International Journal of Applied Engineering Research
ISSN 0973-4562 Volume 10, Number 10 (2015) pp. 24871-24882

http://www.ripublication.com

Development of Debug monitor and Uploading Data to Server Using System-on-Chip

Pranathi Nandhigam[1] and Dr.Jayanta Biswas[2]

M.Tech VLSI Design, Sastra University
Thanjavur, 613401, India
pranathi.nandhigam@gmail.com
Associate Professor, School of Computing
Sastra University, Thanjavur, 613401, India
jayantabiswas@ict.sastra.edu

Abstract

The increasing demand for developing complex sys-tems for entertainment and lifestyle products needs the develop-ment of debugging and monitor tools for the developers. These debugging and monitor tools reduces the time to market for new product development cycle. The aim of this paper is to design de-bug monitor for FPGA modules and System-on-chip environment. This process sends the data over socket interface to a remote server in packetized form. Remote server saves the data in files and dis-plays data in real time. *Nios-f* processor is configured on *Altera DE2 education board* using system on chip mode and uClinux op-erating system is loaded using *nios-f* processor. Networking and socket interface are enabled on *nios-f* processor. Synchronization mechanisms between multiple threads in multithreaded architec-ture is implemented for realtime data acquisition system. Avalon bus provides an interface between *nios-f* processor and custom logic. Values of registers and wires are converted to parallel data of fixed width in custom logic synthesized on FPGA. Sampled data is transferred to *nios-f* processor over Avalon bus. *Nios-f* proces-sor sends data in packet format to PC (server) over socket with synchronization. The correctness of the implementation is verified and achieved an average speed of 5 Mbps using *nios-f* processor.

I. INTRODUCTION

Internet enabled lifestyle products and entertainment prod-ucts are based on complex systems. There is a need to reduce the time to market for these products. This leads to develop-ment of debug and monitoring tools for complex systems. The increased complexity in todays state-of-the-art systems make them hard to analyse, test, and debug. Systems-on-Chip (SoC) comprises of reconfigurable software and hardware logic al-lowing for hardware/software co-design and this technology is meeting the demanding requirements of high performance portable embedded system. Custom hardware and software are integrated in a seamless fashion in SoC environment and hard-ware and software work in tandem. Microprocessors (uPs) are deeply embedded within an ASIC which presents a significant debug challenge.

To succeed in developing complex systems, efficient design methods and tools are essential. Observability is the impor-tant aspect in development process. System functionality and timing behaviour should be properly adjusted to attain observ-ability. Mixed signal oscilloscope (MSO), Logic analyzer, de-bug monitor are techniques used to debug embedded systems. MSO cannot analyze more than 16-digital channels at once and logic analyzer is incapable of controlling the target processor in multithreaded environment. To overcome disadvantage of MSO and logic analyzer *debug monitor* is used to increase the observability and performance of the system. An advantage of the *monitor* as a debug methodology is integration with the pe-ripherals and target processor, where the additional cost is ad-justed with low overhead and monitor framework can be used by each system developer independently.

FPGA provide flexibility to prototype a high-density ASIC. FPGA can be used to complete the validation and development.

High-density FPGAs and Hardcopy devices provide customers a competitive advantages by reducing time for development and resources and accelerating time-to-market. Implementing ad-vanced algorithms becomes easy because of the multiprocess-ing function of FPGA. Nowadays, FPGA is considered to en-hance the operation speed of advanced algorithms. With the evolution of VLSI technology, FPGA is more accepted and commonly used in different applications due to *low power con-sumption, short design cycle, fast time-to-market, programma-bility and high density.*

Basic SoC application with socket interface consists of pro-cessor, avalon bus and application oriented custom component shown in Fig 1. Flow of data between the custom logic and processor is done through avalon bus. Bus interface writes the data in to the custom logic through input port and the proces-sor reads the data from the custom logic through output port. Processor interacts with the developed socket interface. Ether-net is configured on ALTERA DE2 Education FPGA board to develop socket interface.

This paper introduce a design scheme for debug monitor and uploading data to server using SoC framework which uses uClinux to boot up the *nios-f* processor. Ethernet/wireless is used as a interface while uploading data to server.

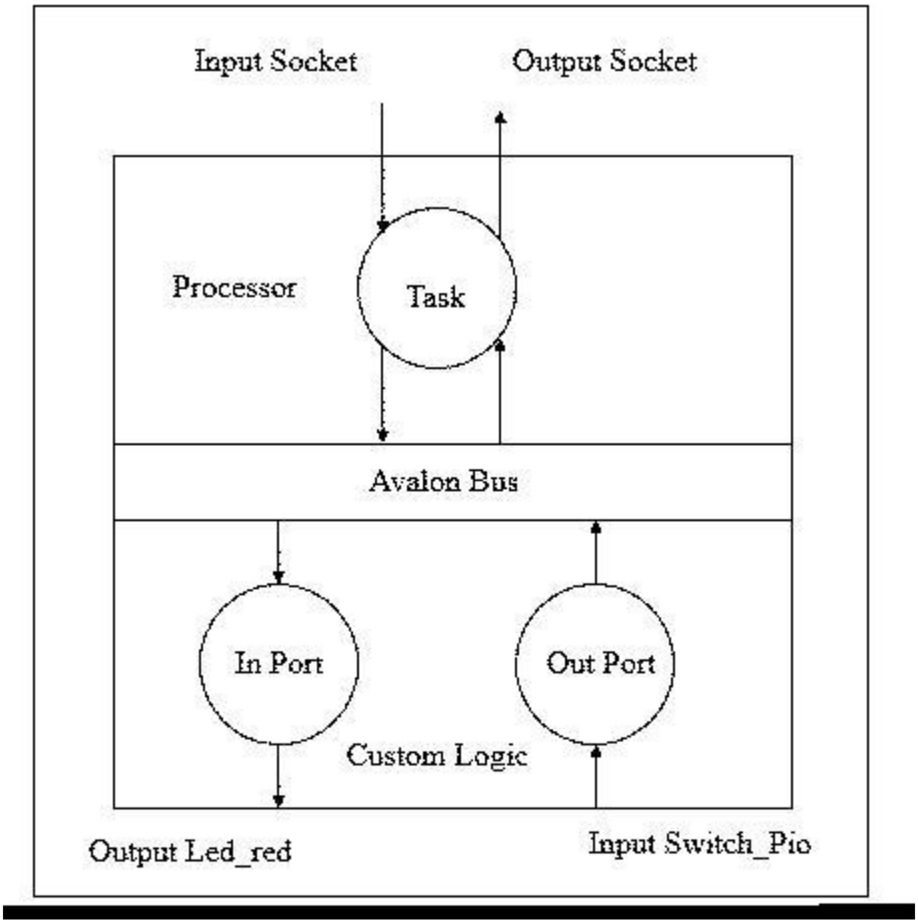

Fig. 1. Block Diagram of SoC Architecture

A. Paper Organization

In section II, a brief description of implemented system us-ing SoPC builder, uClinux, Nios II processor and custom logic is provided. Section III describes the problems encountered during implementation. Section IV describes the details of the design setup. Performance and evaluation of the design is pro-vided in section V and concluded in section VI.

II. SYSTEM DESCRIPTION

High speed data is captured and data is sent over socket in-terface to a remote server in packetized form. Remote server saves the data in files and displays the data in real time. uCli nux is used to boot with *nios-f* processor on Altera DE2 Educa-tion FPGA board using system on chip mode. Networking and socket interface is enabled on *nios-f* processor. Single threaded and multiple threaded software architecture are implemented on *nios-f* processor. However, single threaded architecture is used as multiple threaded architecture is not working because of memory limitation of

ALTERA DE2 Education board. Syn-chronization mechanisms are put in place in *nios-f* processor. Values of registers and wires are converted to parallel data in custom logic. Sampled data is transferred to *nios-f* processor over avalon bus. *Nios-f* processor sends the data in packet for-mat to PC (server) over socket with synchronization. Design of the implemented system is shown in Fig 2.

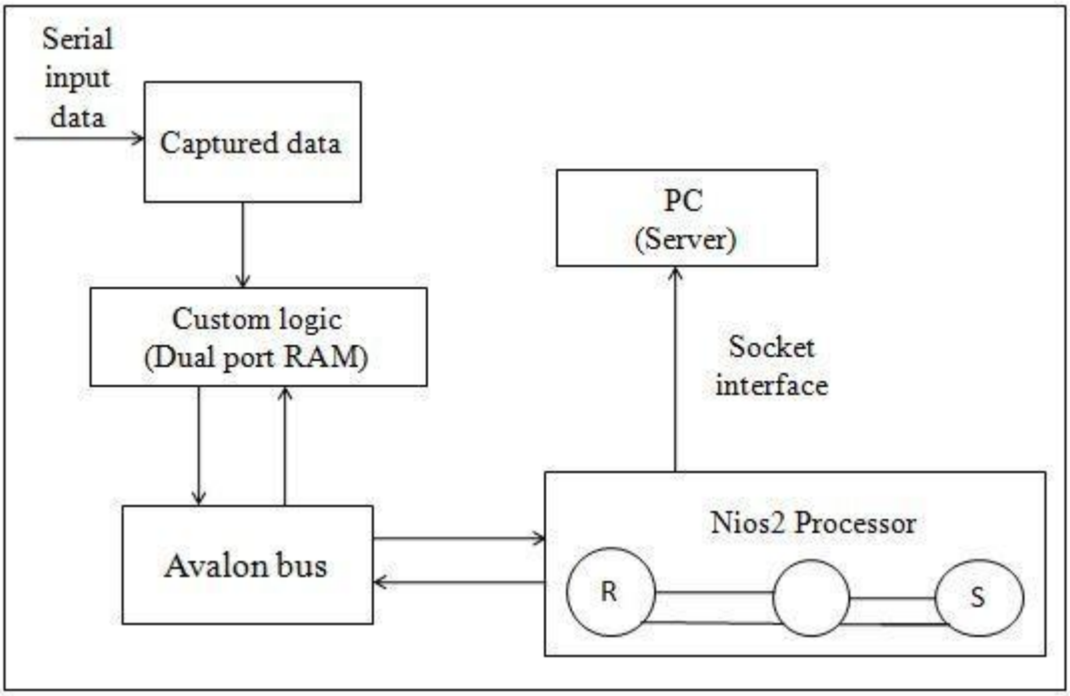

Fig. 2. Design of Implemented system

A. Hardware Design of the SoPC System

System on a Programmable Chip Builder (SoPC) is a tool that connects software and hardware components to form a complete system that runs on various FPGAchips. Library of predefined components (Nios II soft processor,memory con-trollers, interfaces and peripherals) and an interface for inte-grating custom components are incorporated by SoPC builder. Avalon bus is used for interconnections. Bus width matching, clock domain crossing square measure and bus width match-ing are handled by SoPC builder during system generation. A memory accesses and this process includes memory protection, tanslation of virtual addresses, cache control, and memory al-location for software process. In case with no MMU kernel developer manages all the memory accesses. *Nios II* (nommu) core is reduced to less than half when compared to processor with MMU.

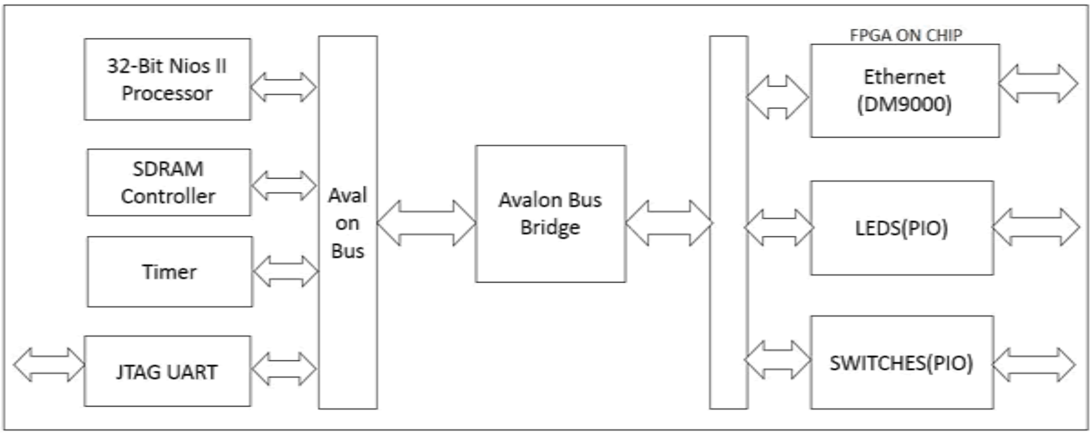

Fig. 3. SoPC builder design of implemented system

graphical user interfaceis used to configure the soft-hardw are components and to specify the bus topology. The *virtual* system is then connected to the outside world via FPGA programmable pins or internally connected to other soft components. The hardware design of the SoPC system is generated after inte-grating the custom component.

FPGA may be configured using JTAG interface. Various ap-plications can be developed using FPGA board and JTAG in-terface. FPGA contain JTAG interface which acts as a key to processor on board. Fs2 software uses JTAG interface to pass the data to oscilloscope [11]. This procedure needs ALTERA JTAG megafunction access license which is very costly. In pro-posed method Ethernet/wireless is used to configure FPGA to transfer the data to PC (server) which offers low cost and flex i-bility rather than JTAG interface.

B. NiosII Processor

Nios II is a 32-bit processor architecture specifically de-signed for entry point cyclone FPGAs in embedded applica-tions. *Nios II* incorporates several improvements over the actual

Nios architecture, making it very appropriate for various em-bedded computing applications, from DSP to system-control. In this paper *Nios II/f* core is used for the design because *Nios II/f* processor offers high performance at the expense of core size. *Nios II/f* processor used in this paper is without mem-ory management unit (nios nommu). Processor without mmu provides advantage of lower cost and low power.

C. uClinux

In this paper, uClinux is used for *Nios II* processors without memory management unit. Memory management unit handles

D. Custom Logic

Debug monitor is a small amount of code which is stored in ROM and it provides observation and control of processors and on-board peripherals. The monitor software does the basic de-bugging and testing functions such as the setting of break points and modification of the processor registers when the target p ro-cessor is halted. Debug monitor is used to analyze the different operating states of signals of interest (internal signals like wires or registers) which are directly or indirectly used to analyze the system performance.

In this implementation the internal states of 127 wires and registers are analyzed. The analyzed data is fed as input to the DPRAM. Custom logic defines the 18-bit Dual-port RAM (DPRAM) implemented in Verilog HDL. DPRAM is used be-cause DPRAM allows multiple reads or writes to occur at the same time. DPRAM has different interface for writing the data to the RAM and reading the data from the RAM. Size of DPRAM depends upon the number of internal registers and wires to be analyzed. Reading rate from DPRAM is faster when compared to writing rate into the DPRAM to avoid the over-flow. Synchronization logic between read and write is defined within the *custom logic*. The width of the input and output ports is 18-bit. The signals involved in the custom logic are *Clock in-put (clk), Chip select (CS), Reset, Write, Writedata, Outport.*

Reading and writing to the DPRAM require different clocks and reset signals. Custom logic consists of internally generated clock and reset signals. Internal clock for writing data into the DPRAM is generated using the counter. An 8-bit counter is used where the MSB of the counter acts as a clock for write cy-cle. Bus clock is used for read cycle. ALTERA DE2 Education FPGA board allows only 50MHz bus interface clock. A slower clock is internally generated in custom logic using bus clock.

The phase of the internally generated clock is shifted from the original clock but the phase shift does not affect the data. Reset condition corresponding to slower clock is specified by proc es-sor. Reset condition (data in == 0xFC00) to slower clock is generated by CPU bus interface commands. On reset custom logic initializes all registers and input signals to initial values. Input register (data in) functions as reset, *count* functions as clock during write cycle and bus clock, internally generated re-set signals are used during read cycle as shown in Fig 4.

Remote cpu sends instructions to be performed on the data. Fig 4 shows the four different instructions *reset, incremental in-sert, incremental delete, complete delete* used in custom logic. The MSB (most significant bit) of 18-bit data functions as en-able bit shown in Fig 6. CPU gives the reset instruction and custom logic attached to bus checks for enable bit in the chan nels. Data is written into input register for channels where en-able bit is 1 during write cycle. This process is described in the Fig 4. *Start of frame* (Sof) is defined with a value (0xfefe). *Sof* is written before writing data to ensure the synchronization be-tween data sampling process in custom logic and data receiving process in the processor. Sampled data is read from DPRAM by processor during read cycle. Data is read in the same sequence as it is written into DPRAM shown in Fig 5.

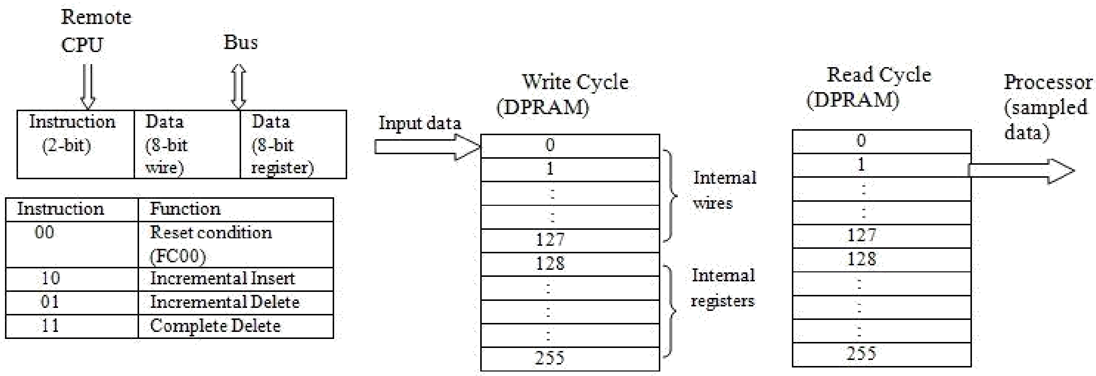

Fig. 4. Functionality of Design

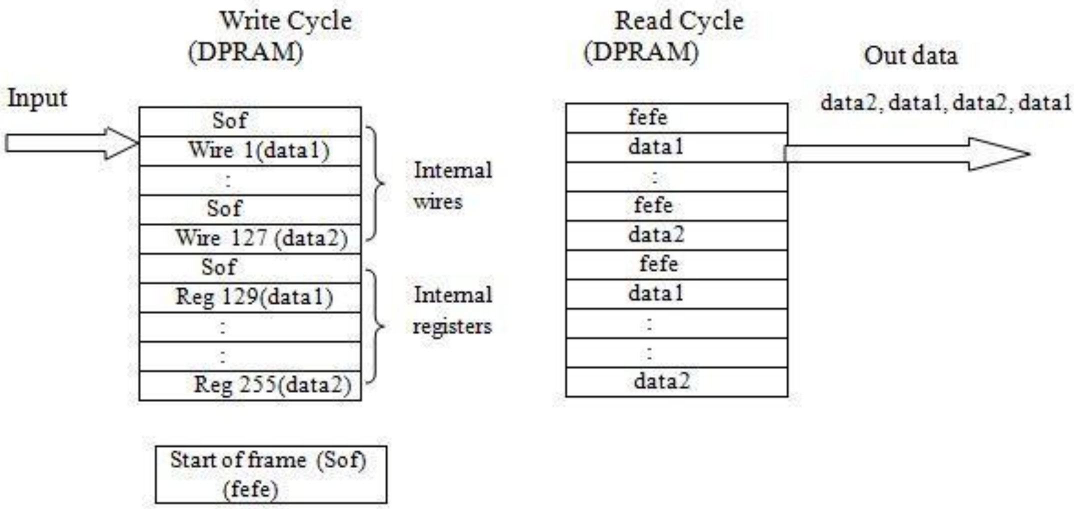

Fig. 5. Functionality of Design

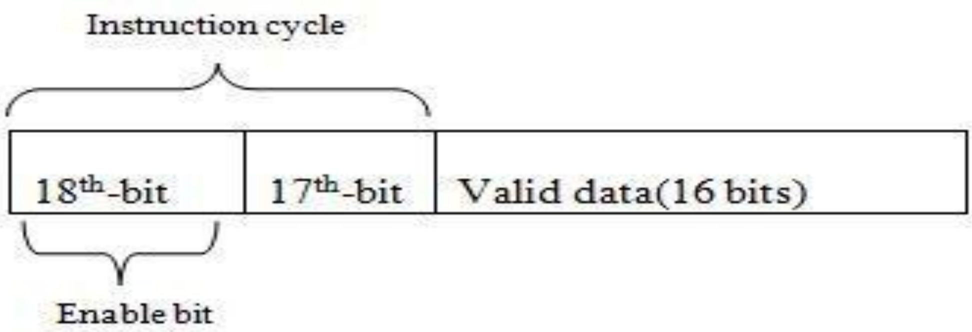

Fig. 6. Representation of input data

Acknowledgement (ack) signal, signature register, write in-dex and *read index* pointers are used to ensure that processor reads a valid data from DPRAM. *Ack* signal is used as hand-shake signal between processor and bus. Signature register is defined with same value as in the *nios* processor. Signa-ture is incremented every time when valid data is written in to DPRAM is shown in Fig 8. At every read cycle proces-sor checks for the *signature, acknowledgement, read index* and *write index* pointer values as shown in Fig 7.

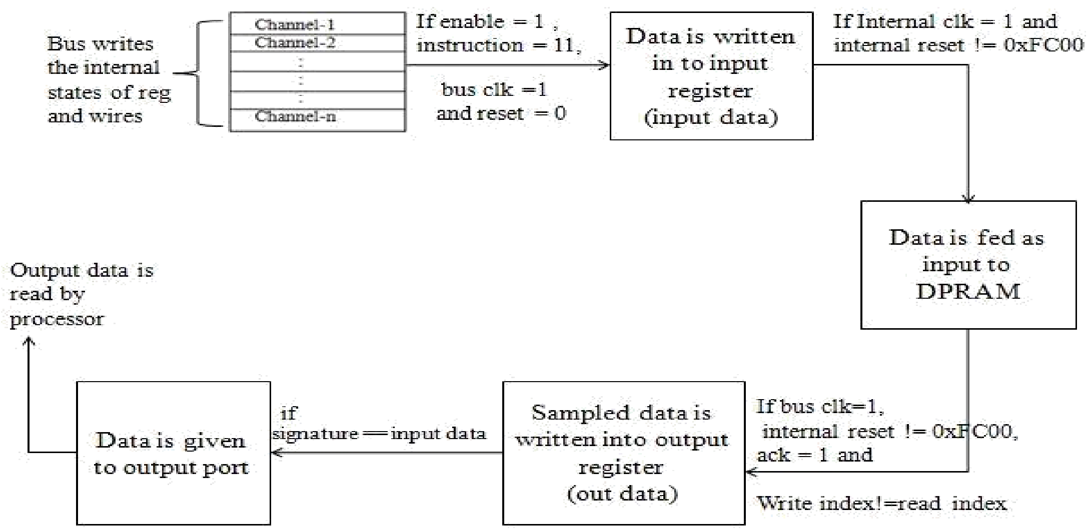

Fig. 7. Block diagram of custom logic

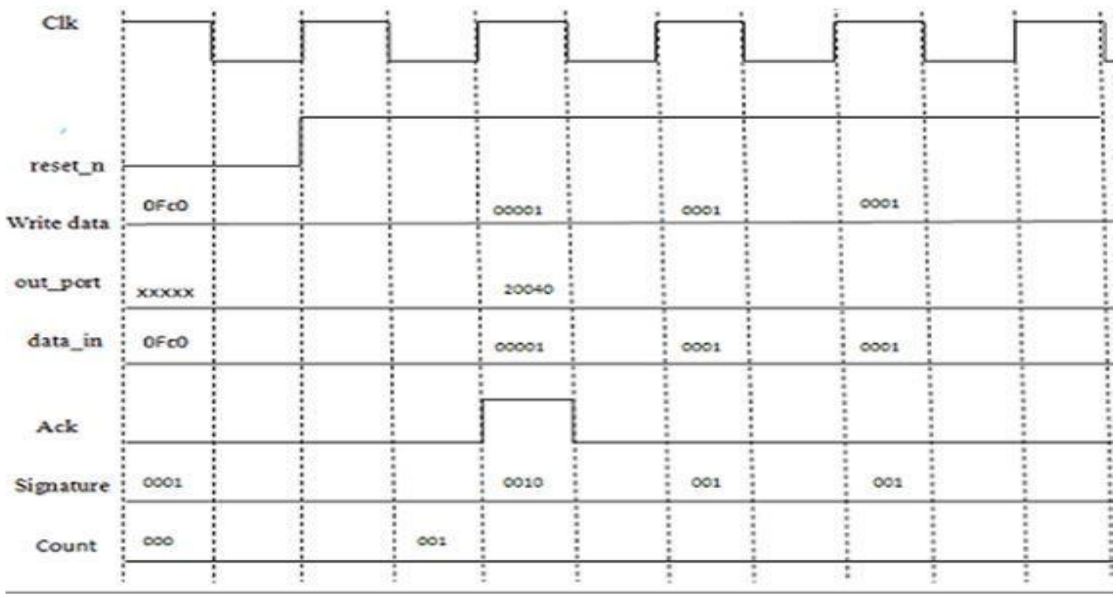

Fig. 8. Timing diagram of Custom logic

III. EXPERIMENTAL SETUP AND METHODOLOGY

SOPC builder is used for designing the development of de-bug monitor. ALTERA DE2 Education FPGA board is used for implementation. Approached methods are

1) Custom component is created with two interface for read and write in SoPC builder and it is attached to bus. Cus-tom component is defined in SoPC builder and system is generated. Functionality of this procedure fails when tested on board. ALTERA DE2 FPGA board does not allow to use the external clock.
2) Selecting the appropriate existing component where the interface of the component is similar to the custom logic from the SoPC builder. Replacing existing system gener-ated Verilog HDL code with the custom logic using the same input and output ports. This technique arises *size issue*, i.e. memory allocation failure when processor boot up.
3) To overcome above problem we take a different approach. Components that are not required for the application are defined at the bottom and required components are de-fined at the top in SoPC builder. This method works but because of difference in verilog code size, the later com-ponents defined in SoPC builder are affected by the ear-lier components.

In this paper, new technique which is not introduced before is used for adding a custom logic to the system i.e, replacing existing system generated Verilog HDL code with the custom logic code having same input and output ports. The input ports and output ports used in the custom logic are connected exter-nally using wires by assigning General-purpose input/output (GPIO) pins to switches and leds used in ALTERA DE2 Ed-ucation FPGA board. Red leds are used as output ports and switches are used as input ports. The data is written in to the leds and read through the switches in the ALTERA DE2 Educa-tion FPGA board shown in Fig 1.

IV. DESIGN IMPLEMENTATION

uClinux is stacked onto the *Nios II* processor, which permits sockets operation for communication over network. Sockets are utilized to accomplish the high speed data transfer and offers flexibility and low cost shown in design setup Fig 9.

1) Bus writes the data corresponding to nodes or internal signals into input register of DPRAM. The output of the custom logic is read by *NIOS II* processor through Avalon bus. The bus width is 18 bits, therefore both read and write signals are also 18 bits wide.
2) Socket interface acts as a medium for transferring data between *NIOS II* processor and the PC. Socket applica-tion is developed using C programming language.

Fig. 9. Design implementation setup

A. I/O Synchronization

The I/O to the user logic is 18 bits wide. Processor reads the data and checks the data for validity. If the MSB is 1, data read from custom logic is valid. *Signature* is 16 bit data which is maintained at *nios* processor and custom logic. This is used for synchronization between processor and custom logic through avalon bus. After reading the data CPU writes the *signature* to custom logic and increments *signature* value. Custom logic matches the *signature* value received from bus and sets *ack* to 1. *Signature* value is also incremented in custom logic. Custom logic reads data from DPRAM read port if *ack* is 1 and *read index* is incremented accordingly and *ack* is set to 0 as shown in Fig 8. *Write index*, *read index* and *signature* registers are used to ensure that processor reads the valid data and aviods data overflow in DPRAM. The input data is internal operating states of wires and registers and the output data is the sampled data. Accordingly, the rate of output data is much more than its input rate. This is analysed during run-time with the help of C programming on *Nios II* processor.

B. Communication

High speed data transfer is achieved between the cus-tom logic and the *NIOS II* processor through avalon bus. uClinux uses a rich set of driver during system on chip boot mode. Socket interface is developed on configuring the ether - net/wireless driver on ALTERA DE2 Education FPGA board which is of low cost and flexible.

C. Memory Mapped I/O

Memory-mapped I/O addressing scheme is used in SoPC builder to connect the peripherals to outside world. The base address of peripherals are available in user interface of SoPC builder. It is easy to access peripherals by utilizing a C applica-tion running on uClinux terminal.

D. Software Execution

The input/output to the external world is implemented effec-tively utilizing uClinux and it is stacked as a part of the *Nios II* processor. C application runs on the board, custom logic gen-erates the data and the generated data is read through socket interface developed between FPGA board and PC. Data is dis-played on PC in real time.

V. PERFORMANCE OF THE DESIGNED SYSTEM

High speed data transfer is achieved between the custom logic and the *NIOS II* processor through Avalon bus. uClinux uses a rich set of driver during system on chip boot mode. Syn-chronisation mechanisms between multiple threads in multi-threaded architecture is implemented for real time data acqui-sition system and achieved an average speed of 5 Mbps. The LUT count of registers and wires is 397 and for memory is 128 in ALTERA DE2 Education FPGA board excluding 127 regis-ters and wires, which adds an extra overhead to the processor.

Speed can be further increased by using ALTERA STRATIX board which has high clock frequency when compared to AL-TERA DE2 Education board.

VI. CONCLUSION

In this paper, a high speed and high-precision debug monitor is designed with the ALTERA DE2 Education FPGA board us-ing System-on-Chip environment and socket interface. Debug monitor is developed using Verilog HDL and implemented on FPGA, thus allowing highly sampled, accurate, flexible debu g monitor system. This proposal is based on FPGA/SoPC based system and the performance may be tuned to actual needs at any time.

REFERENCES

[1] Wang Fei, Wu Zhijie, Chen Hong, Xi Yi2, "High-Speed Data A cquisition System Based on FPGA/SoPC", *In Proceedings. International Conference on Electronic Measurement and Instruments*, pp. 21-27, 2011.

[2] Dalay, B. "Accelerating system performance using SOPC bu ilder." System-on-Chip, 2003. *International Symposium on IEEE*,2003.

[3] F. M. Sun, et al., "FPGA-based Embedded System Design," *In Proceed-ings. IEEE Asia Pacific Conference on Circuits and Systems (A pccas 2008)*, Vols 1-4, pp. 733-736, 2008

[4] J.O. Hamblen, Using an FPGA-based SoC approach for senior design projects, *In Proceedings. Int. Conf. Mircroelectronic Systems Education*, pp. 18-19, 2003

[5] Altera SOPC Builder datasheet, Altera Corporation, San Jose, CA, De-cember 2010

[6] Nios II Hardware Development tutorial, Altera Corporation, San Jose, CA, May 2011
[7] Avalon Interface Specification, Altera Corporation, Sa n Jose, CA, August 2010
[8] www.uClinux.org, uClinux resource website
[9] wwww.alteraforum.com, forum for issues and solutions for Altera products
[10] www.fs2.com