4'b0010,   //write reg of fas
fswr   =4'b0110,   //file system wirte
fsrd   =4'b0111,   //file system read
dmawr =4'b1000,   //dma write
fasrd  =4'b1101,   //read reg of fas
rx422  =4'b1001,  //rx422
tx422  =4'b1010,  //tx422
fifowr =4'b1011,  //fifowr
 exint  =4'b1100;  //exint
```

Where , *reset* is the reset state, *init* is the initialization state and *wait1* is the waiting state, waiting until CCDOK is effective，*faswr* is the writing of the DMA command, *fasrd* is the interpretation of the DMA state.This process is often considered to complete successfully, and it can be simulated through the delay，*fswr* and *fsrd* are the writing state and read-write state of the file system， respectively, *exint* is the processing state of the frame interrupt signal, and *rx422* is the processing state of the RS-422 receiving interrupt. The above-mentioned two states are asynchronous mode, in addition to *reset* and *init* states , the other states might also appear .

Among them， the implementation of the serial communication function has been explained in the camera controller modeling， here we will not go further on this issue . According to the principle of simplifying system, the operation of the protocol chip is mainly to control the beginning of the DMA, and thus using the *dmawr_begin* as the control signal to control the DMA start-up. The port definition of the module is as follows.

module mcu_sim(rst_n, rxen, rxdat, rxclk, ccdok_n, huafu_n, dmawr_begin, dmawr_over, frame, fifoen, fifowrclk, fifodat);

(6) The Protocol Chip Modeling. The protocol chip modeling is the behavioral description to the protocol chip[10]，its corresponding file is named *" fas_sim "*. The part modeling mainly implements the drive of the DMA controller and the interface of the SCSI hard disk and the protocol chip. According to the microprocessor modeling method，*dmawr_begin* is the start signal of the DMA, and *dmawr_over* is the reset signal of the DMA. The *dmaclk*，*dreq* and *dwr_n* of the implementing part of the DMA are driven by the protocol chip module. In addition, a parallel interface was provided for the SCSI hard disk, which are write clock *scsiwr*, write enable *scsien* and write data *scsidb*, respectively. The port definition of the module is as follows .

module fas_sim(rst_n, dmawr_begin, dreq, dack_n, dwr_n, dmawr_over, dmaclk, dmdb, scsidb, scsiwr，scscien);

(7) The SCSI Hard Disk Modeling .The SCSI hard disk modeling is behavioral description to the SCSI hard disk，its data come from the protocol chip module. Here we use the file *"IO"* to simulate the writing of the SCSI hard drive and use the hard disk space as the data storage space. The port definition of the module is as follows.

module hd_sim(rst_n, dmawr_begin, scsidb, scsiwr)

D. The System Simulationr Results

For this system, the microprocessor is the core. For the microprocessor, the serial information of the camera controller and the frame interrupt signal are nonsynchronous signal. Ideally, the camera controller sends the reference information to the image memory card, the image memory card temporarily stores them after receiving them ,and then writes them into the FIFO until the external interrupt signal arrives, and thus writes them into the SCSI hard drive. For common image data, there have always been the buffer process and the DMA writing process. The FIFO writing and the DMA transfer process are illustrated in Figure 3. Among them，the writing signal of this figure comes from *ccd_sim*， *dma_begin* controls the beginning of the DMA transmission process, *dma_over* indicates the end of the DMA operation.

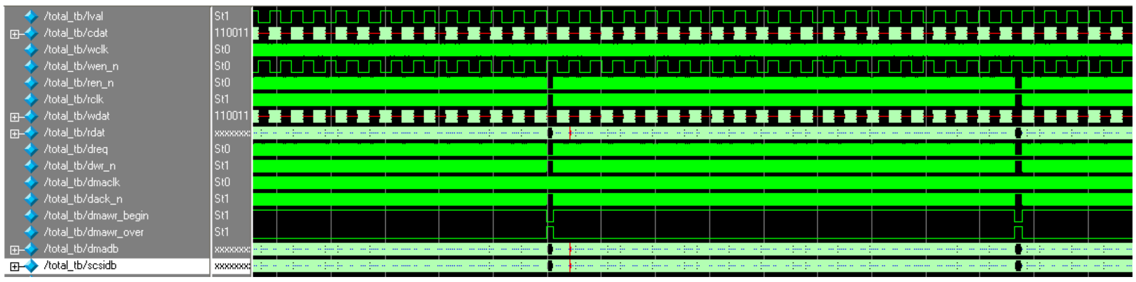

Figure 3. The FIFO writing and the DMA transfer process

The writing process from the microprocessor to the FIFO is illustrated in Figure 4. The microprocessor sets *fiforst_n* effective after the frame interrupt signal is effective , thus the FIFO is reset.Then the microprocessor drives *FIFOEN, FIFOCLK* and *FIFODAT* and writes the reference information into the FIFO.

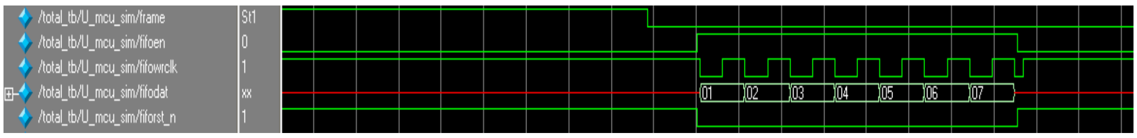
/total_tb/U_mcu_sim/frame
St1
/total_tb/U_mcu_sim/fifoen
0
/total_tb/U_mcu_sim/fifowrclk
1
/total_tb/U_mcu_sim/fifodat
xx
/total_tb/U_mcu_sim/fiforst_n
1
01
02
03
04
05
06
07

Figure 4. The writing process from the microprocessor to the FIFO

The communication between the camera controller and the microprocessor's RS-422 port is illustrated in Figure 5. *RXEN* is the enable signal, *RXCLK* is the clock signal and *RXDAT* is the data signal in Figure .5. Throught the simplified parallel communication mode, the RS-422 serial communication is modeled very well.

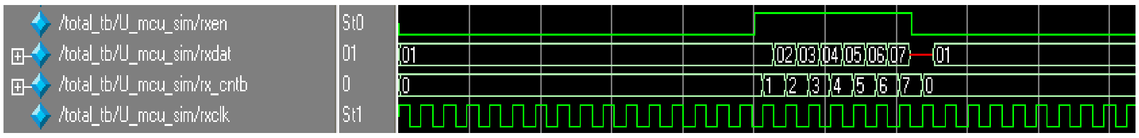
/total_tb/U_mcu_sim/rxen
St0
/total_tb/U_mcu_sim/rxdat
01
/total_tb/U_mcu_sim/rx_cntb
0
/total_tb/U_mcu_sim/rxclk
St1
01
02
03
04
05
06
07
01
0
1
2
3
4
5
6
7
0

Figure 5. The communication between the camera controller and the microprocessor's RS-422 port

The MatLab program mainly involves reading and writing files and data processing , we can see whether the system function is perfect by comparing the final generated files .

IV. Conclusions

The paper proposes a functional verification method of system-level simulation to the complex electronics system by modeling. First , the modeling method is analyzed , and then a high-speed image real-time storage system is used as an example to explain the application of the modeling method in the actual system. Finally, each module has been verified throught ModelSim and Matlab platforms. Practice has proved that the operation reliability and the function completeness of the sytem can be validated very well by using the method, thus the new functional verification method for debugging complex electronics system can be proposed.

References

[1] Zhao Guangzhou, Zhang Tianxu, Wang Yuehuan. High performance reconfigurable hardware system for real-time image processing. Journal of Systems Engineering and Electronics, 2005, 03

[2] Fernandes JM, Machado RJ. System-level object-orientation in the specification and validation of embedded systems. 14th Symposium on Integrated Circuits and System Design , IEEE Computer Society Press, 2001

[3] Grady Booch, James Rumbaugh, Ivar Jacobson. The Unified Modeling Language User Guide Second Edition. Addison Wesley Professional Press, 2005

[4] Xia Wenyu, etc. Verilog Digital System Design Course. Beijing University of Aeronautics and Astronautics Press, 2003

[5] Xue Dingyu, Chen Yangquan. System Simulation Technology and Application based on matlab/Simulink[M]. Beijing: Tsinghua University Press, 2002.

[6] Da Xuanfu, Zhang Boheng. A Method of High Speed Image Data Storage. Acta Photonica Sinica ,2003 , 32 (11) :1393-1395 .

[7] Huang Jin, Guo Li-hong. High speed digital video sampling and storing in optoelectronic theodolite. Optics and Precision Engineering,Vol.12 No.4 2004

[8] Wen- Hsin Chan. CCD camera design and ASIC applications[C] . Proceedings of the Eighth Annual IEEE International Conference , 2003 .

[9] Zhengyu Gu, Zhiyi Yu etc. Functional verification methodology of a 32-bit RISC microprocessor[C]. Proceedings of ICCCAS, 2002, 1454～1457

[10] SCSI Scripts Processors Programming Guide [M]. LSI logic Corporation ,2000.