Design and confirmation of a CAN-bus controller model with simple user interface

Yisu Lv
Key Laboratory for Sensor
Beijing Information Science and Technology University
100101,Beijing,China
lvyisu123@qq.com

Wenjie Tian,Shujuan Yin
Key Laboratory for Sensor
Beijing Information Science and Technology University
100101,Beijing,China
twjkm2008@163.com, yinsj03@163.com

***Abstract*—CAN-bus occupies a very important position in the bus field, which has the advantages of high rate, strong resistance to electronic interference, automatic error detection, etc. This article describes a CAN-bus controller model and the user interface which is used to simplify the operation of model. The controller model is full compliance with the definition of "CAN Specification V2.0", and in order to facilitate the use of controller model, we design a user interface outside of the model. The functions of the controller model were simulated by Modelsim and consistent with the protocol**

Keywords-verification; Verification model; CAN bus; verilog

I. INTRODUCTION

Since the CAN-bus communication protocol was designed, its high performance and reliability have been widely recognized [1]. CAN-bus has been used in various fields of industrial automation, marine, medical equipment, industrial equipment, etc. Electronic control unit based on CAN-bus has become a popular research direction, this paper proposes a CAN-bus controller model described by Verilog HDL, which can fully realize bus communication and data processing functions with all types of frames of "CAN Specification V2.0"[2][3]. The controller model can be integrated into a SOC system as an IP core with slightly modified. To facilitate the users to configure and use the controller model, we specially design a user interface that make controller model as a standard validation model to functionally validate other CAN-bus designs by adding the functions of filling constrained test vectors, error injection, automated testing and so on. These functions can validate measured target's capacity of handling various boundary conditions.

II. DESIGN OF CAN-BUS CONTROLLER MODEL

The controller model library based on CAN-bus protocol fully support the frame of standard format and the extended format communication. CAN-bus controller model library is designed by the method of finite state machine. The controller model library is divided into 3 submodules according to the structure: bit timing logic, top protocol logic and user interface. In order to reduce the complexity and improve the reuseability of design, the controller model also uses the modular design method. The constitution and relationship of each submodule of CAN-bus controller model library is shown in Figure 1.

The synchronization and sampling functions that are employed to receive and send data on the bus are realized by the bit timing logic submodule. CAN-bus protocol logic is processed by an independent unit named Top protocol logic; the user interface is responsible for getting test instructions and sending output information. Test instructions are translated into test operations in the user interface, and the test operations are processed by a series of hierarchical submodules of top protocol logic and bit timing logic to generate test frame on the bus, finally, after the test operation, the test results are sent by the user interface.

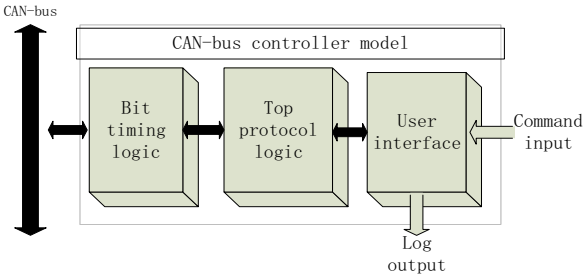

Figure 1. Structure of CAN bus model base

A. Bit timing logic

CAN bus protocol uses nonreturn to zero code as the way to communicate. As the nonreturn to zero code doesn't have additional synchronization information in communicating, CAN bus protocol uses bit timing logic to reduce the deviation caused by the clock frequency error and the phase delay on the transmission path. Figure 2 shows the state machine of bit timing logic [4].

National Natural Science Foundation of China (60968001), Beijing Natural Science Foundation (4122030), Beijing natural science foundation of key projects (B) (KZ201511232037).

978-1-4673-7723-2/15 $31.00 © 2015 IEEE
DOI 10.1109/IMCCC.2015.140

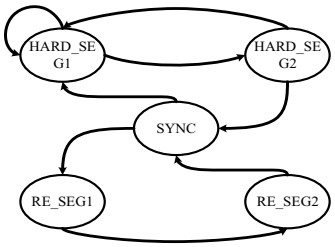

Figure 2. State machine of bit timing logic

The state of PHASE BUFFER SEGMENT1 and PHASE BUFFER SEGMENT2 works in hard synchronization is to synchronize the node itself with other nodes on bus when receiving Start of Frame. Affected by the external disturbance or other electrical factors, CAN-bus controller model needs to compare its bit time with that of other nodes on bus, bit timing logic adjust PHASE BUFFER SEGMENT1 and PHASE BUFFER SEGMENT2 of resynchronization, so that the sending and receiving bit time of controller model can be stayed in same. As shown in Figure 3, resynchronization is processed by resynchronization management module according to the state machine. Sampling module is responsible for data reception, the data needed to output will accord with the state of state machine to be sent to the bus. Bit error injection function is realized by bit timing logic. Bit error injection logic module execute the bit error injection command and send data to bit error injection bus interface. Bit error injection bus interface which is responsible for sending the bit error data to interfere data on bus works as an independent output interface shown in Figure 3.

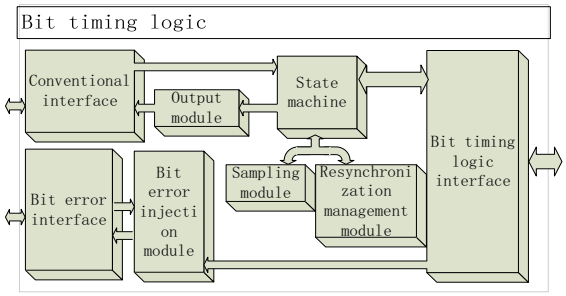

Figure 3. Bit timing logic structure diagram

B. *Top protocol logic*

The top protocol logic is used to process upper-layer protocol, which is responsible for managing the data sending and receiving from bit timing logic, including that sending and receiving frame, processing overload and error information, management of intermission, etc. Top protocol logic also executes error injection according to user interface's command to make sure the measured target's detectability of specified error that sent by the controller model. Figure 4 shows the top-level protocol logic state machine diagram, top protocol logic is the core of processing CAN-bus protocol. Top protocol logic's processing flow can be divided into starting state, idle state, selecting state, sending state, receiving state, overload state, intermission state and suspend state, and each state consists of many sub states in order to process internal data of each main state.

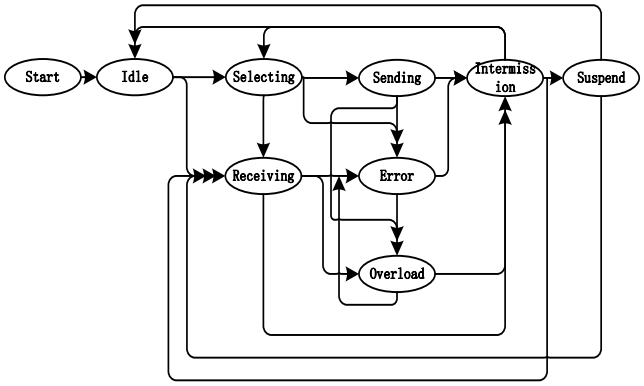

Figure 4. State machine of top protocol logic

In accordance with the CAN bus protocol, the controller model starts at starting state to complete the communication preparation, then enters into idle state and accords to the user interface's command and configuration information to decide to enter select state or receiving state. The frames need selecting state and sending state to complete data transmission, in the selecting state and sending state the controller model will complete transmission, arbitration, CRC calculation and acknowledgement during the sending process and then enter intermission state or suspend state, if the model's node error types is error passive, when the sending is successfully completed then the model jumps into suspend state. In the receiving state, the controller model is responsible for performing filtering, CRC calculating and storage of received information. If received information meet the receive identifier and the CRC sequence is correct, then controller model will answer this frame in response field. When the controller model transmits information, if error occurs, the state will jump into error state [5]. Verification model is responsible for error frames send and monitor in error state. If the last bit of data frame, overload frame and error frame or the first two bits of intermittent field is dominant bit, controller model will jump into overload state and monitor the overload frame. When data frame, overload frame or error frame are transmitted completely, controller model will jump into the state of intermission and read user interface information so that complete the next operation, thus the controller model can complete the test process automatically [6]. As shown in Figure 5, each module is responsible for each function, state machine module send control signals to coordinate each module work together.

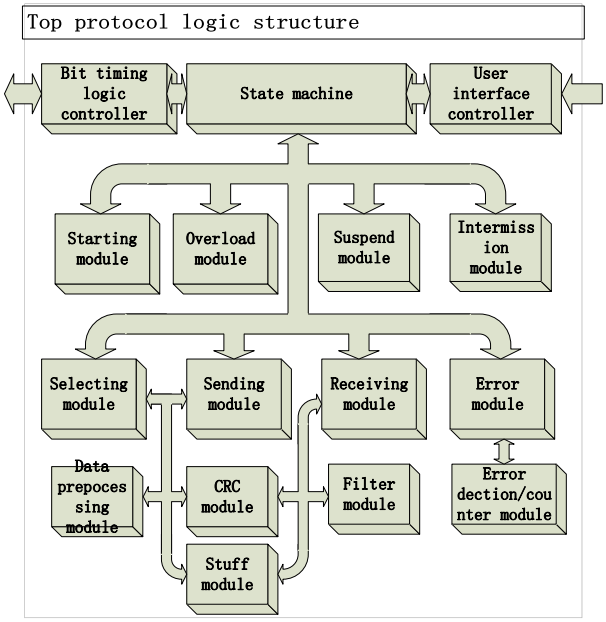

Figure 5. Top protocol logic structure

C. User interface

As the standard protocol model, CAN-bus model library not only works as an independent bus controller but also validates the functional correctness of measured target (other CAN-bus designs). CAN-bus model library execute the test communication with measured target by frames, then processes the data and reports the capacity of handling conventional condition and boundary condition of the measured target.

We prepare the user interface to facilitate users invoke the controller model, so that users are able to control the internal complex operations with some simple interface tasks. User interface execute the work of test command input and test result output, in which the input work is composed of many tasks according to different needs, each task needs respective input parameters to fill. User interface links with top protocol logic and bit timing logic to transmit command. Finally, Controller model's test results will be output by user interface in the form of text document.

Figure 6 shows the test system based on CAN-bus controller model, test system connects measured target with verification model by bus, the controller model send test frame according to the entered test instructions by user interface [7].

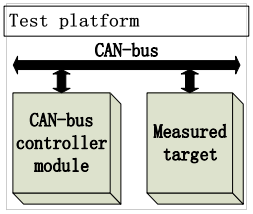

Figure 6. Verification environment based on CAN-bus controller model

a) User interface command

The user interface can be mainly functionally divided into two parts, the correct frame interface and the special frame interface. The correct frame interface can complete transmission of standard data frame, standard remote frame, extended data frame and extended remote frame. Users need fill correct frame interface only with information of ID field, R0 field, R1 field, DLC field and data field, other information which need to transmit in a standard form will be automatically filled into the top protocol logic, this approach can greatly simplify the testing effort. The error handling or other special handling capacity tests of measured target are completed by the special frame interface. Special frame interface is responsible for writing the data frame, special remote frame, error frame, overload frame, intermittent field and the function of error injection into the top protocol logic, and the special frame interface provides the setting of every bit of frame, so that the test coverage will be improved.

b) Test log

User interface records the whole process of the communication between verification model and measured target. After test vectors are executed, the user interface reads information from top layer protocol logic and organizes the result information into test log in a fixed format. The output includes:

Bus state information. This output indicates whether the controller model's error state is "bus-on" or "bus-off". Controller model in "bus-on" state can communicate with other nodes on the CAN-bus. If in "bus-off" state, controller model is removed from the bus and will wait several cycles to return to "bus-on" state.

Error status information. This output indicates whether the transmission results of all vectors are consistent with CAN-bus protocol.

Test vectors send completed state information. This shows whether the test vectors are transmitted successfully, if it fails, the error position information will be displayed.

Information of sending or receiving error counter. This shows the value calculated by error counter, that is convenient for error information's management;

Data information. The information of ID field, R0 field, R1 field, DLC field and data field that are successfully transmitted or received through frames on bus.

The final result information. According to the transmitted and received information, verification model summarizes a comprehensive result.

III. CAN BUS CONTROLLER MODEL'S CORRECTNESS VERIFICATION

According to the requirements of CAN-bus protocol testing standard "Road vehicles - Controller area network (CAN) - Conformance test plan", CAN-bus verification model passes all three types and seven sub-classes verification vectors. The three types are Received frame type, Transmitted frame type and Bi-directional frame type. Each type contains seven sub-classes, which are valid frame

format class, error detection class, error frame management class, overload frame management class, passive error state class, error counter management class and bit timing class. All standard tests ensure the functional correctness of the CAN-bus controller model library.

IV. EXAMPLES OF VERIFICATION

This section lists a test example, we establish an example verification platform constituted by CAN-bus verification model and measured target, the platform can be run in the Modelsim environment. Example of the Testing process shown in Figure 7 and Figure 8, the figures show the test vector transmitting and the result output respectively. The waveform simulation results shown in Figure 7, Figure 8 shows the test output log results.

Figure 7. Waveform of simulation results

```
O_BSR=0, bus state:on line
O_BSR=0, error state:correct
O_TCS=1, transfer state:transfer finished
O_ECC=00000000  error type:bit error,  error direction:receive error,  error position:NO ERROR
O_RXERR=    0, receive error amount
O_TXERR=    0, transfer error amount

receive node info:
O_BSR=0, bus state:on line
O_BSR=0, error state:correct
O_TCS=0, transfer state:transfer unfinished
O_ECC=00000000  error type:bit error,  error direction:receive error,  error position:NO ERROR
O_RXERR=    0, receive error amount
O_TXERR=    0, transfer error amount

transfer node transfered standard data frame
ID=00000000010, R0=0, DLC=0000

receive node received standard data frame
ID=00000000010, R0=0, DLC=0000

test result:pass!
```

Figure 8. Test log of the simulation results

The waveform in Figure 7 shows the procedure of CAN-bus verification model communicate with measured target on bus. “rt” is the virtual signal of CAN-bus, the datas carried by frame communicate on the “rt” signal. If the frame is successfully transmitted or received, the verification model will mark the state of frame successful sent or received respectively. The data transmitted on bus by sending node and recorded in test log by receiving node. The test log contains the test information as mentioned above, with these results we can analyze of the successful completion of the transmission of a frame.

V. CONCLUSION

This paper presents a solution of CAN-bus verification model based on Verilog HDL. This model not only completes the correct frame transceiver, CAN bus protocol error detection, error frame management, overload frame management, negative error state processing, error counting management and bit timing logic and other protocol functions, but also supports error injection, which can be used to test measured target’s error handling capacity.

ACKNOWLEDGMENT

This work was supported by National Natural Science Foundation of China (60968001), Beijing Natural Science Foundation (4122030), Beijing natural science foundation of key projects (B) (KZ201511232037), Key Laboratory of Modern Measurement & Control Technology (Beijing Information Science & Technology University), Ministry of Education

REFERENCES

[1] Q. M.Lai, Taught You How To Learn Can Bus, Beihang University Press, 2010

[2] C.P.Szydlowski, CAN Specification 2.0: Protocol and Implementations . Society of Automotive Engineers, 1992.

[3] R.Bosch, CAN Specification Version 2.0. Bosch, Sep, 1991.

[4] L.Xu, Sh.J.Duan and Ch.N.Li, "Design and confirmation of CAN bus controller based on Verilog HDL", Modern Electronics Technique, vol.35(10),pp. 43-46, 2012.

[5] L.Lei and X.G.Gu, "Analyzing and Implementing of State Machine in CAN Controller", Automatic Measurement and Control, vol.27(3),pp. 82-84,2008.

[6] Sh.G.Dai,M.Gan and Zh.W.Song, "CAN-Bus Fault Analyzer Based on FPGA", Journal of Jilin University (Information Science Edition), vol. 30(5),pp.503-509,2012.

[7] L.Dong, S.Zhang and Z.G.Yu. "Research and Application of an Efficient Testbench Based on Verilog", Microelectronics and Computer, vol. 23(1),pp.55-58,2006.