International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 4 Issue 08, August-2015

Implementation of Asynchronous Pipelines and Its Applications using FPGA

Sameena Begum
Dept. VLSI Design and Embedded Systems
VTU RC, Kalaburagi
Kalaburagi, India

Abstract— **Asynchronous designs have interesting features because of absence of the clock signals and it is another option while designing a digital systems. By overcoming the system timing overhead, asynchronous designs works at high speed and it provide high throughput, utilizes the dynamic power, and also provides the elasticity. However, out of several design styles, the most suitable designs for FPGA platforms are bundled data micro pipelines style, the reason behind it is simplicity of control. In this project, we propose pipeline architecture in a bundled data micro pipeline style to implement the asynchronous digital systems by targeting the FPGA devices. The execution of program or software is achieved using FPGA through parallel processing. The design summary of each architecture shows that the design architecture have better throughput by providing more number of input output bond, less delay by reducing the number of look up tables(LUT), and occupy less area by reducing the number of flip-flop's. The designed architecture is applied for UART design in order to increase the throughput and also for FIR filter design to reach the high efficiency.**

Keywords—2-Phase, MOUSETRAP, Asynchronous Pipelines,and Application

I.INTRODUCTION

Field Programmable Gate Arrays Devices (FPGA) has become a very popular way to design and implement the digital circuits, because of their cost and design time. A high performance FPGAs are mainly implemented in DSM-MOS technology i.e. Deep-sub micron technology were the delay on the lines are considered, which may be greater than the delay of logic gates.

In this technology clock signal requires extra attention due to electromagnetic interference, noise generation, and power consumption. Besides these factors, it is difficult task to distribute the clock signal along the chip which increases the complexity because of clock skew problem which reduces the performance of the system.

This technology reaches 130% in VLSI implementation which worsens the performance when FPGAs are employed. Most of the digital systems are designed in synchronous paradigm i.e. they use global clock signal to synchronize their operations. They are well known because of the availability of CAD tools for automatic synthesis and simplicity of design.

A pipeline is a set of data processing elements that are connected in series, were the input of one element is the output from previous element. When elements of pipeline connected in time sliced fashion in that case some amount of storage is required.

Asynchronous pipelines use hand-shaking protocol to avoid the problem of switching power, switching noise, and clock skew which is generated by the global clock signal which is usually found in synchronous designs. The asynchronous circuit design proves better design for overcoming synchronous timing overhead, power requirement and the problem due to clock skew without affecting the functionality of the system.

In this design style it is easier to implement the micro pipeline style in commercial FPGAs due to the simplicity of its control. This is very important because it is difficult to obtain the hazard-free asynchronous controllers especially at this platform. Different architectures of linear micro pipeline design have been introduced; most of them target the VLSI implementations that may employ full-custom design. In FPGA oriented pipelines one uses the delay mechanism and other does not obey the fundamental mode.

In this project we propose a FPGA oriented asynchronous pipelines, it consists of flip-flops as a registers due to large availability of flip-flops in FPGA. Compare to other pipeline architecture, the control unit of this architecture is simple because it is based on the logic gates and it is mapped into two look-up-tables. This proposed architecture gives high throughput and reduces the latency compare to MOUSETRAP architecture. To increase the throughput of serial communication this architecture is used.

II. Basics of Project

A. Comparison between Synchronous and Asynchronous Pipelines

The fundamental difference between the asynchronous and synchronous pipelines is highlighted in the Fig 1.

The four important features for asynchronous approach to provide flexibility and modularity are as follows:

- *Delay between each stage is unequal:* In synchronous systems, the worst-case delay of every stage must be less than the clock period and all the stages must operate at the same fixed rate. In contrast, in asynchronous systems, the balanced stages tend to provide high system performance, and it is not a requirement for correct operation of system. As a result, stages are concatenated to form a working system with different static delays.

In some asynchronous pipeline styles, each stage has a dynamically variable delay; this is mainly found in asynchronous adder that has data-dependent completion. Hence, this per-stage variable delay tends to improve the throughput and average system latency.

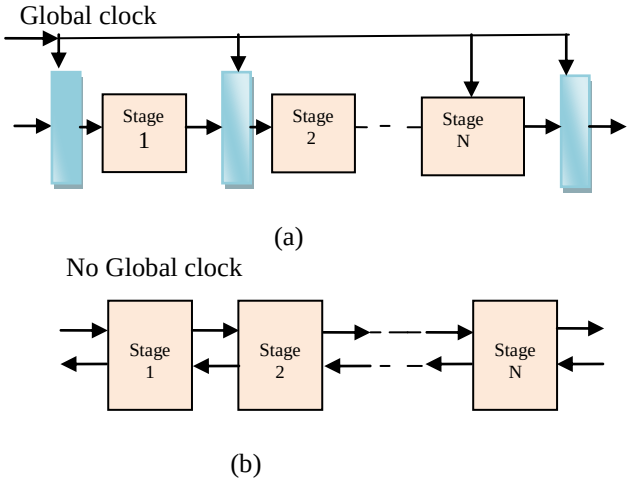

Fig 1: (a) Synchronous pipeline (b) Asynchronous pipeline

- *Asynchronous pipelines provide elasticity:* Unlike in synchronous pipeline, this pipeline has a variable number of data items that can appear in the pipeline at any time. When the data items are injected at wide intervals and if there is no congestion then the data items are widely spaced in the pipeline and travel rapidly through pipeline. The spacing becomes tighter between the data items when the input rates are higher.In case, with a slow or stalled output environment, the data item becomes bunched at close intervals. In all other cases, the data items are processed and arrived even with an irregular or unknown arrival rate and there is no wait for a next clock edge. Hence the throughput rate and the spacing between the data items are determined dynamically.

- *Asynchronous pipelines automatically provide flow control:* The hand-shaking protocol provides overflow and underflow protection, even in variable-delay environment. In contrast, a synchronous pipeline does not provide the flow control. It requires credit-based techniques that needs registers and a stall signal must be synchronized to the clock at every stage to support the flow control.

- *Asynchronous pipelines consume dynamic power only on demand:* The switching activities occur only when data items being processed otherwise the stages and their control is quiescent. This approach achieves the advantage of automatic clock gating without extra instrumentation at arbitrary levels of granularity in the design including the rapidly changing traffic patterns. These pipelines eliminate the need for global clock distribution. The control for asynchronous pipelines operates in distributive manner which eases for defining the each stage's performance only by local parameters.

B. Linear and Non-linear Pipelines

In a linear pipeline, a series of data processing stages arranged linearly to perform a specific function over a data stream. Whereas a non-linear pipeline is also called as dynamic pipeline configured to perform various functions at different rate, there is also feedback or feed forward connection.

C. Asynchronous Communication Protocol

The transition signaling is used to provide the communication between sender and a receiver and there will be many data wires and two control wires between them as shown in Fig 2.

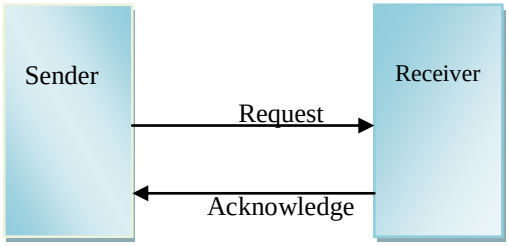

Fig 2: Communication between sender and receiver using asynchronous protocol

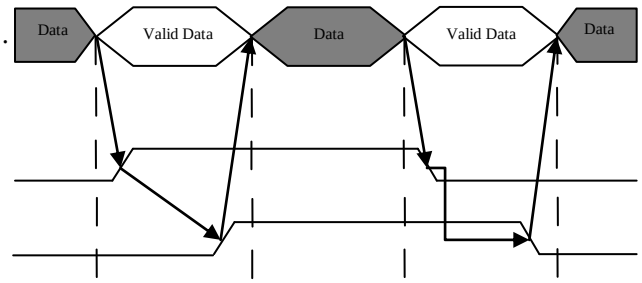

Fig 3: 2-Phase Hand-Shaking Data Convention

In the 2-phase bundled data convention, the three events occur cyclically in the sequence as shown in above Fig 3. They are:

- During sender's active phase, the sender may change the data as shown in Fig 3 as arrow, producing the request event, after establishing a correct data value.

- During the receiver's active phase, the sender must hold the data constant, which is shown with dotted arrows.
- The receiver ends its active phase by producing the acknowledge event, after receiving the valid data. As the controlling unit decreases, either phase may last for as much as less time, and the request-acknowledge events are alternate.

III.DESIGN METHODOLOGY

Minimal Overhead Ultra High Speed Signal Transition Asynchronous Pipeline (MOUSESTRAP)

Various pipeline architectures have been proposed but in this project we focus only on asynchronous pipeline. Among several architectures, MOUSETRAP architecture is one of the FPGA oriented asynchronous pipelines architecture, it has better performance and it is based on logic gates which can be implemented on FPGAs. The MOUSETRAP asynchronous pipeline architecture consists of control logic and processing logic as shown in Fig 4.

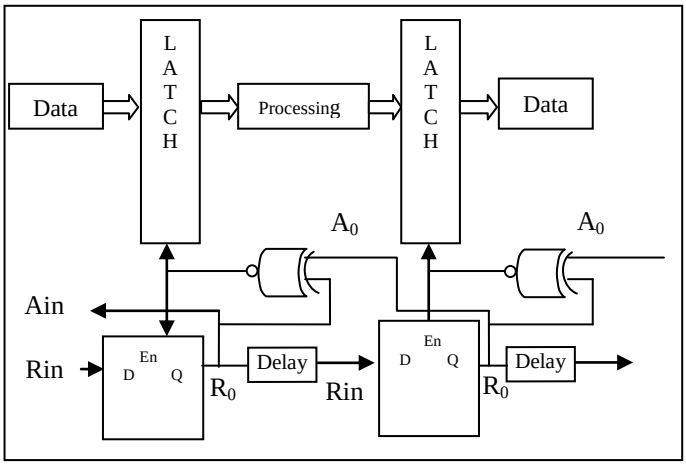

Fig 4: MOUSETRAP Architecture

For rise transition or first request on Rin, data starts flows through the pipeline. As a result R0 changes to one and the enable changes to zero which closes or lock the latch of particular stage in order to block any new request. The same process is applied to other stages which resembles as bundled form i.e. the data arrives before the request.

IV.PROPOSED ASYNCHRONOUS PIPELINES

In this architecture, the register is enabled by the latch. Initially the design was empty, but as soon as it receives the data on Rin (data-in) pin the corresponding latch enables the register to store the current data and to be processed. Simultaneously, the data is available on R0 (data-out) pin which act as input to the next stage after some delay.

Now the next stage is active and the previous stage is made inactive or blocked to avoid the further data-in until it receives the acknowledgement. Similarly the current stage copies the data on input pin to the output pin, this resemble an acknowledgement signal for the previous stage.

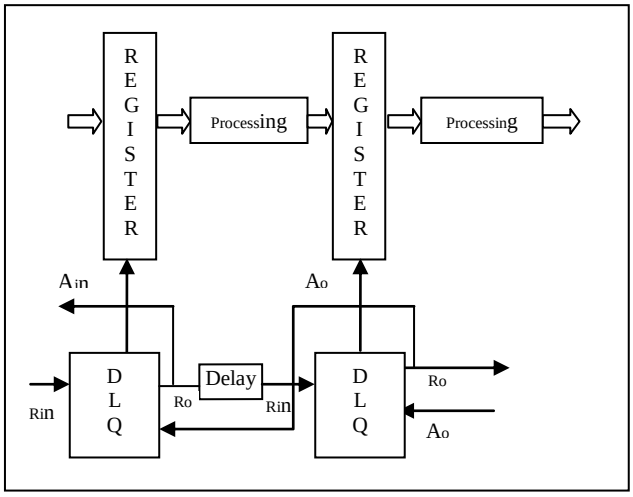

Fig 5: Proposed Asynchronous Pipelines

Finally, the data is processed stage by stage by processing logic and displays on output pin of processing block. Here the design for control is simple and hence reduces the circuit complexity, design area and designing efforts. The Fig 5 represents this architecture.

V.APPLICATION

A. Asynchronous Pipelines for UART

A UART is a device that allows the transmission and reception of information in a serial and asynchronous way. The data is transmitted as a 11-bit frame: 1-bit for start bit, 1-bit for parity, next 8-bits are reserved for data, and the last bit indicates the stop bit. The whole structure of UART that is shown below is made to pipeline the data, to increase the throughput.

- *Transmitter Section*

  It consists of an 8-bit THR (transmitter hold register) and TSR (transmitter shift register) and a FIFO as shown in Fig 6.

  The LSR is set to '1' when THR is empty.

  The transmission begins after the data is loaded into THR, at this time LSR is set to '0'.

  The data bits in TSR are shifted out with a word length that is defined in LCR followed by a parity bit if enabled.

  The next frame is send immediately after a first frame is get fully transmitted if the THR is not empty.

  This back-to-back transmission of data frames increases the transmission bandwidth and hence the throughput.

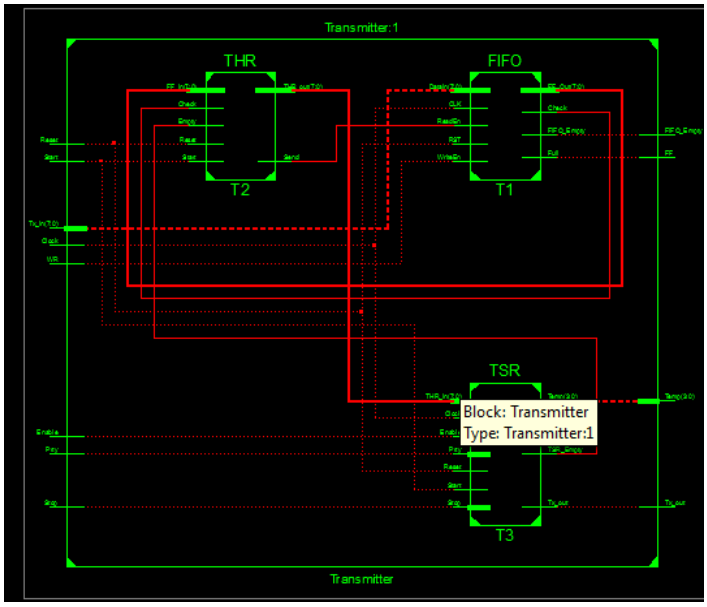

Fig 6: RTL Schematic at Transmitter section of UART

- *Receiver Section*

  It consist of an 8-bit RHR (receiver hold register) and (receiver shift register) RSR, sampling block and a status register. Fig 7 is a RTL schematic at receiver section which shows that the data is converted from serial to parallel form.

  The main function of sampling block is to check the entire data received properly or not by providing acknowledgement if not, an error is detected by this status register.

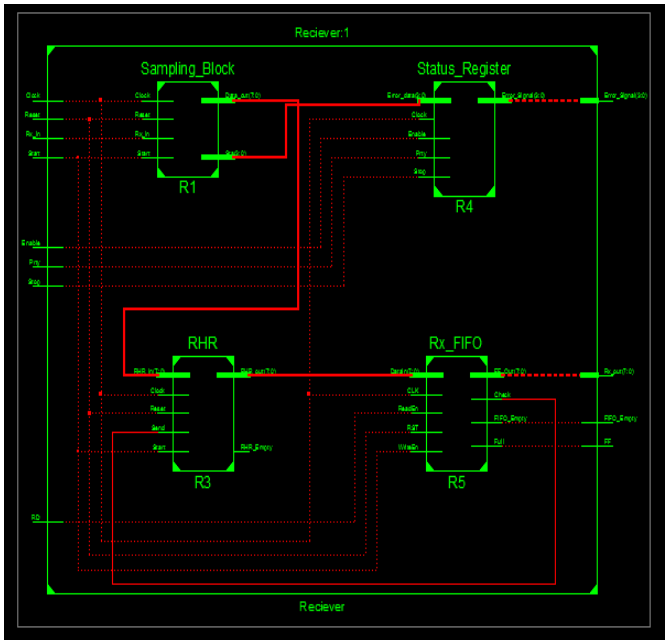

Fig 7: RTL Schematic at Receiver section of UART

D. Asynchronous Pipelines for FIR Filter

In Finite impulse response (FIR) filter, the feedback is absence which guarantees the impulse response to be finite.

- *Impulse Response:* It is just the set of FIR filter coefficient. If these impulses are applied to the FIR filter that consists of '1' sample followed by many '0' samples then the filter output is a set of coefficients this is due to the '1' sample moves past each coefficient in turn form the output.

- *Filter Tap:* It is simply a delay pair/coefficient. The number of filter tap indicates the number of calculation to be performed, the capacity of filter to perform filtering, and the amount of memory required to implement the filter i.e. to save the filter co-efficient.

- *Multiply and Accumulate (MAC):* The multiplication of sampled data and its corresponding coefficient and accumulation of result of multiplication is done in this unit.

A 60-tap asynchronous FIR Filter was designed as shown in Fig 8 it represent the working of a pipeline architecture in FIR Filter according to the basic equation of FIR Filter which is shown below:

$$Y[n] = \sum_{i=0}^{4} x[n-i] \times h[i]$$

Where h[i] is a FIR Filter coefficients, x[n] and y[n] are the input and output signal samples.

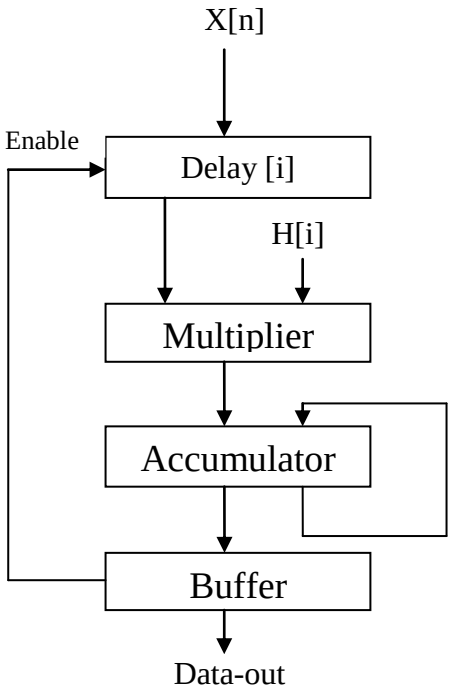

Fig 8: Block diagram of asynchronous pipeline for FIR Filter

VI. SIMULATION RESULTS

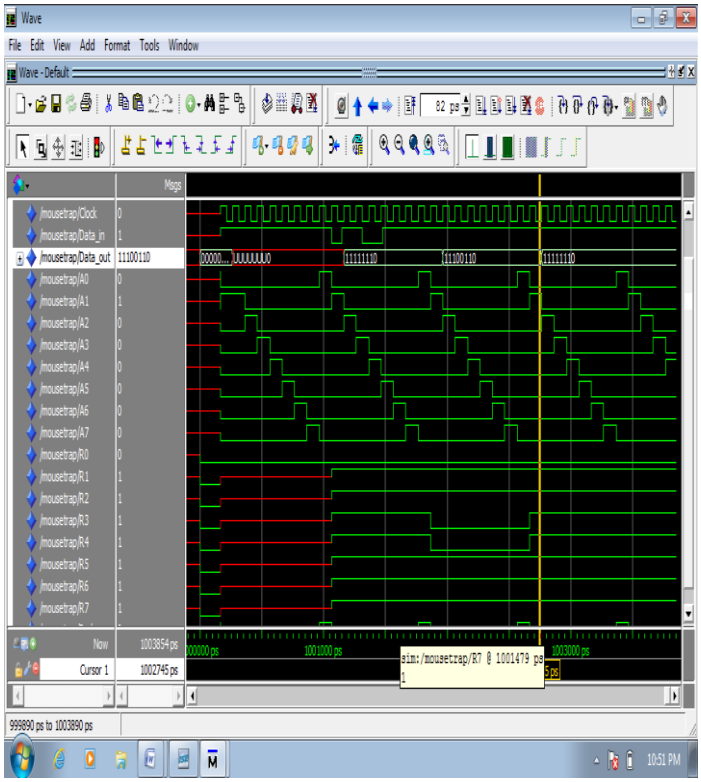

Fig 9: Simulation results for MOUSETRAP

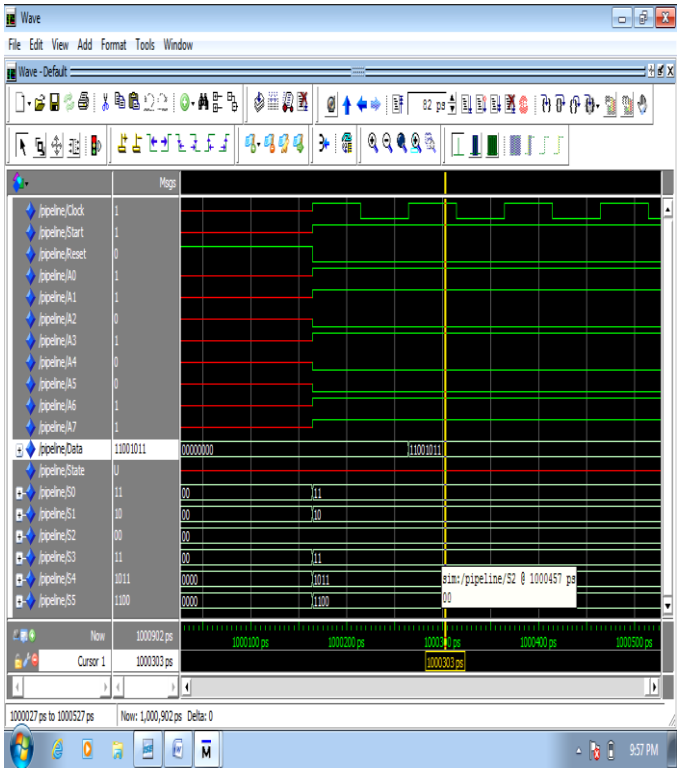

Fig 10: Simulation Result for proposed Pipeline Architecture

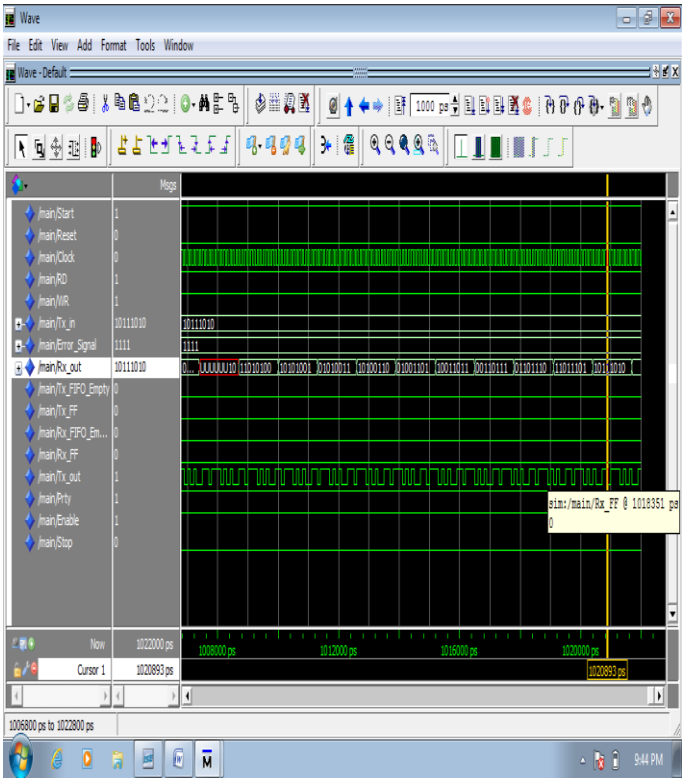

Fig 11: Simulation Result Asynchronous Pipelining for UART

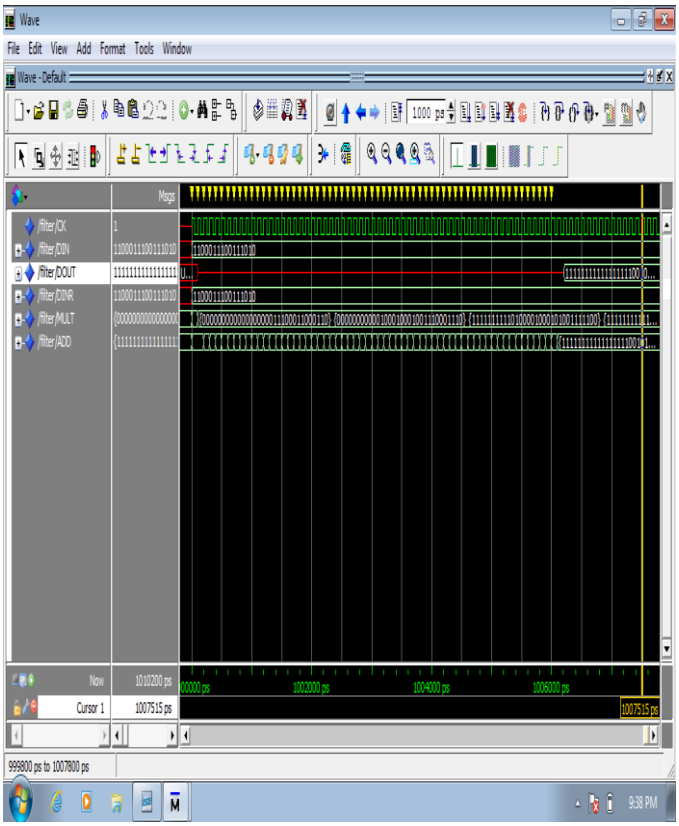

Fig 12: Simulation Result of Asynchronous Pipeline for FIR Filter

CONCLUSION

A MOUSETRAP architecture is implemented which is a high speed signal transition design and it provides high performance. In this project, the proposed design is FPGA oriented linear micro-pipeline architecture. This architecture posse an important advantages due to its simplicity of control and provides greater throughput compare to MOUSETRAP. The implementation of control for the proposed pipeline is free of essential hazard. This project focuses on increase in throughput, reducing design area and delay using 2-phase handshaking protocol. All the code was written in VHDL language.

In Future:

- We can use this for designing a high performance asynchronous processors.
- This can be used for Ethernet routing chips, very low power or robust designs such as sensors.
- We can also use in mixed-timing environment and also for high speed FPGA's.

RESULTS AND DISCUSSIONS

Table 1: Comparison between MOUSETRAP and Proposed Architecture

	MOUSETRAP	Proposed
Number of Flip-Flops	32	24
Number of LUTs	26	1
Input /Output Bonds	12	19

As the numbers of Flip-Flops are decreased in proposed architecture, area occupied by the design is reduced and increases the computation speed by storing the previous data. The reduced number of LUTs shows the reduction of delay, and increase in input-output bonds increases the throughput.

ACKNOWLEDGMENT

I thank to my parents whose love and affection has made me reach here. I thank to my guide and friends.

REFERENCES

[1] D. Goldhaber-Gordon, et al., "Overview of Nanoelectronic Devices,"*Proc. of the IEEE*, vol. 85, No. 4, pp.521-540, April 1997.

[2] J. J. Rodriguez, et. Al., "Features, Design Tools, and Applications Domains of FPGAs", *IEEE Trans. on Industrial Electronics*, vol. 54, No. 4, pp.1810-1823, August 2007.

[3] J. Cortadella, et al., "Coping with the variability of combinational logicdelays," *ICCD*, pages 505–508, 2004.

[4] C. J., Myers, "*Asynchronous Circuit Design*", Wiley & Sons, Inc., 2004, 2a edition.

[5] J. Sparso e S. Furber, "*Principles of Asynchronous Circuits Design*", Kluwer Academic Publishers, 2001

[6] I. E. Sutherland, "Micropipelines", *Communication of the ACM*, vol. 32, No.6, pp.720-738, June, 1989.

[7] S. B. Furber et al., "Four-Phase Micro pipeline Latch Control Circuits," *IEEE Trans. on VLSI Systems*, vol.4, no. 2, pp.247-253, June, 1996.

[8] G. S. Taylor and G. M. Blair, "Reduced Complexity Two-Phase Micro pipeline Latch Controller," *IEEE Journal of Solid-State Circuits*, vol. 33, Nro. 10, pp.1590-1593, October, 1998.