Design and Implementation of an Asynchronous Controller for FPGA Based Biosignal Processing

T.N.Prabakar
Saranathan College of Engineering
Tiruchirappalli

G. Lakshminarayanan
National Institute of Technology
Tiruchirappalli

ABSTRACT
In a clause of combinational circuits, the throughput can be increased, without (wave) pipelining, by introducing data dependent delay feature thus avoiding the worst case delay. That is, in circuits like multipliers and adders which are the basic building blocks of any DSP system [1], the processing delay can be varied according to the magnitude of the input data. This makes the circuit asynchronous and necessitates a controller to arbitrate the data. Systems like FIR filters, where a series of combinational multipliers are used, can be asynchronously pipelined with a controller regulating the data between stages. With this system level pipelining, speed of circuit level pipelining can be achieved provided the data are of low magnitude. In this paper, two controller architectures are presented to regulate the data flow between asynchronously pipelined stages. Firstly, as a stepping stone, Altera's soft-core NIOS II processor is used and secondly, an exclusive asynchronous controller is designed using HDL. These controllers are designed to suit asynchronous implementation in conventional FPGAs, to effectively handle repeated data and to perform self-test. These controllers issue the control signals to the various dual edge triggered pipelined registers to process the data in both the edges for further improving speed. In the HDL version of the controller, programmable delays are generated by a 'logic locked' high frequency counter without using delay elements. To verify the efficacy of these controllers Hanning filter is implemented using Braun array multipliers and adders. Thus, this approach consumes lower power and achieves data dependent throughput and also avoids the need for global clock signals and skew problems.

General Terms
Asynchronous Pipelined Systems, FPGA

Keywords
Asynchronous, FPGA, Low power, Asynchronous Controller, Data Dependent Delay.

1. INTRODUCTION
The complexity in distributing a high speed accurate clock over a large circuit area without skew as well as the inherent high power consumption caused by the clock network has prompted designers to reconsider the role of asynchronous circuits. A synchronous combinational circuit works on worst case delay while an asynchronous combinational circuit works on average case delay. But, when a synchronous or asynchronous system is pipelined, both should work theoretically with the same worst case delay, even though the hand shaking signals in asynchronous system contribute some excess delay. In synchronous systems the clock frequency governs the speed of the whole circuit, which is determined by its worse case delay. On the other hand, asynchronous circuits rely on the use of completion-detection methods to determine when a combinational logic block has completed its operation. Several methods of completion detection are available in literature. Asynchronous systems, based on dual rail coding techniques carry the disadvantages of a very high hardware overhead coupled with low operation speed while the systems based on bundled delay approach fails to exploit the data dependency of internal delays[2]. Delay dependency is also attempted in [3] [4]. But, controller architecture to handle repeated data and delay mechanism is the novelty presented here. In addition, conventional FPGAs are suitable only for synchronous implementations and lack of proper CAD tools negate the use of asynchronous systems. In this paper, two controller architectures are presented to subdue the challenges of using conventional FPGAs for asynchronous applications. With a controller in the system, the following advantages are realized:

i. When same data is repeatedly coming, the present output is again latched at the output and immediately the next data is read. This avoids the repetition of same processing.
ii. Controller is sensitive to both the edges and return to zero transitions is avoided.
iii. Duplicating delay logic is removed by a 'logic locked' counter structure with a delay resolution of 4 ns.
iv. Fine tuning delay, prohibits glitches passing between stages. Hence, the reduction in dynamic power consumption is achieved.
v. The controller is a general purpose and can be used for any application and any number of stages.
vi. The controller is suitable to implement asynchronous systems in currently available synchronous FPGAs.
vii. The controller has the built-in self test feature.
viii. Problems with synchronous circuits like clock skew are avoided.
ix. If the circuit is not under the clause of data dependent delay, setting all the delays as same, will give a simplest asynchronous pipelined architecture suitable for conventional synchronous FPGAs.

x. In addition to the above, the controller has the benefit of all the advantages offered by the asynchronous systems.

2. DATA DEPENDENT DELAY

In DSP operations like correlation, convolution, and filter banks for multirate signal processing, multipliers and adders are used as a foundation blocks. Most of the arithmetic operations have the property of data dependent processing delay or can be designed to have this property. Out of various algorithms available for multipliers and adders, Braun array multiplication algorithm and ripple carry adder algorithm are chosen for implementation to realize data dependent delay feature [5]. The block diagram of an 8×8 pipelined Braun array multiplier is given in fig.1.

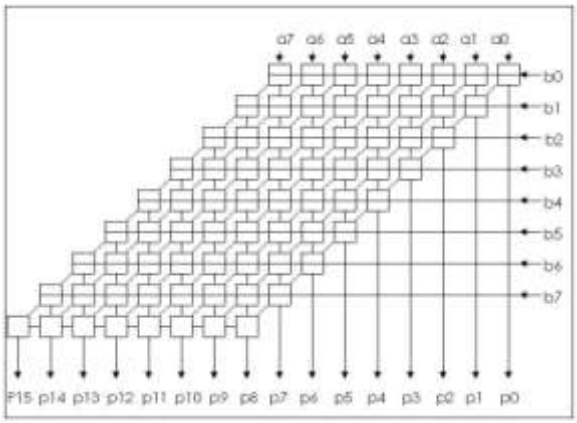

Fig 1. 8×8 pipelined Braun Array Multiplier

The process delay estimation is performed by finding the magnitude of the inputs. Referring to the 8×8 Braun array multiplier in Fig.1, if the inputs are "00000010" and "00000010" the output is "0000000000000100". In this case, bits up to P2 only are used. In that case, the delay is less. When the input data has all the 8 bits, then comparatively higher delay is needed to receive all bits up to P15. So, the position of first '1' from the MSB in both the operands decides how many bits are in the output and the processing delay. The following Table.1 depicts this relation. Hence, with a controller in the system, the dynamic data dependent delay control can be achieved.

Table 1. Data Magnitude and No. of stages used with Prefix result in a Braun Array Multiplier

S.No	Data A (8Bits)	Data B (8Bits)	Prefix	Result	Stages Used
			(16 Bits)		
1	0000 0000	xxxx xxxx	0000 0000 0000 0000		0
2	0000 0001	0000 0001	0000 0000	1	1
			0000 000		
3	0000 001x	0000 01xx	0000 0000 000	1 xxxx	4
4	0000 1xxx	0000 1xxx	0000 0000	1xxx xxxx	7
5	1xxx xxxx	1xxx xxxx		1xxx xxxx xxxx xxxx	8

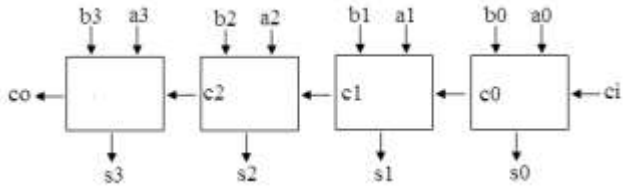

Fig 2. 4 bits Ripple carry adder

Table 2. Data Magnitude and No. of stages used with Prefix result in a Ripple carry adder

S.No	Data A (4Bits)	Data B (4Bits)	Prefix	Result	Stages Used
			(5 Bits)		
1	0000	0000	0 0000		0
2	0000	0001	0000	1	1
3	0001	0001	000	10	2
4	001x	01xx	0	1xxx	3
5	1xxx	1xxx		1 xxxx	4

Similarly, in a ripple carry adder, the addition of non-zero MSB positions on both the operation plus one result in the number of output bits. Fig 2 and table 2 depicts this relation. Hence, any arithmetic circuit implemented as a combinational logic followed by a latch or register can be operated with data dependent speed.

3. NIOS BASED ASYNCHRONOUS PIPELINED SYSTEM

NIOS II is a virtual microcontroller which can be brought out from an Altera Brand FPGA. NIOS II is fully software configurable with necessary on chip and off chip memory, interrupts, interrupt priority assignments and the required number of parallel input and output pins. For this application,

the microcontroller needs to do minimum work and hence, the design was done with minimum features to consume minimum area. Also, an attempt was made to simultaneously use NIOS II processor for some other functions but that worsens the control activity [6]. The asynchronous pipelined system with NIOS II is designed by replacing pipelining register with multiplexer based latches. The circuit diagram of a one-bit multiplexer based latch is shown in Fig. 3.a. The advantage of using multiplexer based latch is it consumes comparably lower power. The microcontroller runs a 'C' code and delays are assigned using 'C' code. But these delays are in the order of microseconds and suffer from lack of accuracy. Hence, separate control logic circuits (series of delay elements) are used. The system works as described below: The processor receives the data first and verifies it's a different data from the previous one. It opens the first Multiplexer based latch to allow the data in to the stage-1 of the process. At the same time a data '1' is passed to the control logic. The control logic-1 is designed in such a way that, it has an equal delay with the stage 1. Hence, the transmitted 1 has to come through all the delay and by the time it comes out of control logic, the process in the stage 1 must be over. Now, this 1 is used to trigger the interrupt1, which opens the next multiplexer based latch to allow the data to the stage-2. Simultaneously another 1 will be passed through the control logic-2. This process continuous till the data ceases. The registers are replaced by Multiplexer based latches for lower power consumption. The control logic circuits use series of logic elements (using LCELL feature – LCELL gives one Logical element delay and is not be optimized out during compilation). The number of logic elements (LEs) is decided based on the logic depth of that particular stage. For this circuit to operate the accurate delay estimation of each stage is needed and this is accomplished from the timing report or from the logic depth. This information is obtained as shown in figures below. Manual fine tuning of delays will improve the performance with this systems.

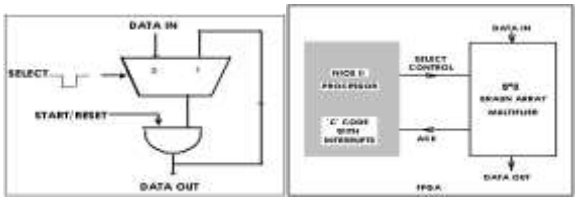

Fig. 3 a. One-bit version of multiplexer based Latch & Fig. 3 b. NIOS based Asynchronous Pipelined System

As an example, Fig. 4.a and 4.b are given from which the critical path propagation delay (t_{pd}) and logic depth are advised. It has a logic depth of 8 Logic elements. Hence, in the control logic there are around 12 LCELLs are in series which includes the routing delay between the logic elements. Equal number of LCELLs is not sufficient as the increase in fan out in the actual circuit introduces additional delay which should be considered in delay estimation.

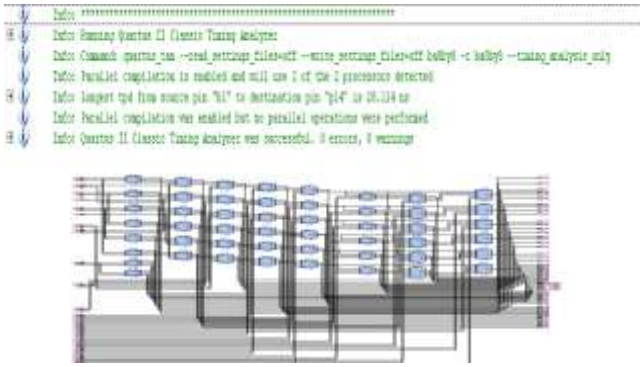

Fig. 4. a. & 4.b. Logic depth determination using t_{pd} timing report and using Technology map viewer

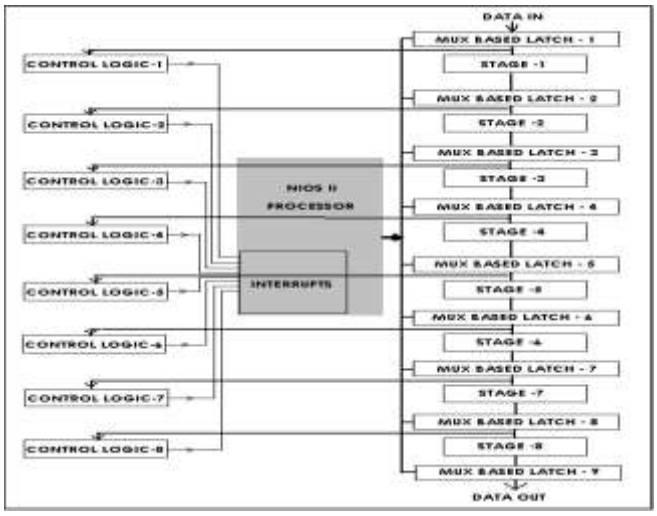

Fig.5. NIOS based 8 Stage Asynchronous Pipelined System (For conceptual reference, not dealt in this work)

But, with this implementation, NIOS II processor consumes large number of Logical elements and hence, the maximum speed of operation is limited to around 200 MHz. Also, the programming is done using high level language 'C' and interrupts are not efficient in real time control of the system. Importantly, in this implementation, the NIOS II processor also takes care of Built in Self test feature [7]. For a set of test vectors, the results are stored in the NIOS II memory and this is also used to validate the results. The complete scheme is presented in fig. 5.

4. INTASYCON BASED ASYNCHRONOUS PIPELINED SYSTEM

The asynchronous controller named as "INTASYCON" (stands for INTelligent ASYnchronous CONtroller) [8] is an HDL based hardware module used to control the asynchronous data flow inside an asynchronous circuit. The data manipulations inside the controller are described below:

1. The controller has an in-built free running counter as shown in Fig 6.
2. The source and destination of data are assumed to memories. After the first start signal, the controller itself fetches the new data from the memory. The controller receives the data and verifies it is not a repeating data. If it is a new data, the controller fetches the current value in the counter.
3. The magnitude of the data is analyzed and for every stage, the delay counts are selected based on the input magnitude.
4. The delay values are added with the initial count value so that the completion time of all the stages can be estimated.
5. A critical case happens when the previous data consumes more time and the current data consumes less time. In that case, the current data has to wait in the n^{th} stage till the previous data gets manipulated in the $n+1^{th}$ stage. In conventional asynchronous circuits, this is taken care by the Muller-C element and acknowledgement signal. Here, since the controller dictates the end point of stages it compares the current data's end point (of n^{th} stage) and previous data's end point (of $n+1^{th}$ stage) and which is greater is chosen as the current data's endpoint (of n^{th} stage).
6. The process continues till the data ceases.

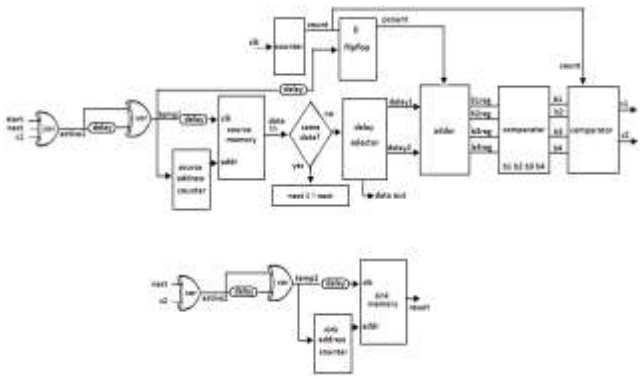

Fig. 6 INTASYCON with source and sink memory

4.1 CIRCUIT DESCRIPTION

1. The process is started by a leading edge transition of 'start'. Since 'next' and 'c1' are at 0 states, positive edge in 'start' will drive 'active' to toggle its state.
2. 'Active1' is dual edge sensitive, but conventional memories are positive edge triggered, hence a delay with xor gate converts dual edge sensitive into positive edge sensitive as in fig. 7.a & 7.b. 'Temp1' is positive edge sensitive signal and given as the clock after delay to source memory where input data are stored. The same is connected to an address counter to generate addresses.

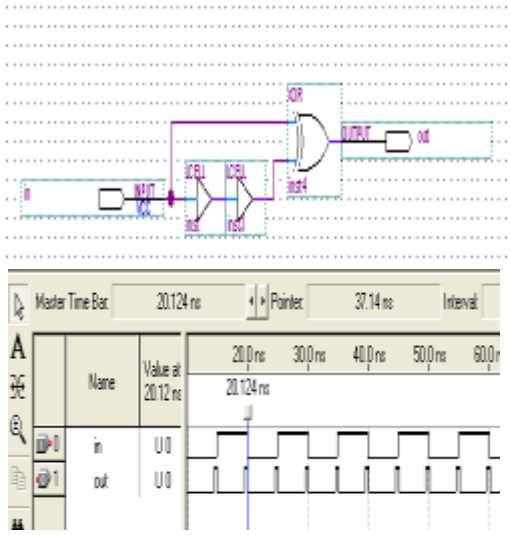

Fig. 7.a & 7.b Dual Edge to Positive Edge conversion (Used for using Conventional Memory with dual rail Asynchronous process)

3. The data in releases the data which is compared with the previous data. If it is a repeating data, 'next' is toggled which is connected to xor gates as shown in the fig.6. and is used to trigger both memories. In the sink memory, the previous output is latched again and in source memory next data will be released because of this toggle in 'next'.
4. If the data is new, the delay selector, depending upon the magnitude of the data gives out delays for various stages. In this fig. 6. it is for two stages. And the data has given out at 'dataout'. The data processing need not wait for this control manipulations. The data will get simultaneously processed in the stage-1.
5. 'Temp1' after suitable delay is used to pick the current value from the free running counter. This 'pcount' is the counter value when the data enters into the process.
6. Since, there are two stages, two data will be available in them and therefore 4 delays are available at any time. Hence, signals b1reg (=pcount+delay1) and b2reg (=pcount+delay2) are completion of data1 and signals b3reg and b4reg are completion counts of data2.
7. Assume data2 is under process and first stage will over at b3reg. The second stage cannot be opened if previous data1 has not gone out of stage2. Hence, this b3reg is compared with b2 which gives second stage completion of data1.
8. Likewise, b1, b2, b3 and b4 are the obtained after comparison.
9. These values are compared with present counter value and when counter reaches the end point calculated, c1 and c2 toggles.
10. c1 indicates the completion of stage1 and hence next data can be taken from memory. So, c1 is also connected to input xor gate.
11. c2 is used to trigger the output memory so that the output from stage2 can be properly latched.

12. c1 and c2 are also dual edge sensitive signals and hence, dual edge triggered flipflops are used to transmit data between stages.

5. BIOMEDICAL APPLICATION OF INTASYCON BASED SYSTEM

As a case study, filters for telemedicine application have been chosen. A mobile telemedicine is even more stringent where a patient is moving where the power supply facilities may not be available. In such circumstances, the acquisition and processing of physiological signals should consume minimum power and the devices should be portable. The physiological signals directly acquired from a patient gets corrupted or interfered with many noise sources like power line interference, base line wander noise, motion artifact noise, muscle contraction, Instrumentation noise generated by electronic devices etc. Various types of filters are needed for physiological signal processing for various purposes. For example linear phase filters are used to preserve the QRS complex in an ECG waveform and Smoothing filters [9] to reduce high-frequency noise. Some sources of high-frequency noise include 60-Hz, movement artifacts, and quantization error. One simple method of reducing high frequency noise is to simply average several data points together. Such a filter is referred to as a moving average filter or a Hanning filter, Notch filter and Derivative filters.

These are mainly used to smoothing the data to reduce the high frequency noise. In ECG the main sources of high frequency noises include 60Hz noise, EMG noise, movement of artifacts and quantization error. This smoothing filter uses a simple concept of reducing high frequency noise to simply average several data points together [9]. The hanning filter computes a weighted moving average, since the central data point has twice the weight of the other two [12]. The difference equation can be given by

$$y(nT) = ¼[x(nT)+2x(nT-T)+x(nT-2T)] \quad (1)$$

x(nT) and y(nT) are input and output sequences for the current sample time and x(nT-T) is the past input sample point and its z-domain input is $X(z)z^{-1}$.

$$Y(z) = ¼[x(z)+2x(z)\, z^{-1}+x(z)\, z^{-2}] \quad (2)$$

The following Fig.8.a shows the Hanning filter.

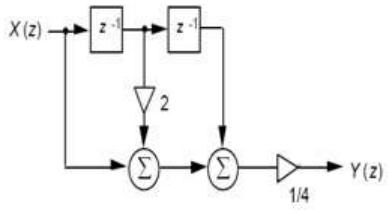

Fig. 8 Structure of Hanning Filter (Smoothing Filter)

6. INTASYCON BASED HANNING FILTER

Fig.9 shown is INTASYCON based Hanning filter in which two adders are coded with data dependent delay. Multiplication by 2 and divided by 4 operations are performed using shifting operations, which also can be coded with Data dependent delay. After preprocessing of data, the data is exposed to the first stage (Ripple carry adder). Based on the data magnitude, C1 and C2 are issued. C1 and C2 operate DETFFs to pass the data between stages. The completion times of all the stages for a particular data are known and hence, the controller verifies whether the previous data has come out from a particular stage before the next data is presented.

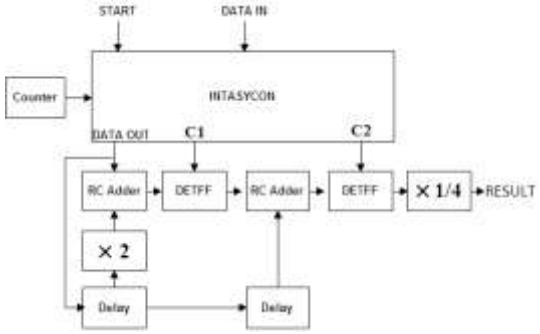

Fig. 9. INTASYCON Based Hanning Filter

7. ISSUES IN THE DESIGN

The delays specified in the design are calculated based on the logic depth of the combinational logic and from the path delays reported by the timing analyzer. But, increase in fan-in or fan-out also changes the delay required. The fine tuning of delays can be done by the self-tuning of this controller by using a set of test vectors and this work is in progress.

INTASYCON is designed as a general purpose controller to suit any application and this requires minimal tuning to be performed. Depending upon the application, the following parameters are configured.

1. Time period of the free running counter,
2. Counter range (number of bits) and
3. Delay count assignment for the specific application.

For example, the critical path of Braun array multiplier when implemented in Cyclone II FPGA is 28 ns. When a clock of 4 ns time period is used, it requires 7 counts to cover the worst case delay. Hence, the counter used has 3 bits (8 states). If the number of bits in the counter and counting frequency is high, number of transitions increases and hence dynamic power consumption also increases. But, with reduced time base a tight time control is not achieved. For example, for an application with 28 ns critical, if 5 ns time base is used, a delay of 30 ns is possible. Hence, for every data is idle for 2 ns. Hence, optimum values are selected according to the critical path of the application.

7.1 Significance of tuning parameters

The following factors influence the selection of tunable parameters.

1. Time period of the free running counter

The time period of the free running counter is selected based on the critical path of the combinational logic. If the delay values of

a three stage system are 15 ns, 20 ns and 25 ns, then clock period can be chosen as 5 ns. If higher clock is selected, unnecessarily, the number of transitions increases for the same delay.

2. Counter range (number of bits)

Based on the critical path and time period of clock, the counter range is selected. If the number of bits in the counter increases, unnecessarily the number of transitions increases.

3. Delay count assignment for the specific application

Based on the critical path and clock frequency, the delay counts are assigned. For the sample data given above, the delay counts will be 3, 4 and 5 respectively.

8. RESULTS AND CONCLUSION

Asynchronous circuits have many potential advantages over their synchronous equivalents including lower latency, lower power consumption, less noise, design reuse and lower electromagnetic interference (EMI). The new approach of synchronizing the data between multiplexer based latches using NIOS II processor and between DETFF using INTASYCON has been verified. In the INTASYCON system, the addition of controller a) increases the area because of controller architecture but compensates by removing the pipelining registers, b) reduces speed when compared to a pipelined architecture but compensates by providing data dependent delay.

The control signals generated by these processors are avoiding a global synchronizing signal used in synchronous systems. The data dependent delay assures the delay to be varied according to the data and hence, the throughput of the system is further increased.

Table 2. Comparison of configurations of Hanning Filters

Configuration	LEs used	Speed
Without Pipeline (Combinational)	242	(Critical path delay 15.876ns) 62.9 MHz
With Pipeline-Synchronous	540	252.46 MHz
With Pipeline – Asynchronous & NIOS	2708	101.10 MHz
With Pipeline – Asynchronous & INTASYCON	721	Data dependent throughput

The intelligence is introduced by having a soft core processor via NIOS, but since, the operating frequency is very low (101.10MHz) and consumes 2708 LEs as shown in Table 2. INTASYCON controller, with data dependent delay feature, is implemented with the same functionality using HDL. This enables to achieve a higher speed while maintaining low LE count of 721. The overhead is comparably higher for a Hanning filter but, the same INTASYCON controller can be used for even higher order circuits with minimum modification. Hence, as the complexity of the application increases, the overhead due to INTASYCON is acceptable. The control signals generated by these processors are avoiding a global synchronizing signal used in synchronous systems. The data dependent delay assures the delay to be varied according to the data and hence, the throughput of the system is further increased. When the data is all of minimum magnitude, the critical path will be 7 ns hence the throughput will be doubled. When the magnitudes of all data are large, then the critical path equals 15.876 ns as shown in combinational case. Hence, the throughput depends on the nature of data input.

For normal implementations, as the application complexity increases, the delay also increases which occupies more area in the hardware. However, using this scheme, the area remains almost constant irrespective of the application. In fact, as the complexity and critical path delay increases, the efficiency of the INTASYCON scheme also increases.

9. REFERENCES

[1] K.K. Parhi, "VLSI Digital Signal Processing Systems", Wiley Interscience.

[2] Jens Sparsø, Steve Furber, "Principles of asynchronous circuit design", Kluwer academic Publishers, 2001.

[3] S.M.Nowick, "Design of a low-latency asynchronous adder using speculative completion", IEE Proc. –Comput. Digit. Tech., Vol. 143, No. 5, September, 1996.

[4] S.M.Nowick, Kenneth Y. Yun, Peter A.Beerel, Ayoob E.Dooply, "Speculative Completion for the Design of High-Performance Asynchronous Dynamic Adders" pp 210-223,1997.

[5] G. Lakshminarayanan, T.N.Prabakar, "Design & Implementation of Asynchronous Braun Array Multiplier", International Conference on Advanced Computing and communication, pp 61-66, 2007.

[6] T.N.Prabakar, G. Lakshminarayanan, Dr. K.K.Anilkumar, "SOPC based asynchronous pipelined DCT with self test capability", IEEE International Conference on Microelectronics, Egypt, 2007.

[7] G. Lakshminarayanan, T.N.Prabakar, "On board Testing of Digital Systems using Nios Processor", IEEE International Conference on Signal Conditioning & Networking, 2007.

[8] T.N.Prabakar, G. Lakshminarayanan, Dr. K.K.Anilkumar, "Asynchronous Pipelined Multiplier with Intelligent Delay Controller", IEEE International Conference on System on Chip Design, Korea, 2008.

[9] "Biomedical Digital Signal Processing" by Willis J. Tompkins University of Wisconsin-Madison book was printed by Prentice Hall, Upper Saddle River, New Jersey 07458.