A Flexible Hardware Architecture for Slave Device of I2C Bus

Cang Liu
National Key Laboratory of Science and Technology on Vessel Integrated Power System,
Naval University of Engineering,
Wuhan 430033, China
liucang@nudt.edu.cn

Xianqiang Bao
National Key Laboratory of Science and Technology on Vessel Integrated Power System,
Naval University of Engineering,
Wuhan 430033, China
baoxianqiang@nudt.edu.cn

Qingyun Meng*
National Key Laboratory of Science and Technology on Vessel Integrated Power System,
Naval University of Engineering,
Wuhan 430033, China
meng_igbt@163.com

Chengcheng Xu
National Key Laboratory of Science and Technology on Vessel Integrated Power System,
Naval University of Engineering,
Wuhan 430033, China
xuchengcheng@nudt.edu.cn

Tao Liao
National Key Laboratory of Science and Technology on Vessel Integrated Power System,
Naval University of Engineering,
Wuhan 430033, China
252489009@qq.com

***Abstract*—I2C module is one of the most vital components in lots of chips. A flexible hardware architecture for slave device of I2C protocol is designed in this paper. The designed hardware architecture is divided into two levels: protocol level and application level. The protocol level implements the basic operations of I2C protocol, and the application level, which bases on the protocol level aiming to the demand of various applications. When complete the implementation and simulation of the designed hardware architecture, a verification platform is built to verify the performance of the designed hardware architecture.**

Keywords- I2C module; Hardware architecture; Protocol level; Application level.

I. INTRODUCTION

Inter-integrated circuit (I2C) protocol is a common communication protocol. It is very suitable for communications between on-board peripherals to transfer low/medium speed data. The I2C module is widely used in various controllers, sensors and some other integrated circuits [1–4]. There is no central server to resolve the data conflicts, they are resolved by the wired-and configurations of the serial data (SDA) signal and serial clock (SCL) signal. Moreover, the acknowledgment signal would be sent by the receiver when every byte is transfered, which can prevent the data loss of the SDA signal.

The I2C protocol only defines the time sequence of writing or reading one or several datas. However, the I2C module of a specific design must define the more detailed protocol to provide the meaning of every byte. Therefore, the flexible hardware architecture of I2C is urgently needed to meet various demands. A generic design on the FPGA platform is presented in [5], the entire design has been verified by Quartus II 6.0. An I2C master controller is implemented in [6], the designed I2C master controller is interfaced with DS1307 provided by MAXIM, which acts as a slave. This design is also verified by FPGA. An I2C protocol design method is presented in [7], the design is divided into protocol level, signal level and interface level in the proposed method. FPGA is used to verify the performance of the designed hardware architecture in [7]. An I2C master controller is designed and implemented in [8]. Some practical hardware architectures have been presented in the literatures. However, most of the existing works only can be used in a fixed scenario described in the corresponding paper.

A flexible hardware architecture is designed for slave device of I2C protocol in this paper. The design is divided into two levels: protocol level and application level. The protocol level provides the signal and state defined by basic I2C protocol. And the application level implements the needed function of the corresponding application based on the protocol level. Because the protocol level can meet various

978-1-7281-4076-6/19/$31.00 ©2019 IEEE
DOI 10.1109/EEI48997.2019.00074

needs, the designed hardware architecture, which can easily be modified to meet other needs is more flexible than the existing works. The designed hardware architecture is verified by FPGA in this paper.

The rest of this paper is organized as the follows. In Section II, we describe the I2C protocol. In Section III, the designed hardware architecture for slave device of I2C protocol is presented. In Section IV, we provide the implementation results and comparisons. Finally, Section V, draws conclusions.

II. I2C PROTOCOL

I2C protocol can communicates with many devices in the same bus network, each device is recognized by its unique address. The data is transmitted by two wire, bidirectional serial bus (SDA and SCL) of the I2C bus network. The start signal and the stop signal mean that the transfer would start and the transfer has ended respectively. The conditions of start and stop have been showed in Fig.1. The negative edge and the posedge edge of SDA in the high level of SCL represent the start signal and the stop signal respectively. The signal of SDA would not reverse in the high level of SCL when the other signals are transmitted. An acknowledgment bit (ACK) or a not acknowledgment bit (NACK) must be transmitted after transmitting the data of one byte in the I2C protocol. The messages of I2C protocol can be divided into two kinds: the address frame and the data frame. The address frame would be transmitted after the start signal, the higher seven bits of the address frame represent the address of slave device, the last one bit is the operator (high level and low level represent the operator of read and write) respectively. The data frames follow the address frame closely. The phase of data frame would transmit several bytes continuously until the stop signal is coming. The meaning of every byte would be different in various applications.

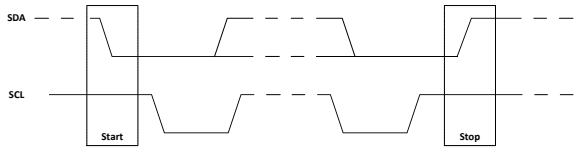

Fig. 1: The start and the stop conditions

III. DESIGNED HARDWARE ARCHITECTURE FOR SLAVE DEVICE OF I2C PROTOCOL

The flexible hardware architecture of I2C is urgently needed to satisfy various applications. We would design a hardware architecture of I2C in this section. The designed hardware architecture consists two level: protocol level and application level. The protocol level, which can be reused in various application to implement the basic function of I2C protocol. The application level bases on the protocol level to implement the function oriented the actual demands.

A. Designed Hardware Architecture of Protocol Level

The designed hardware architecture of protocol level is shown in Fig.2. The signal of SDA and SCL is sampled by CLK signal, which is a clock signal of 50MHz. Because the frequency of CLK signal is much higher than the band rate of I2C bus, the frequency of CLK signal need not to change for different band rate. The sampled signals of SDA and SCL are buffered by registers. Then, the shift-register is used to obtain the posedge edge and negative edge of SDA signal and SCL signal. The edge signals of SDA and SCL are mainly used to complement the following modules:

- Com. Start: This module is used to obtain the start signal. When the negative edge of SDA is detected in the high level of SCL, the start signal would be enabled, that means the transfer has started.
- Com. Stop: This module is used to indicate that this transfer has be ended. The stop signal would be enabled when the posedge edge is detected in the high level of SCL.
- Sel. Device: This module is used to compare the device address sent by master and the local device address to determine this slave device is selected or not, and this module also obtains the type of the following operation (reading operation or writing operation). The results of this module would be sent to the modules of Tran. Data, Recv. Data and the application level for the following operations.
- Tran. Data: When the local slave device is selected and the reading operation is enabled, this module would send the data provided by application level to the master device. The data must be converted to the serial data before sending to the output buffer. The state of this module should be provided to the application level for determining the next operation by application level.
- Recv. Data: When the local slave device is selected and the writing operation is enabled, this module would receive the data sent by master. When every byte is received completely, this module would send a completing signal to the application level, then the application level would obtains the received byte. This module and the Tran. Data module maybe execute circularly until received the stop signal.

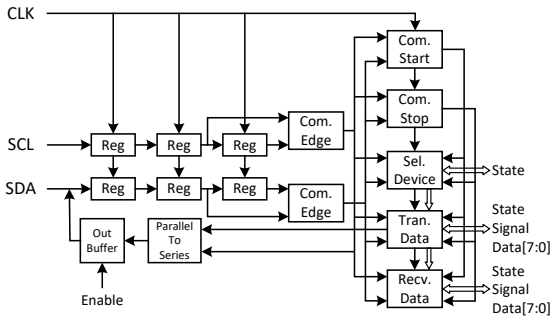

Fig. 2: The hardware architecture of protocol level

B. Designed Hardware Architecture of Application Level

The application level, which bases on the protocol level is implemented by the finite state machines (FSM). This level is used to implement the function oriented different actual demand. The FSM is presented in Fig.3.

In the Idle state, the local device detects the SDA signal and the SCL signal until the device receives the start signal. Then, the Local Device Address state would judge that the device address sent by master is equal to the local device address or not. The operation provided by the designed circuit can be classified into three categories: Current Address Reading (CAR), Random Sequential Reading (RSR) and Random Sequential Writing (RSW).

- CAR: This category is used to read the data of several bytes from the beginning of the first address of register files. This operation would complete until the slave device receives a NACK signal, and the stop signal is received after the NACK signal. It is noteworthy that the NACK signal and the stop signal are both sent by the master device.
- RSR: This category is used to read the data of several bytes from the beginning of subaddress, which is a 16 bits data. The master writes the subaddress firstly. Secondly, the master would send the start signal and the local device address again. Then, the slave device sends the corresponding bytes one by one until receives the the NACK signal and the stop signal sent by master.
- RSW: The master would write the data of one or several bytes from the beginning of subaddress to the slave device in this category. The master also writes the subaddress to slave device firstly. Then, the slave would receive the corresponding bytes sent by master until the stop signal is received.

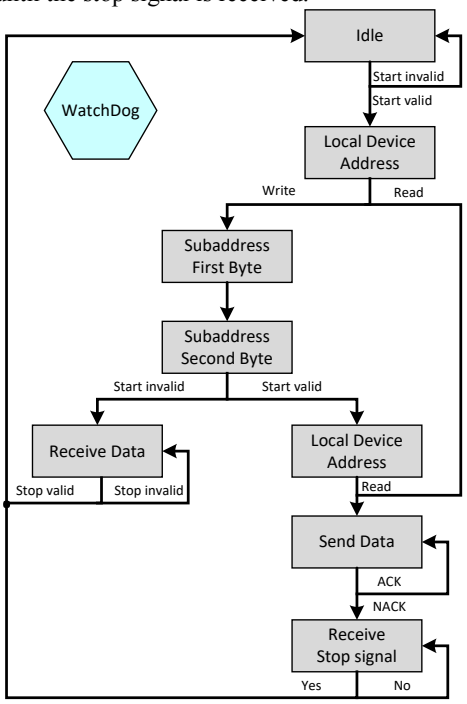

Fig. 3: The state diagram of application level

A watchdog module is contained in the application level. It is used to reset the circuit to the initial state (Idle state) when the circuit enters the error state and over the time threshold. The watchdog module can effectively protect the designed I2C module in the actual application.

IV. Implementation Results and Comparisons

The designed hardware architecture is synthesized and implemented by ISE of Xilinx. The implementation results is showed in Tab.1. Due to needing lots of resource to implement the register files in the application level, the occupied resource of application level is much larger than the occupied resource of protocol level as shown by Tab.1.

Table 1: The implementation results of designed hardware architecture

	Application Level	Protocol Level
FPGA Device	XC6SLX9-2TQG144	XC6SLX9-2TQG144
Slice Registers	2226	41
Slice LUTs	2995	55
Occupied Slices	816	24
MUXCYs Used	40	0

We also simulate the designed hardware architecture by ISim of Xilinx. Fig.4 is the simulation results of start and stop signal. Fig.5 and Fig.6 are writing twelve bytes to slave device and reading twelve bytes from slave device respectively. The simulation results show that the timing sequence of designed hardware architecture can satisfies the demands of I2C protocol. Maximum number of writing and reading can be set to 216 = 65536 bytes in this design. We are limited to the resource of FPGA, the depth of register files is set to 256. The bytes of writing and reading, which is generated by Matlab is random numbers in the simulation.

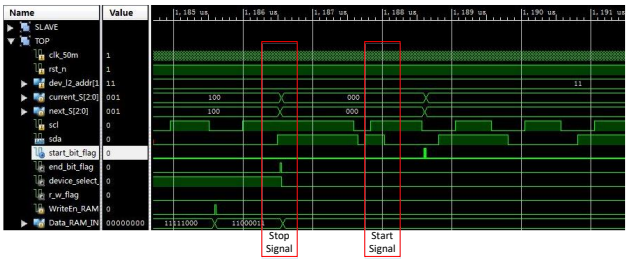

Fig. 4: The simulation of start and stop signal

Fig. 5: The simulation of writing twelve bytes to slave device

Fig. 6: The simulation of reading twelve bytes from slave device

As shown in Fig.7, we built a verification platform to verify the performance of the designed hardware architecture. The FPGA is used to run the designed hardware architecture, the coding switch can set the FPGA with various device address. The designed master module of I2C bases on the chip of CH341T[9], which can convert the I2C protocol into USB protocol. we also develop an upper computer software by python to implement the I2C operation of writing and reading several bytes. The designed software can easily writes data to the corresponding slave device from the excel document, and reads data from the corresponding slave device. The start address, band rate and the length of data can also be set easily by the software. Fig.8 and Fig.9 show the configuration interface and the reading/writing interface of the designed software respectively.

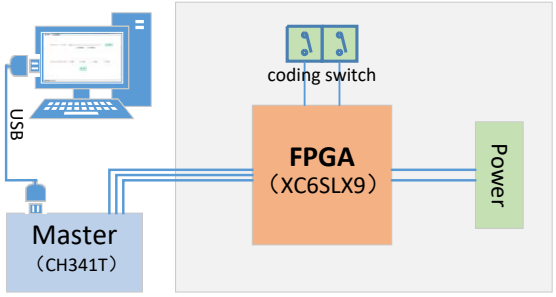

Fig. 7: The structure diagram of verification platform

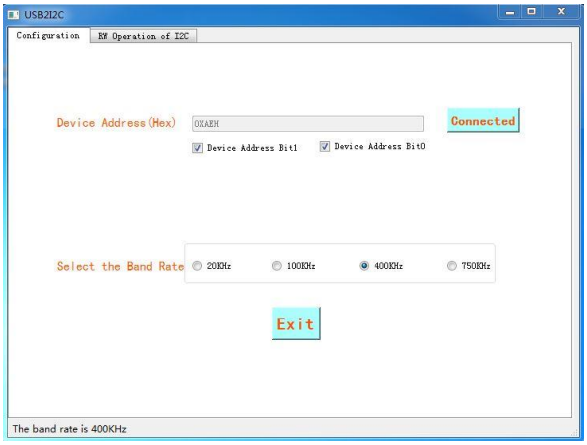

Fig. 8: The basic configuration of slave device

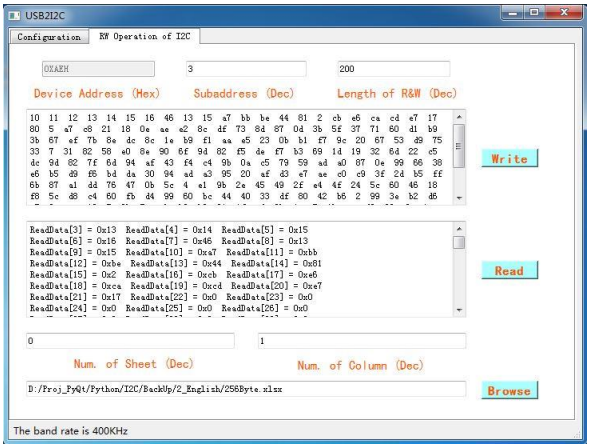

Fig. 9: The configuration of slave device to write data and read data

The designed hardware architecture is divided into two levels: protocol level and application level. The protocol level implements the basic operation defined by I2C protocol, and the application level would aim to the function needed by the different users. Therefore, the designed hardware architecture is more flexible than the existing works. An upper computer software is also designed. Compared to the existing works, the writing and reading operation completed by the designed software is more humanized.

Table.2 illustrates the comparison results of the designed hardware architecture performance with several existing hardware architectures. The hardware architecture designed in [6] and [8] only can be used to read and write one byte in a transfer, that is similar to the function of the protocol level designed in this paper. However, less resource is needed in this paper. Moreover, the designed hardware architecture of protocol level implements the basic operations of I2C protocol, which can be used in various applications.

Table 2: Performance comparison of I2C module

	App. Lev (This Work)	Pro. Lev (This Work)	[6]	[8]
FPGA Device	XC6SLX9-2TQG144	XC6SLX9-2TQG144	Spartan 3AN	-
Slice Registers	2226	41	169	78
Slice LUTs	2995	55	325	98
Occupied Slices	816	24	83	60
MUXCYs Used	40	0	-	-
Note: App. Lev and Pro. Lev represent application level and protocol level respectively				

V. CONCLUSIONS

The slave device of I2C bus is one of the most vital components in lots of chips. The I2C protocol only defines the basic operations: start, stop, writing several bytes and reading several bytes. Various applications would define different combination of the basic operations, that lead to different

application needing different hardware architecture. Hence, the flexible hardware architecture is urgently needed. The designed hardware architecture is divided into two levels: protocol level and application level. The protocol level implements the basic operations defined by I2C protocol. The application level is used to implement the function oriented different applications. The designed hardware architecture is implemented and simulated by ISE and ISim of Xilinx. A platform, which consists of FPGA, master device of I2C and upper computer software is built to verify the designed hardware architecture.

ACKNOWLEDGMENT

This research was supported by the National Science Foundation of China (Grant No. 61804183) and the Hubei Provincial National Science Foundation of China (Grant No. 2018CFB291).

REFERENCES

[1] G. Panic, D. Dietterle, Z. Stamenkovic, and K. Tittelbach-Helmrich, "A system-on-chip implementation of the IEEE 802.11a MAC layer," in Euromicro Symposium on Digital System Design, 2003. Proceedings., Sept 2003, pp. 319–324.

[2] Z.-Y. Cao, Z.-Z. Ji, and M.-Z. Hu, "An image sensor node for wireless sensor networks," in International Conference on Information Technology: Coding and Computing (ITCC'05)-Volume II, vol. 2, April 2005, pp. 740–745 Vol. 2.

[3] STMicroelectronics, "The datasheet of STM32F767xx," https: //www.st.com/resource/en/datasheet/stm32f767zi.pdf.

[4] S. A. Fechtel, P. Schollhorn, M. Speth, G. Fock, and C. Schotten, "Advanced receiver chip for terrestrial digital video broadcasting: architecture and performance," IEEE Transactions on Consumer Electronics, vol. 44, no. 3, pp. 1012–1018, 1998.

[5] P. Venkateswaran, M. Mukherjee, A. Sanyal, S. Das, and R. Nandi, "Design and implementation of FPGA based interface model for scale-free network using I2C bus protocol on Quartus ii 6.0," in Computers and Devices for Communication, 2009. CODEC 2009. 4th International Conference on. IEEE, 2009, pp. 1–4.

[6] B. Eswari, N. Ponmagal, K. Preethi, and S. Sreejeesh, "Implementation of I2C master bus controller on FPGA," in Communications and Signal Processing (ICCSP), 2013 International Conference on. IEEE, 2013, pp. 678–681.

[7] Z.-w. Hu, "I2C protocol design for reusability," in Information Processing (ISIP), 2010 Third International Symposium on. IEEE, 2010, pp. 83–86.

[8] J. K. Singh, M. Tiwari, and V. Sharma, "Design and implementation of I2C master controller on FPGA using VHDL," IJET, vol. 4, no. 4, 2012.

[9] XinHeng, "The datasheet of CH341," http://www.wch.cn/dow nloads/CH341DS2 PDF.html.