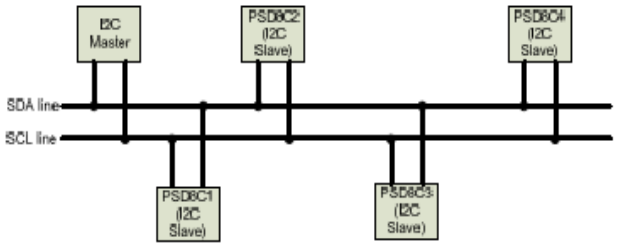
Analysis, Verification and FPGA Implementation of Low Power Inter Integrated Circuit Interface for Custom Asics

Jaswant Singh Rathore[1]
[1]M. Tech. Research Scholar
[1]Mewar University

Abstract---This paper presents the design and simulation of an I2C (Inter-Integrated Circuit) serial interface. The design was described using the Verilog® hardware description language. The I2C Interface is intended for use in a family of integrated circuits (ICs) used in the detection of ionizing radiation. These custom ICs aid scientists in carrying out a wide variety of nuclear physics experiments. The ICs are being developed by the Integrated Circuit Design and were fabricated in a 5-Volt AMIS 0.5 µm, double-poly, tri-metal CMOS process (C5N). The current ICs only provide for analog outputs. Storing data in a digital format on chip before transmittal to a host computer via an I2C or "I2C-like" interface should result in improved system performance since the transmission of digital data is much less susceptible to interference form environmental noise. A large number of ICs are required in these types of instrumentation systems and therefore, at some point in the future, the I2C interface described herein will likely be incorporated along with an on-chip ADC and buffer RAM into second-generation versions of our current chips. The design of the proposed I2C interface was prototyped using inexpensive, readily-available Xilinx Spartan3E Starter development boards manufactured by Diligent. The prototype system consisted of two I2C "slaves" (emulating a pair of iii custom ICs) and an I2C "master". The "master" gathered results from the "slaves" and transmitted the data to a personal computer where it was displayed. The prototype system worked flawlessly at 100 KBits/sec but can accommodate up to 128 slaves.

I. Introduction

The Interconnect Integrated Circuit or I2C interface [Phi:00] was originally developed by Philips Semiconductors Company for data transfer among ICs at the Printed Circuit Board (PCB) level in early 1980s. All I2C-bus compatible devices have an interface allowing them to communicate directly with each other via the I2C-bus. The concept provides an excellent solution for problems in many interfacing in digital design. I2C is now broadly adopted by many leading chip design companies like Intel, Texas Instrument, Analog Devices, *etc*.In I2C, only two bi-directional lines are required to carry information between devices, a Serial Data (SDA) line and a Serial Clock (SCL) line. Each device can be recognized by a unique 7 or 10 bits address. The device initiates a communication is called the Master, and at that time, all the other devices on the bus are considered Slaves. Normally, Masters are Microcontrollers. The I2C bus is a multi-Master bus, but only one Master can initiate a data transfer at any time.



. Fig.1: PC BUS Configuration

In the figure above, there is one Master; the other devices are all Slaves. When a Master wants to initiate a communication, it issues a "START" condition. At that time, all devices, including the other Masters, have to listen to the bus for incoming data. After the "START" is issued, the Master sends the "ADDRESS" of the Slave that it wishes to communicate with along with a bit to indicate the direction of the data transfer (either read or write). All Slaves will then compare their addresses with the address received on the bus. If the addresses are identical, the Slave with the matching address will send an "ACKNOWLEDGEMENT" (ACK) to the Master. Slaves whose addresses do not match will not send an ACK. Once communication is established, the two lines are busy. No other device is allowed to control the lines except the Master and the Slave which was selected. When the Master wants to terminate communication, it will issue a "STOP" signal. After that, both SCL line and SDA line are released and free.

II. I2C Implementation on FPGA

In this paper I am going to implement an I2C Master and an I2C Slave on Xilinx Spartan3E (500E) FPGA development boards manufactured by Digelent. Both the Master and the Slave will be described using Verilog. Before going into detail about how to implement the I2C Master and I2C Slave, it is important that one understands the architecture of the FPGA and the development board. The MicroBlaze is a soft microprocessor which can be placed into a Xilinx Field Programmable Gate Array (FPGA). The MicroBlaze processor is a 32-bit Harvard Architecture Reduced Instruction Set Computer (RISC) architecture. It has separate 32-bit instruction and data buses, and it can access data from both onchip

And external memory at the same time. The MicroBlaze can be connected to many types of peripherals; for examples, UARTs, GPIOs, Flash peripherals, *etc*. This is done via the Processor Local Bus (PLB) or the On-chip Peripheral Bus (OPB) [Jes: 06]. In the figure below, we can see an example of MicroBlaze System.

We can also add a User Peripheral (in our case the I2C Master) to perform a specific application and then use the MicroBlaze to control it. The User Peripheral itself can be connected to an external hardware either inside or outside

the Spartan 3E board, *i.e.* ADC, DAC, DIP Switches, LCD, LED or I/O pins, *etc*.

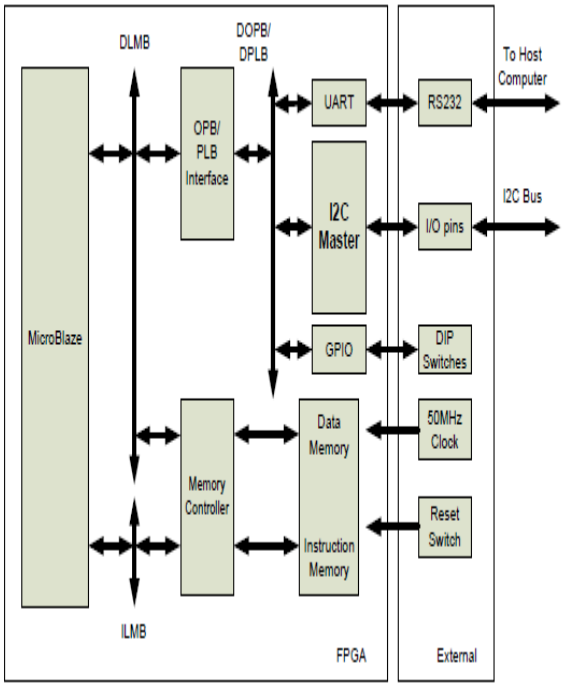


Fig.2: Basic Architecture Microblaze System

Slave Implementation on FPGA

As mentioned above, there are two types of devices on the I2C bus, Master and Slave. A Slave cannot initiate a data transfer. It just monitors the SDA and SCL lines waiting for a START signal. After detecting a START signal, the Slave will compare its address to the address received. If the addresses match, it will perform an action requested by Master by either sending or receiving data. There are six main functional blocks in the design of a Slave:

1) Shift Register
2) Start Stop Detector
3) Sequence Counter
4) Address Comparator
5) ACK Block

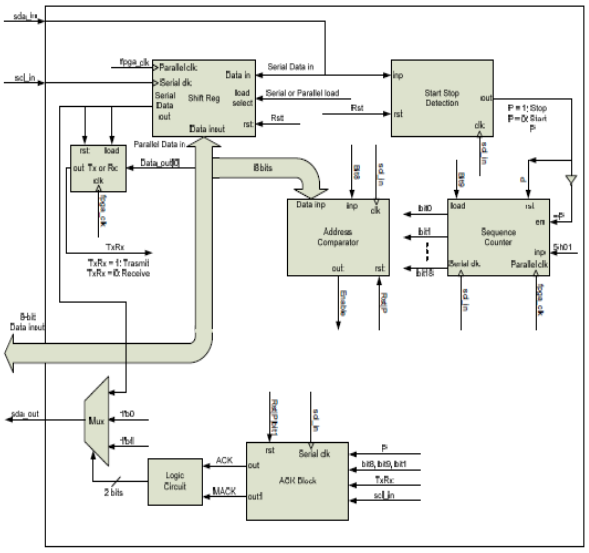


Fig.3: Slave Block Diagram

Recall that the Master Controller is a modified version of an automatically created Verilog file named "user_logic.v". The Master Controller has 16 software accessible S/W registers from which, a user can get hardware status, data, *etc*. and to which, the user can send his command to control the Master. The interface between the Master Controller and the Master is illustrated in Fig 4

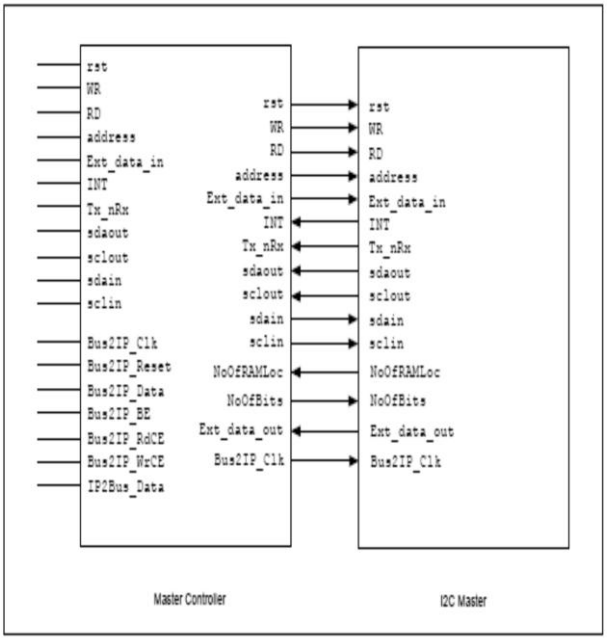


Fig.4: Interface between the Master Controller and the I2C Master.

III. PROTOTYPE SYSTEM

I have described the implementation of a Master microcontroller, an I2C Master, and an I2C Slave. Here in Chapter 4, a prototype system will be introduced to integrate the I2C Interface onto an emulated PSD8C
Chip (using an FPGA) and prove that the I2C Interface is likely to work on the PSD8C chip.

As we know, in the PSD8C chip, the analog output signal will be converted to digital signal using an on-chip ADC and stored in an on-chip RAM. To emulate the chip, we will take advantage of the ADC on the Spartan 3E board to sample an input signal, store the result in a RAM and transfer data to a host computer using an I2C bus.

At first, the ADC's will be reset to sample two analog input signals and the results will be stored in two separate RAMs which are connected to two Slaves, Slave1 and Slave2. The Master, downloaded to another board, is then reset to start communicating with either Slave1 or Slave2 and transfer data it receives over the RS232 DCE port to a host computer. On the host computer, a C# Program is created to receive data from its RS232 DTE port, convert digital signal back to an analog format and display the

Signal as a waveform on a display screen. The C# Program has an option for a user to select which channel (Slave 1 or Slave 2) to display.

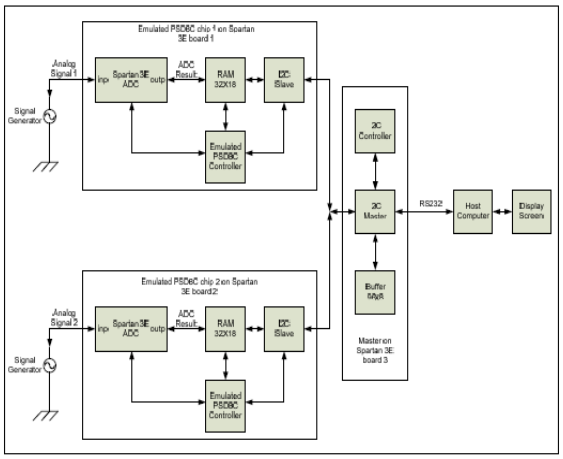


Fig.5: Prototype System.

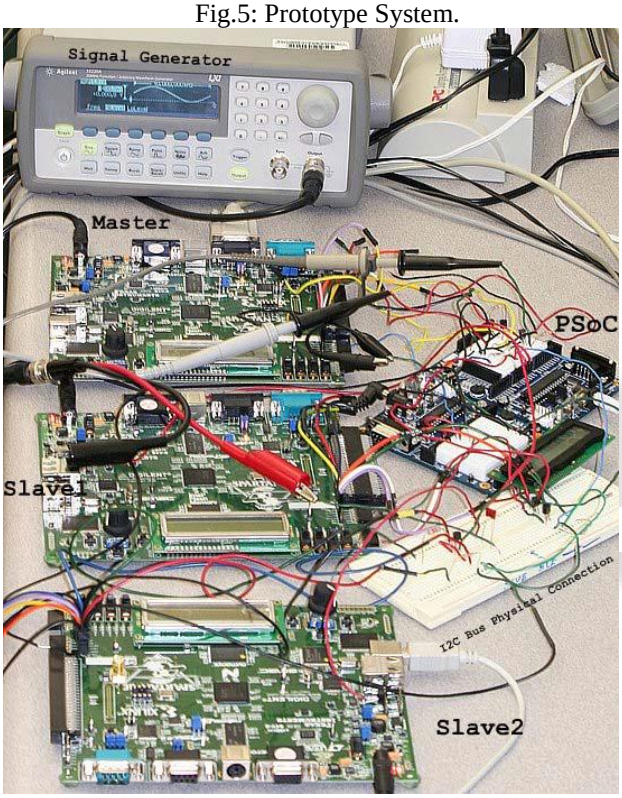


Fig.6: Testing prototype system

Summary: An I2C interface (described using Verilog®) has been developed and fully tested on inexpensive, readily available Xilinx FPGA development boards. The I2C interface was tested at an operating of frequency of 100 KBits/sec. Further testing and additional work is necessary in order to achieve the desired operating frequency of 10 MBits/sec. The present system operates in 7-bit address mode and has the ability to transfer up to 2_{29} bytes per session. The I2C Interface not only simulates
Successfully but operates correctly when implemented on a Xilinx FPGA Development board.

IV. FUTURE WORK

The current PSD8C IC provide for analog outputs. Storing data in a digital format on-chip before transmittal to a host computer over the I2C Interface will result in an improved system performance since the transmission of digital data is much less susceptible to interference form environmental noise sources. As described in Chapter 1 a full system requires the use of many (a few dozen) PSD8C chips.
The system is arranged in the following way. At present, two PSD8C chips are mounted on a chip board along with cable driver circuits that allow the analog outputs of the PSD8C chip to drive long cables which connect to a VME-based ADC. Up to 16 chip boards can then be plugged into a motherboard. In the near future, we plan to modify the PSD8C chip-board. The current plan is to remove the cable drivers from the chip board and replace them with an ADC (similar to or perhaps even the same ADC which is on the Xilinx development board used in this project) and an FPGA which would be used to implement the I2C Slave interface described in this thesis. It will also be necessary to modify the FPGA code to control the ADC and to interact with the PSD8C readout electronics. Finally, in the distant future the I2C Slave may be combined with a twostage flash ADC (currently under development) and a 32 location by 8 bit static RAM and integrated onto a second generation PSD8C chip. Since the I2C Slave was described using a hardware description language, porting the design to an IC is relatively straightforward. One only needs to re-target the design for a standard cell library rather than for the Xilinx FPGA.

REFERENCES

[1] [Xil:07] Xilinx, "Spartan-3E Libraries Guide for HDL Designs", (1995-2007).
[2] [Xil:08] Xilinx, "Spartan-3E FPGA Starter Kit Board User Guide", (June 20, 2008).
[3] [Jes:06] Rod Jesman, Fernando, Martinez Vallina and Jafar Saniie, "MicroBlaze Tutorial: Creating a Simple Embedded System and Adding Custom Peripherals Using Xilinx EDK Software Tools", 2006.
[4] [Phi:00] Phillips Semiconductors, "THE I2C-BUS SPECIFICATION", Version 2.1, (January 2000).
[5] [The:06] Theresa Chou, "MicroBlaze Overview", (2006).
[6] [Das:08] D. Dasari, "Design of On-chip ADC for Custom ASICs Used in the Detec-tion of Ionizing Radiation," SIUE EE M.S. Thesis, (2008).
[7] [Eng:07a] G. Engel, M. Sadasivam, M. Nethi, J. M. Elson, L. G. Sobotka and R. J. Charity, "Multi-Channel Integrated Circuit for Use in Low and Intermediate Energy Nuclear Physics - HINP16C" in Nucl. Instru. Meth. A573, 418-426, (2007).
[8] [Eng:07b] G. Engel, NSF-MRI proposal. This can be found at the PI's WEB site.
[9] [Pro:07] J. Proctor , "Design of a Multi-Channel Integrated Circuit for Use in Nuclear Physics Experiments Where Particle Identification is Required," SIUE EE M.S. Thesis, (2007).

