Design and Verification for Data Acquisition Interface

ADC_USB IP Core

Bo Ling\textsuperscript{a*}, Zheying Li\textsuperscript{b}, Shaozhen Zhang\textsuperscript{a}

\textsuperscript{a} School of Electronic and Information Engineering, Beijing Jiaotong University, Beijing, 100044, China
\textsuperscript{b} College of Information, Beijing Union University, Beijing, 100101, China

Abstract

A series of portable mass storage devices are arising due to the effective support from USB interface for its special features, such as easy to use, high-speed and low power. ADC_USB IP core is to achieve data acquisition and efficient data transfer to PC. The USB bus protocol and communication principle is introduced firstly. Then the design and verification of ADC_USB IP core are discussed in detail. In addition, data streams and transport are analyzed in detail. Modules of ADC_USB protocol controller are designed with Verilog HDL. The design is synthesized with Quartus tool and verified by FPGA.

1. Introduction

USB (Universal Serial Bus) is a new interface between PC (Personal Computer) and peripheral devices. It has some advantages comparing with some other interfaces, such as high speed, low power, plug to play and easy to install. Now, USB has become the first choice of connections between PC and peripheral [1] [2]. So, nearly all the chips designed have a USB interface module. In this way, the chips can easily communicate with PC at a high speed.

With the development of computer technology, data acquisition plays a more and more important role in the medicine industry. High performance data acquisition equipment is necessary when doctor want...
to get some data from body [3]. All of the data acquisition chips transferred data using RS-232 interface before. The solution cannot satisfy the requirement now because of its low speed [4]. ADC_USB can transfer data between PC and peripheral at a high speed and can be configured by PC through the connection.

2. USB Interface Communication Principle

2.1. The Basic Unit of USB Interfaces

Packet is the basic unit of information in the USB system, structure as shown in Figure 1. The packet types of USB are including token packet, data packet, handshake, and special packages [5].

<table>
<thead>
<tr>
<th>SYNC Field</th>
<th>PID Field</th>
<th>Data Field</th>
<th>CRC Field</th>
<th>EOP Field</th>
</tr>
</thead>
</table>

Fig.1 USB data packet structure

2.2. USB Data Transmission Pattern

USB transfers data between host buffer and end point device through channels. To achieve universal, USB protocol provides four different data transmission types: control transmission, bulk transmission, isochronous transmission and Interrupt transmission [6].

The IN (input) / OUT (output) transaction of synchronous transmission has not any handshake packet, only consisting of the token packet and data packet. So, the error of data does not automatically retry. As shown in Figure 2, the data packet PID (packet identifier) is replaced by DATA0, DATA1, DATA2 and MDATA in turn.

![USB data packet structure](image1)

![USB data transmission pattern](image2)

![USB interface circuit model](image3)
3. The Overall Structure of the ADC_USB IP Core

In Fig. 3, host uses interrupt transmission to configure external ADC’s start enable through EP1 to start ADC. Then EP2 receives ADC conversion data, and converts the data into an 8 bit common data which is sending to PC by USB1.1 IP. The communication between IP core and host is carried out in accordance with USB interface Transfer Protocol.

EP3 is a self-test endpoint, which loops output data from 0 to 63. The practical application of EP3 is to determine the ADC_USB interface working properly or not by detecting the output data.

4. ADC_USB IP design and Implementation of Each Module

ADC_USB IP module is consisting of EP1, EP2, EP3, and USB interface which is mainly including PHY (USB Transceiver Macrocell) and SIE (Serial Interface Engine) component. The circuit structure is as shown in Fig 4.

4.1. Design of PHY Module

The USB bus data line is formed by the DP and DN. Host distinguishes low-speed USB devices from high-speed USB devices through DP and DN, pull-up, pull-down [7].

4.2. Design of SIE Module

Serial Interface Engine (SIE) includes PL (Protocol Layer) and UTMI (UTM Interface). PL includes PD (Package Disassembly), PA (Package Assembly), and PE (Package Engine) [1].

Before USB and device can communicate normally, UTMI interface is mainly responsible for dealing with the state to coordinate the speed of USB interface and the detection of suspend or reset [8].

PL including PA, PE, PD and two CRC (Cyclic Redundancy Check) cell is a key component of SIE. What’s more, it is responsible for communication control of all input and output data. In fact, CRC unit is used for CRC calculation. CRC unit 1 needs to calculate CRC5 (5 bits Cyclic Redundancy Check) and CRC16 (16 bits Cyclic Redundancy Check), but the CRC unit 2 only needs to calculate CRC6.

PD can analyze packets which are received by UTMI. And in a packet, there are identifier (PID), device address, endpoint address and valid data command. At the same time, it also verify PID (PID [7:4] = ~ PID [3:0]). CRC5 is necessary for token packets, and CRC16 is for data packets. In addition, CRC5 can be expressed by x^5 + x^2 +1, and CRC16 is expressed by x^16 + x^15 + x^2 +1.

According to the PID sent by PE, PA organizes the corresponding information packet. The data transmission is arranged in the corresponding data packet, or token packet. Sending token packets does not need the CRC5 as parity bit. While sending the data packet, the last data needs CRC16 to be effective as parity bit.

PE receives the results after PD’s analysis and verification. If there is the validation error, PE will directly call the PA unit to send NAK (Negative Acknowledge) state packet. But if it is checked correct, PE will judge the received token packet. Each time the transmission completed, PE processes transmission affairs according to the current configuration and the current state. At the same time, PE will real-time update control status register CSR.

4.3. Design of EP1 Module

EP1 is designed to be the OUT endpoint of interrupt transmission; and it receives the transmission of...
information from the host. Then EP1 controls the enable signal of ADC through judging the lowest data received. If the lowest bit is 1, the ADC’s enable signal will be availed; if it is 0, the enable signal of ADC will be invalid. In 12-bit ADC, the enable signal is the key signal of control signals.

4.4. Design of EP2 Module

EP2 is the key part of ADC_USB, and it transfers the results of ADC to PC with synchronization transmission. Additionally, EP2 is also responsible for receiving the 12 bits data from ADC, and it will convert the 12 bits data to 8 bits availed data of PC. When the number of 8 bits data is 64, the data are sending to PC host through USB1.1.

In order to make the data continue at a constant speed sampling, EP2 is designed at double buffer (buffer) mode. And each buffer for 64 bytes, that is just the maximum effective length of a synchronous packet. One of the buffers is used to receive the data from ADC, and another is used to send data through USB. They work in turn to ensure the data acquisition and continuous transmission. Any buffer receives 64 bytes, informing the group bag module (PA) to ready sending data packets.

Another function of EP2 is to start the 12-bit ADC when the enable signal of EP1 is effective and sends the first start signal called clear. Then it will wait for a complete conversion signal of ADC EOC, and after receiving the signal EOC, it will send the signal clear again. If the two buffers are in full state, EP2 will not start, waiting for a signal of one buffer empty again. Continuing working as such, it will stop to send the signal clear until the host makes the EP1 invalid.

4.5. Design of EP3 Module

EP3 is designed for synchronization output endpoint, whose function is writing data between 0 and 63 when the buffer of EP3 is empty. After the host takes the data in EP3 buffer, EP3 will automatically again write from 0 to 63, waiting for the host reading data next time. EP3 is designed for the test of transmission function of ADC_USB in FPGA or the application.

5. Simulation of System

5.1. Verification Platform

Verification is an important part of design of IP core, and the platform’s complete or not is affecting the complete of IP core directly [9].

According to the agreement, there is ROM with descriptor of device in this design. So the function of ADC_USB IP core is of USB interface, and it can be used as a USB device to communicate with the host. To make the functional verification and simulation for ADC_USB, it is very important to set up a complete test platform.

In Fig5, there is the test platform in this design. ADC model can work to produce 12 bits data Dq and the signal eoc. And according to clear, it becomes invalid after the eight clock cycle, the output signal eoc is effective; Every time when Eoc is effective, Dq will add 1, and the initial value of Dq is setting to 12’h000.

PC terminal function is simulated by USB host, which can produce host token bag, the packet, etc. USB host set the address of ADC_USB and PC by calling to set some tasks. Then PC calls OUT task to ADC_USB, EP1 sending a 8byte data to start ADC, according to the sated address. Then, by calling IN task, it will receive the data from ADC to complete a whole verification.
5.2. Results of Verification

As shown in figure 6, there is the simulation result of ADC when it is started and to convert data. The signal of ADC_en goes higher from 0 to 1 after EP1 receiving the packet of start data. Then, ADC starts to work normally according to the control of clear. After the eight clock cycle of clear becomes invalid, the output signal Eoc is effective and taken a clock cycle. When Eoc is of efficient, Dq will be added 1, which shows the right function of ADC_USB EP1. That is to say that the host can set up the enable signal of ADC correctly.

Figure 7 shows a part of the data PC get in the Modelsim Transcript window. For the sake of the scale of data, not all data is showed here. Output data from the ADC is 12-bit width data which is counted from 12’h00 and increased by 1 each time. EP2 reorganizes the data in 8-bit width, convert two 12-bit data into three 8-bit width data. After the reorganization, as we see from figure 7, the output data sequence is 12’h000 12’h001 12’h002 12’h003 12’h004 and so on. This is the same as the data outputted from the ADC. It known that the ADC_USB IP conforms to USB communication protocol and can communicate with PC according to the verification.

5.3. FPGA verification

FPGA verification is an essential step in the digital design process. It can be known that the design can work well in a real environment. The design is downloaded into FPGA EP2C35F672C8N which is a
product of Altera. Clock is configured to a 50M clock source on board. Reset pin is assign to SW0. Signaltap is used to monitor the data transferred through EP3.

Fig 8 is the data from EP3. The output data of the EP3, which is counted from 8’h00 and increased by 1 each time, is sending to host. In fact, C3h means the head of the data package (DATA0). And the following data is output data from EP3. Now, it can make sure that the design can work well in the real environment.

6. Conclusion

The ADC_USB IP Core design here solves the communication problem between PC and data acquisition equipment. The IP is also easy to be applied to other SOC system.

Acknowledgements

The project is supported by the National Natural Science Foundation of China (Grant No.609976024) and Beijing Municipal Education Commission PHR (IHLB) 2009513.

References

[9] GeShen. The Research of SOC Design Methodology Based on Reusable IP Cores. Xi’an: Northwestern Polytechnical University, 2002