IOSR Journal of VLSI and Signal Processing (IOSR-JVSP)
Volume 4, Issue 2, Ver. II (Mar-Apr. 2014), PP 01-08
e-ISSN: 2319 – 4200, p-ISSN No. : 2319 – 4197
www.iosrjournals.org

Development of DMA controller for real time data processing in FPGA based embedded application

Abhisek Bakshi[1], Aditi Burman[2] , Dr. Amlan chakraborty[3,]
[1]*Assistant Professor, Department of Computer Science and Engineering, Bengal Institute of Technology, Kolkata, India*
[2]*Associate consultant in Physical Design Domain(VLSI), Infosys, Bangalore, India*
[3]*Assistant Professor, A.K. Chowdhury school of Information technology, University of Calcutta, India*

Abstract: *In present day technology there is an immense need of developing suitable data communication interfaces for real time embedded systems. Field Programmable Gate Array (FPGA) offers various resources, which can be programmed for building up an efficient embedded system. In recent years the SOC (System on Chip) design eg, in media processing [1] is becoming more and more important in real time embedded applications as SOCs require low power, low area but are still capable of implementing various complex functionalities. In order to achieve SOC architecture, which can run a real time application, we need to develop high-speed data interfaces of the system with the external world through its various I/O ports. The DMA controller, which sends the data from I/O to memory and vice-versa without intervention of the processor, thus plays a vital role in these systems in order to achieve faster I/O data transfer. This paper proposes a technique to implement a DMA controller core on Spartan 3A FPGA hardware, which serves as an essential component for developing a real time data acquisition and processing system.*
Keywords: *FPGA, EDK, DMA controller, ADC, DAC*

I. Introduction

Whenever data is to be transferred from an I/O to a memory, first the processor read the data from the source address and then writes it to the proper destination address. This leads to the wastage of CPU cycles just for data transfer rather than processing. In many applications like image and video processing, where data needs to be transferred frequently from I/O to memory, if the processor is involved in the data transfer operation the throughput and overall system performance may degrade. That is why in those cases we use another controller; called DMA controller is needed, which is responsible for transferring the data without the intervention of CPU. In this paper we have tried to implement a DMA controller core to transfer real time data from I/O to DDR2 SDRAM in the Spartan 3A starter kit.

There are many ways to implement a specific application specific system design, i.e. either with ASIC, microcontroller, microprocessor and Field Programmable Gate Array (FPGA). But the reasons behind the choosing of FPGA are re-configurability, low power consumption and high speed compared to microcontroller. While making a System on Chip (SOC) the Dynamic Memory Access (DMA) controller plays an important role for the data transfer operation between I/O and memory. If large number of data byte comes from different sources, large processor cycles are wasted for the data transfer operation. Thus here we have tried to develop a DMA controller, so that the processor can involve with its own work.

In our work we have sent a real time analog signal to the DDR2 SDRAM through the DMA controller. The analog data form function generator is first converted to corresponding digital bits and then it is transferred to DDR2 SDRAM through DMA controller. After processing the data, taken from memory it as again converted to onboard Digital to analog converter. The output analog signal was shown in the digital CRO.

II. Design Overview

The work include both Analog to Digital and Digital to Analog conversion using LTC 1407 Analog to Digital converter and LTC 2624 Digital to analog converter respectively. Spartan 3A FPGA board has been used for the hardware verification. The analog signal is taken from a function generator and is converted to digital form using Spartan 3A FPGA board. The data kept in the memory through DMA controller. To verify the correctness of the data stored in the memory by DMA controller, it was converted back to analog signal again using LTC 2624 Digital to Analog converter and shown in a digital CRO as well as in the HyperTerminal.

Hardware Architectural Design

This work is implemented using the Xilinx EDK 11.1 (version) and Xilinx Spartan 3A FPGA prototyping board has been used for the hardware implementation and testing. Using the Xilinx platform studio

from EDK (Embedded Development Kit) the hardware portion of the embedded system has been developed. A soft core 32-bit RISC processor Micro Blaze has been used as a CPU for this embedded computing unit and all the required soft core peripherals are UART (used for RS232 Data Circuit- Terminal Equipment port), GPIO (General Purpose I/O) to control different signal lines of onboard ADC and DAC, MPMC (Multi Port Memory Controller) as the DDR2 SDRAM controller and DMA core. The blocks used to build up the FPGA based embedded computing unit is shown in Figure 1.

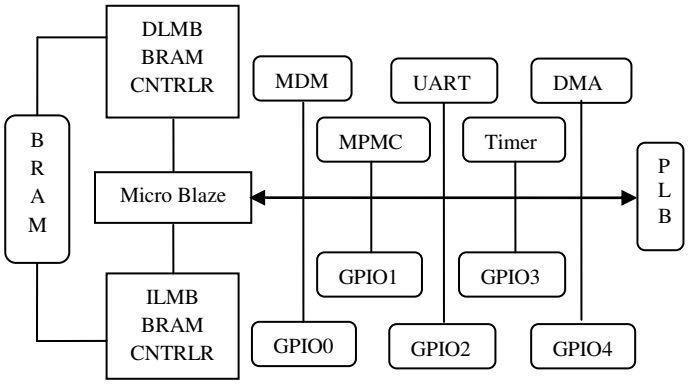


Figure 1: Block diagram architecture of FPGA System [2]

The top level view of the designed system for real time data processing is shown in Figure 2. The function generator, source of the real time analog signal is connected to the input pin of the onboard ADC. The digital output data is stored in memory through DMA controller soft core. The data after processing is converted to digital again using onboard DAC. The output pin of the DAC is connected to the Digital CRO, where we can see the output analog signal.

The necessary software for this design is written using the feature-rich C/C++ code editor and compilation environment provided within the EDK (Xilinx Embedded Development Kit)and SDK (Xilinx Software Development Kit). SDK works with hardware designs created with Xilinx Platform Studio (XPS) [11]. The implementation system setup is shown in Figure 2.

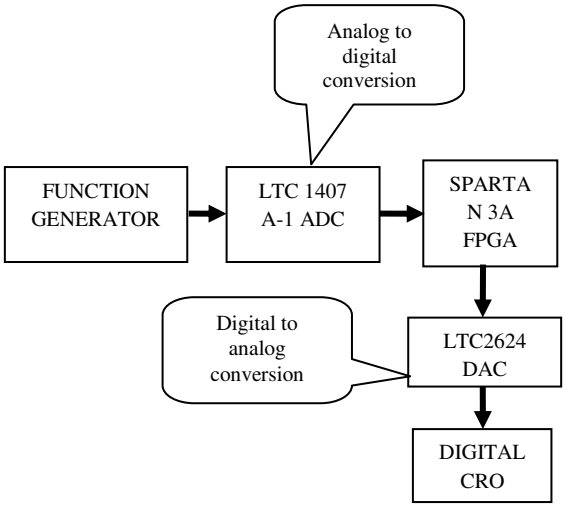


Figure 2: Architectural View of the System[3][4]

Analog To Digital Conversion

The analog capture circuit converts the analog voltage on to a 14-bit digital representation, D[13:0], as expressed by Equation[5]

$$D[13{:}0] = GAIN \times \frac{(V_{IN} - 1.65V)}{1.25V} \times 8192$$

The GAIN is the current setting loaded into the LTC 6912-1 programmable pre-amplifier. The reference voltage for the amplifier and the ADC is 1.65V, generated via a voltage divider. Consequently, 1.65V

is subtracted from the input voltage on input pins. The maximum range of the ADC is ±1.25V, centered on the reference voltage, 1.65V. Hence, 1.25V appears in the denominator to scale the analog input accordingly.

Finally, the ADC presents a 14-bit, two's complement digital output. A 14-bit, two's complement number represents values between -213 and 213-1. Therefore, the quantity is scaled by 8192, or 213.

Digital To Analog Conversion

Each LTC 6912-1 DAC output level is the analog equivalent of a 12-bit unsigned digital value, D[11:0], written by the FPGA to the DAC via the SPI interface. The voltage on a specific output is generally described in equation below.

The reference voltage, VREFERENCE, is different between the four DAC outputs. Channels A and B use a 3.3V reference voltage. Channels C and D have a separate reference voltage, nominally also 3.3V, supplied by the LP3906 regulator designated as IC18. The reference voltage for Channels C and D can be modified. The reference voltages themselves have a ±5% tolerance, so there are slight corresponding variances in the output voltage.

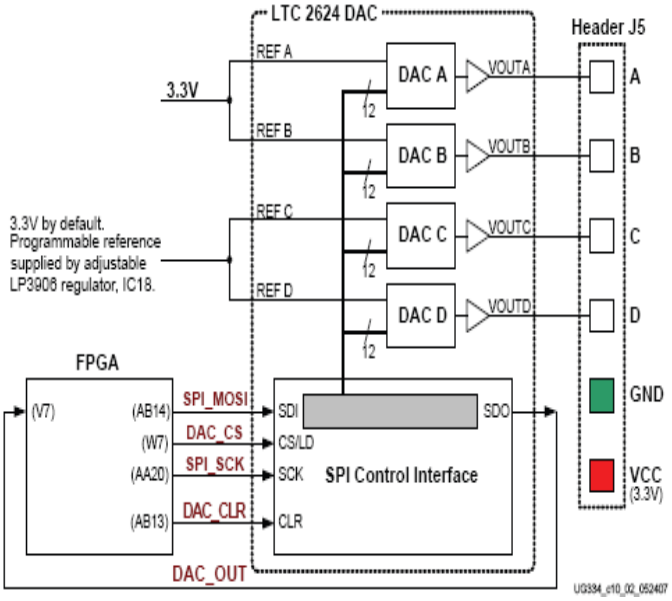


Figure 3: Digital-to-Analog Connection Schematics[5]

$$V_{OUT} = \frac{D[11:0]}{4096} \times V_{REFERENCE}$$

LogiCORE IP XPS Central DMA Controller (v2.03a)

The XPS Central DMA Controller provides simple Direct Memory Access (DMA) services to peripherals and memory devices on the PLB. The controller transfers a programmable quantity of data from a source address to a destination address without processor intervention.

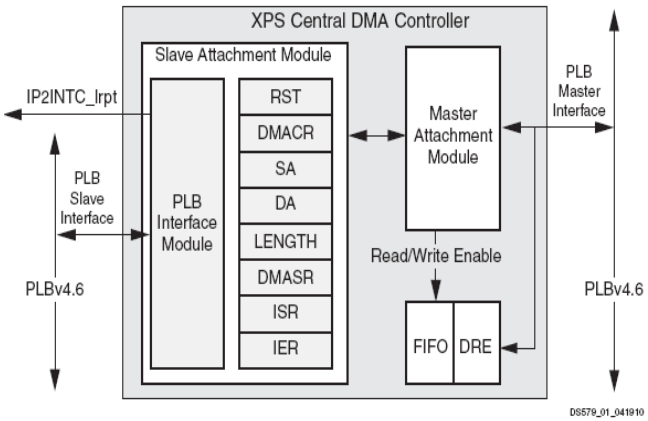


Figure 4: Block Diagram for the XPS Central DMA Controller[5]

System Design

For the verification of the Successful transfer of data through DMA controller the two GPIO core, one timer core, a DMA controller core and DDR2 SDRAM has been included in the system design. The two GPIO cores are used to control the onboard LTC6912-1 programmable preamplifier and LTC1407A-1 Analog to Digital converter chip respectively. The timer core has been used for delay as SPI clock frequency is not exactly equals to board clock frequency. The GPIO, timer, memory, DMA controller and the processor share the same bus in the system. The DDR SDRAM is controlled by MPMC (Multi Point Memory Controller) core. GPIO_1, which controls the serial data output line of the A/D converter, has been used as the source and DDR2 SDRAM has been used as the destination of the data transfer. The design view, shown in Fig. 34, has been implemented using Xilinx Platform Studio (11.1), shows the connection between Processor, DMA core, GPIO and DDR SDRAM (MPMC).

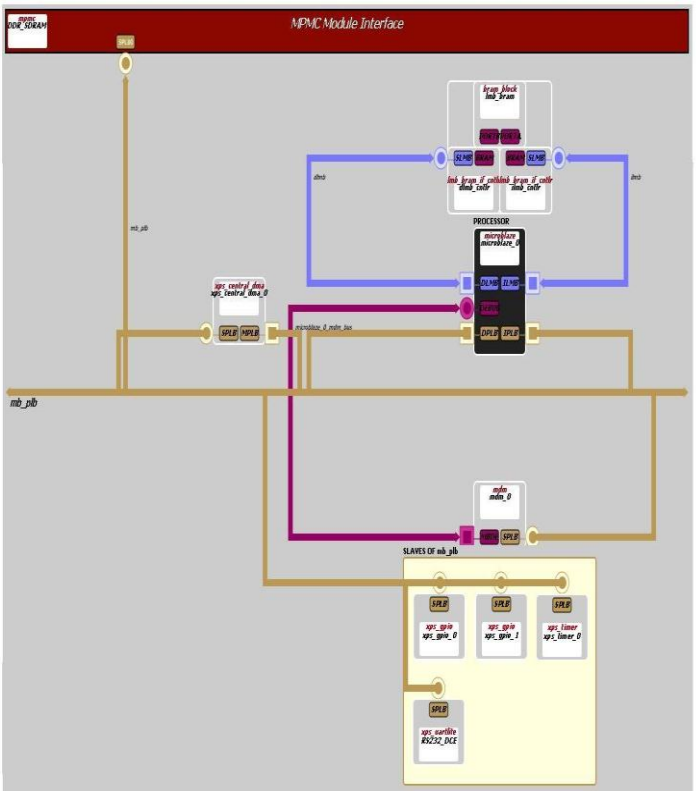


Fig 5: shows the connection between processor core, DMA core, GPIO core and DDR2 SDRAM (MPMC).

III. Algorithm For Transfer Operation

Following the timing diagram of the LTC6912-1 amplifier and LTC1407A-1 analog to digital converter firstly the gain is set in the amplifier. In the LTC6912-1 amplifier chip there is a inverting op-amp and a SPI interface is present. The gain is needed to set by send data bit serially from FPGA to SPI interface in the rising edge of the SPI clock. The op-amp gets 1.65 volt at the positive side by a voltage divider from 3.3 volt onboard supply. The op-amp is used to amplify the analog signal to the LTC1407A-1 onboard Analog to Digital converter. The LTC1407A-1 automatically produces the corresponding 14 bit digital value to the corresponding channel of the SPI interface of LTC1407A-1. Then the digital output is send bit by bit to the memory through DMA controller in the rising edge of the SPI clock. The whole operation is depicted through a flow diagram shown in figure 35 below.

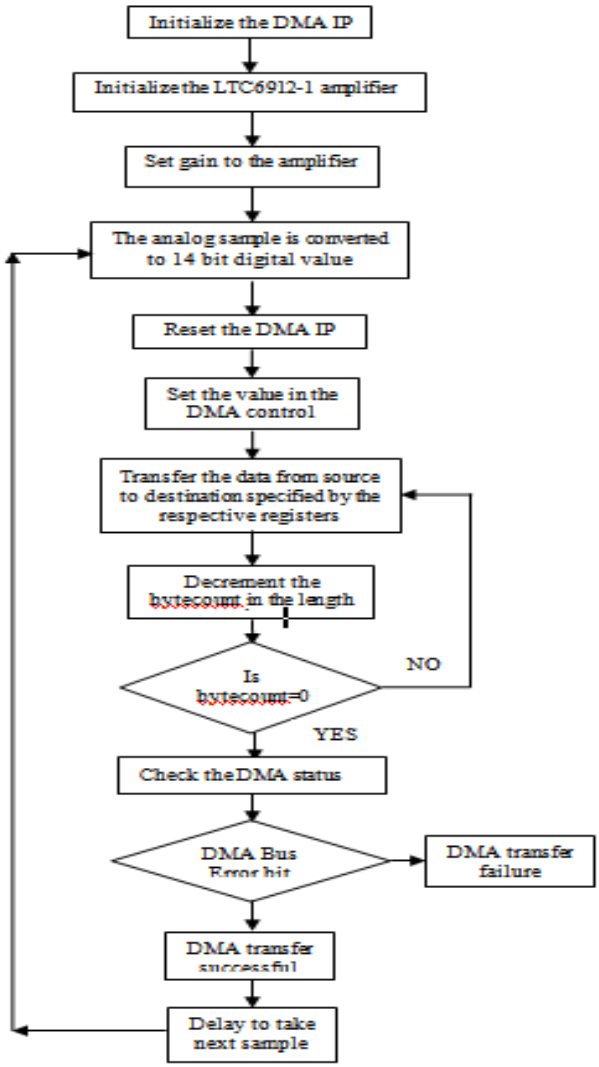


Figure 6: Flow Diagram of DMA transfer implementation [6]

VI. Experimental Setup Snapshot

The snapshots in the figure 36 below show the experimental setup for the successful verification of data transfer through DMA controller core of a real time signal.

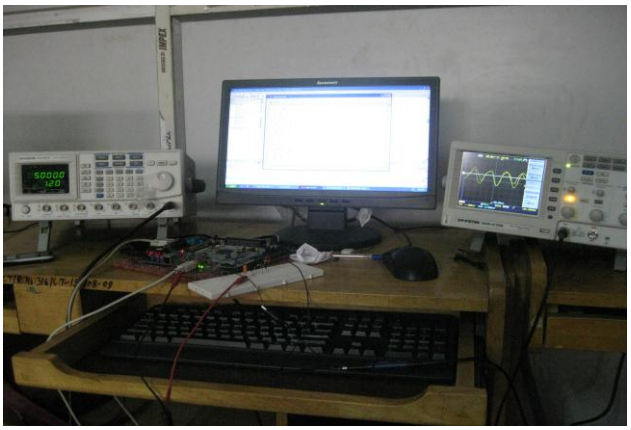


Fig 7: Run time verification of the design

V. Experimental Results

The snapshot, shown in Figure 37 has been taken from the Hyper Terminal of the computer, which was connected with our FPGA device through the RS 232 serial link, this arrangement was done for verifying our design. For the verification of the operation, an analog signal of 10 KHz of pick to pick voltage 1.2 volt was taken as sample analog input signal which was connected to the input of the Analog to Digital converter via a DC blocking capacitor. Here the real time analog signal is converted by the Analog to digital converter and was sent to the memory.

In the snapshot you can see that, six columns have been printed for every bit of data transfer. The first value shows the bit position of the converted digital value. The next one shows the previous value stored in the memory address, shown in fifth column. The third column shows the current value of the same address after the data transfer operation completed. After every bit of transfer the destination address of the memory is automatically increased. To get the proof of the DMA transfer operation DMA status register's value has been checked during the transfer operation, which showed that DMA was busy during transfer operation and no DMA bus error was occurred.

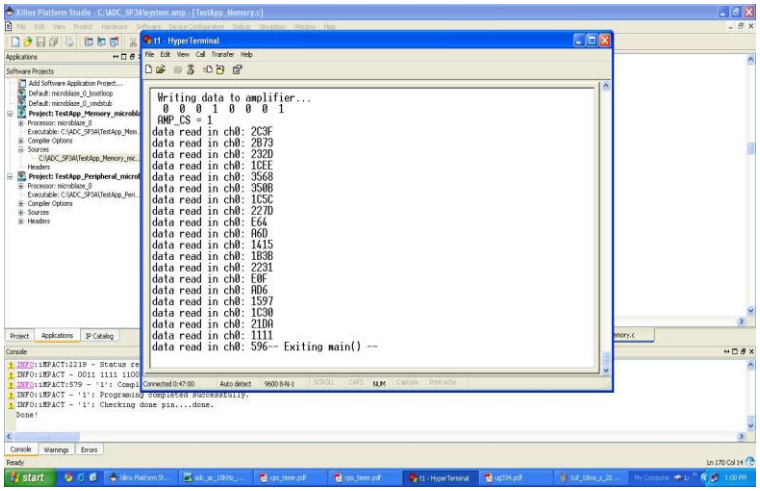
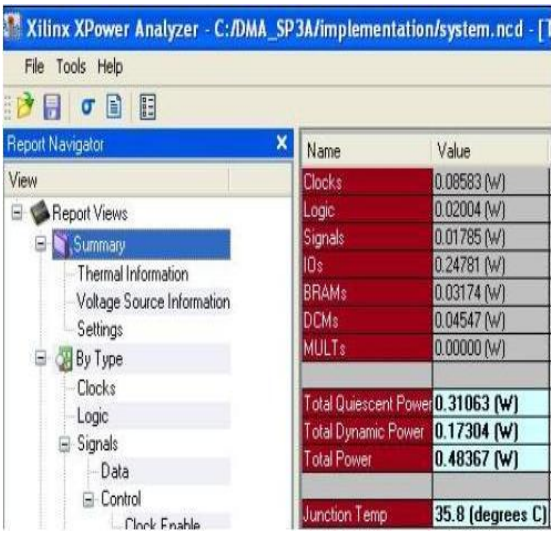


Fig 8: Experimental data view in the Hyper terminal

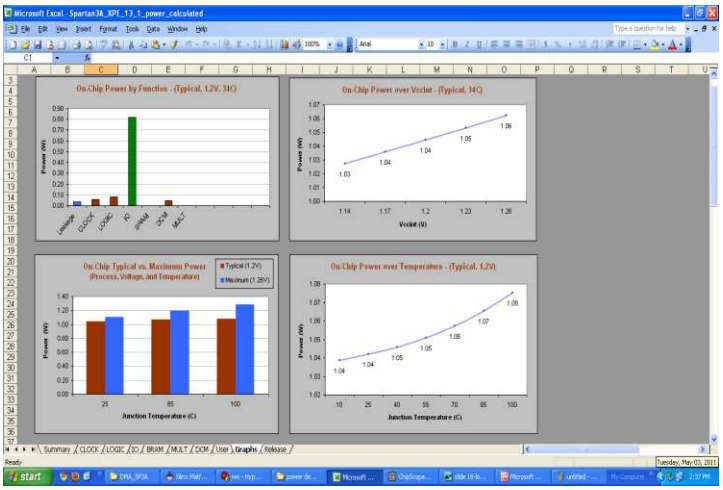
VI. Experimental Summary

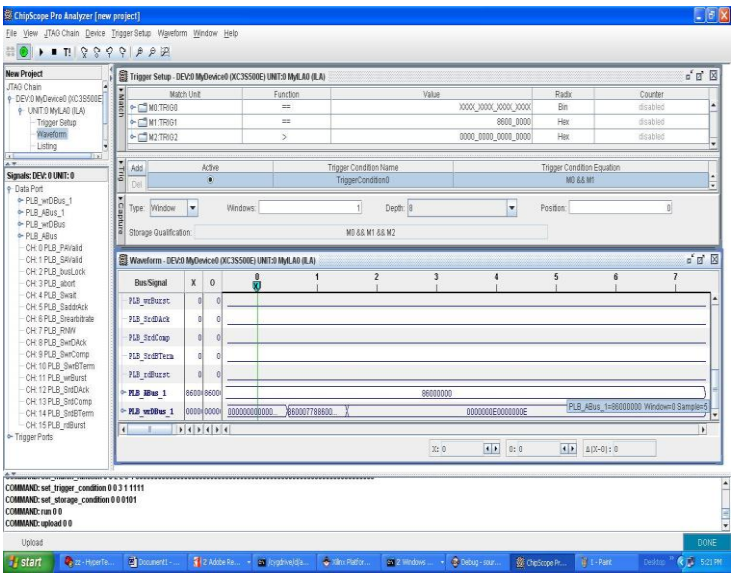
The power analysis shows that total dynamic power 0.17304 W and total power 0.48367 W has generated. The junction temperature was 35.8 degree Celsius. The statistics of the design and the device utilization summary of the design are shown in figure below.




```
Timing summary:
---------------

Timing errors: 0  Score: 0
(Setup/Max: 0, Hold: 0)

Constraints cover 222206 paths,
52 nets, and 26523 connections

Design statistics:
   Minimum period:  15.918ns{1}
   Maximum frequency:  62.822MHz
   Maximum net delay:   2.742ns
```

Device Utilization Summary			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Flip Flops	1,559	11,776	13%
Number of 4 input LUTs	2,269	11,776	19%
Number of occupied Slices	1,750	5,888	29%
Number of Slices containing only related logic	1,750	1,750	100%
Number of Slices containing unrelated logic	0	1,750	0%
Total Number of 4 input LUTs	2,359	11,776	20%
Number used as logic	1,876		
Number used as a route-thru	90		
Number used for Dual Port RAMs	256		
Number used as Shift registers	137		
Number of bonded IOBs	10	372	2%
IOB Flip Flops	8		
Number of BUFGMUXs	3	24	12%
Number of DCMs	1	8	12%
Number of BSCANs	1	1	100%
Number of BSCAN_SPARTAN3As	1	1	100%
Number of MULT18X18SIOs	3	20	15%
Number of RAMB16BWEs	8	20	40%
Average Fanout of Non-Clock Nets	3.56		





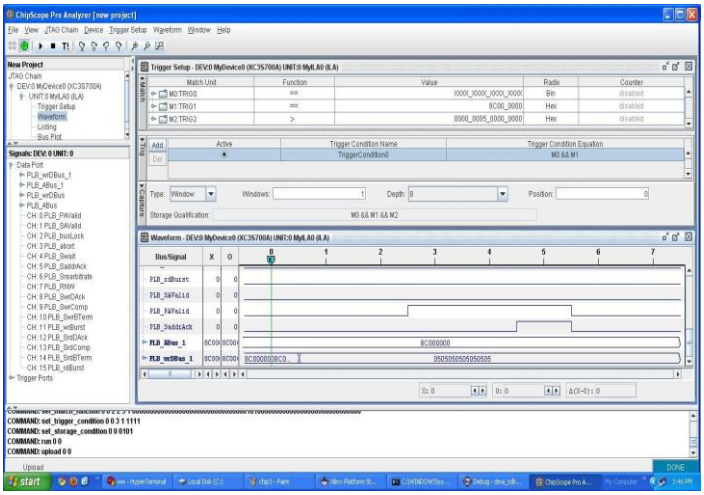


Fig 9: Experimental summary and run time simulation snapshots

VII. Conclusion

The primary goal of this work was to perform a real time data transfer between I/O and memory without the intervention of the processor core and providing a faster processing time. In future we want to perform the communication between two FPGA real time data transfer operation of audio data for voice messaging applications, where analog to digital conversion of the real time audio data is necessary and we also wish to perform the real time implementation of security protocols using FPGA processor cores.

Acknowledgement

This research work is being performed as a collaborative research activity between the Techno India , Salt lake, India, Bengal Institute of Technology, Bantala, Kolkata, India and A.K. Chowdhury School of I.T., University of Calcutta.

References

Journal papers:

[1] Keming Chen, Lingling Qi, Haibin Yu- DOI 10.1109/IITA.2008.493- "Design of Two-Dimension DMA controller In Media Multi-Processor SoC".

[2] Suman Sau, Chandrajit Pal, Amlan Chakrabarti- "Design and Implementation of Real Time Secured RS232 Link for Multiple FPGA Communication".

[6] Implementation of High Speed Real Time Data Acquisition and Transfer System.

[7] An Improved DMA Controller for High Speed Data Transfer in MPU Based SOC.

Books:

[3] Linear Technology LTC2604/LTC2614/LTC2624 BLOCK DIAGRAM FEATURES APPLICATIONS DESCRIPTION Quad 16-Bit Rail-to-Rail DACs in 16-Lead SSOP.

[4] Linear Technology LTC1407-1/LTC1407A-1 BLOCK DIAGRAM DESCRIPTION Serial 12-Bit/14-Bit, 3MspsSimultaneous Sampling ADCs with Shutdown.

[5] Spartan 3A starter kit board user guide, UG334 (v1.1) June 19, 2008.