International Research Journal in Advanced Engineering and Technology (IRJAET)

ISSN (Print) : 2454-4744 | ISSN (Online) : 2454-4752 (www.irjaet.com)

Vol. 1, Issue 2, pp.36-42, July, 2015

RESEARCH ARTICLE

DESIGN AND IMPLEMENTATION OF FPGA BASED WAVE-PIPELINING FOR DIGITAL SIGNAL PROCESSING CIRCUITS

D. Sobya

Department of Electronics and Communication Engineering, Research scholar: Sathyabama University
e-mail: sobyadevaraj@gmail.com

ARTICLE INFO

Article History:

Received 19th,June, 2015
Received in revised form 28th, June, 2015
Accepted 1st July, 2015
Published online 3rd July, 2015

Key words:

FPGA
Wave-Pipelining
Multiplier
Shift Register, DSP Block

ABSTRACT

This paper reveals that the techniques for efficient implementation of Field-Programmable Gate-Array (FPGA)-based Wave-Pipelined (WP) multipliers, accumulators, and filters is presented. A comparison of the performance of WP and pipelined systems has been made. Major contributions of this paper are development of an on-chip clock generation scheme which permits finer tuning of the frequency, a synthesis technique that reduces the area and latency by 25%, a placement utility that results in 10%–40% increase in speed and proposal of an interleaving scheme for filters that reduces the number of multipliers required by 50%. WP multipliers of size 2 6 and the filters using them are found to be 11% faster and require lower power than those using pipelined multipliers. Filters with higher order WP multipliers also operate with lower power at the cost of speed. The delay-register products of such filters are found to be about 60% lower than those using the pipelined multipliers. The project also outlines applications of these techniques for the Spartan II FPGAs and a self-tuning scheme for optimizing the speed.

I. INTRODUCTION

Field Programmable Gate Array based system design is gaining extensive popularity due to the flexibility and complexity it provides. FPGAs with complexities, as high as 10 million gates in a single Integrated Circuit (IC) have become a reality. This has enabled the FPGA vendors to embed the Restricted Instruction Set Computer (RISC) processor in part of the core so that in a single IC the advantages of both microprocessors and FPGAs can be combined, leading to the design of a complete System on a Single Chip (SOC). In view of this, the study of FPGA-based implementation of various systems that have traditionally been implemented either using Application-Specific Integrated Circuits (ASICs) or programmable digital signal processors become important. Design and FPGA-based implementation of digital signal processing blocks using both pipelining and wave-pipelining techniques are considered in this project. They are required to operate the WP circuits at high speeds. Here in one section describes a PC-based testing scheme. This is used for testing all the WP circuits, in this article. The case studies were carried out

on convolvers and multipliers to bring out the better amenability of array multipliers for wave pipelining.

II. LITERATURE REVIEW

Wave-pipelining is projected as one of the method for achieving high speed without the cost of increased area and circuit complexity. The basic criterion used for partitioning the execution path is hard to achieve in practice because of the differing amounts of logic per stage and variations in time delays per logic element [1&2]. A new critical path approach to speeding up wave pipelining technique for Distributed Arithmetic Algorithm (DAA) based Finite Impulse Response (FIR) filter using a control circuit has been presented by Charanjit Singh [3]. Terrence Mak et al [4] proposing a novel wave pipelined signaling scheme to achieve substantial throughput improvement in FPGAs. A new analytical model capturing the electrical characteristics in FPGA interconnect is presented. For Xilinx FPGAs, the physical design editor referred to as FPGA editor may be used for measuring and altering the delays. Using this feature, the implementation of wave-pipelined circuits on Xilinx FPGAs is considered in [5]. The availability of on-chip dedicated multipliers, soft core/hard core processors and IP cores make the FPGAs to be an ideal platform for the implementation of area as well as speed intensive image processing applications such as Discrete Cosine Transform (DCT) and DWT [6].

The micro architecture of the 32 bit sparse tree adder [7] has two main features: asynchronous hybrid wave-pipelined processing and a prefix sparse-tree carry generate-propagate structure for arithmetic. V.KrishnaKumari proposed method [8] modification is done by replacing the parameter 4-bit carry skip adder with 4- bit carry look ahead adder, 4-bit Kogge-Stone adders. The BIST approach requires a number of overheads such as Finite-State Machine (FSM), signature generator and test vector RAM [9&10]. Instead of using a dedicated circuit such as Built-In Self-Test (BIST), a processor may be used to carry out the tuning and retuning tasks [11]. Despite of the irregularity and idiosyncratic nature of FPGA long interconnections, buffers were embedded at switches to speed up the signal propagation [12]. Recently the focus of wave-pipelining had shifted from the logic to the interconnection circuits and a number of interconnect wave-pipelining design for ASIC has been proposed [13-16] in order to achieve a higher throughput of interconnections.

III. METHODOLOGY

The idea of wave-pipelining or maximal rate pipelining was first formalized in shift register. Recently, this concept has been a subject of renewed interest as technology and design techniques have enabled the effective implementation of wave-pipelining in integrated circuits. The concept of wave-pipelining has been described in a number of previous works. To illustrate this concept, graphical representation of the data flow through combinational logic is used. Fig. 1(a) and (b) shows the conventional single-stage system and its associated timing diagram. The combinational logic is surrounded by edge-triggered input and output registers. At the beginning of each clock cycle, data is initiated into the logic block at the input register. Due to the differences in the circuit path lengths and other factors, data delay through the combinational logic will vary. In Fig. 1(b), the shaded regions bounded by the maximum and minimum delays through the logic Dmax and Dmin depict the flow of data through the combinational logic and the variations in the logic block with time. The non shaded areas depict the stable duration of the logic. In the conventional system, the output register is clocked in the non shaded region, and the minimum clock period is chosen to be greater than Dmax. In the WP system, the clock period is chosen to be (Dmax-Dmin) +clocking overheads such as setup time, hold time, etc. To ensure correct operation, the clock to the output register should be delayed so that the active clock edge occurs in the stable period. Moreover maximize the frequency of operation of the WP system, the difference (Dmax-Dmin) is minimized by equalizing the path delays. As the shaded region increases with an increase in the logic depth, the operating clock frequency should be reduced to ensure correct operation. An alternative technique to avoid decreasing the clock frequency is pipelining. However, the need for additional registers increases the area, power, latency, and clock routing complexity. On the other hand, variation of and due to various factors such as difference in rise and fall times, variations due to process, environment, and voltage changes make the delay equalization a challenging task in WP systems. Several methods need to be adopted to achieve the equalization of path delays. Algorithms to automatically equalize the delays in combinational logic circuit are reported. In a WP multiplier is implemented on Normal Process Complementary Pass Transistor Logic (NPCPL), and an algorithm is adopted to bring the shortest path delay equal to the longest path delay. ASIC based WP systems have been successfully implemented for a variety of applications.

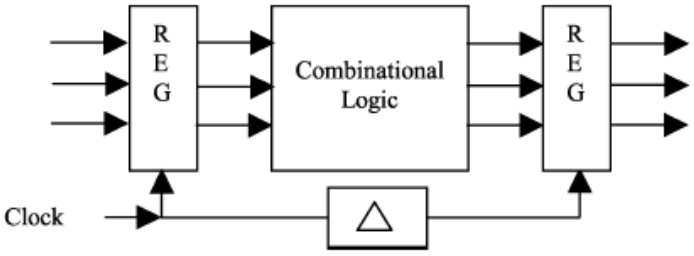


Fig. 1(a) Date Flow through Combinational Logic Circuit

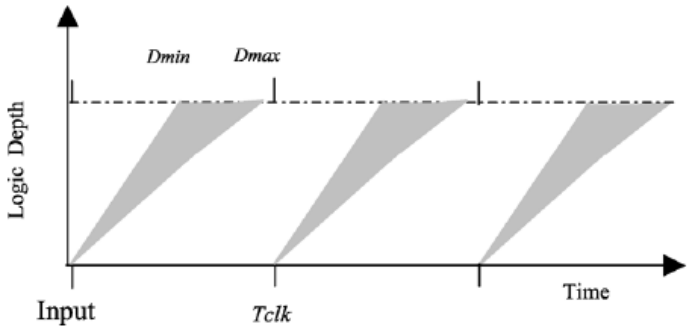


Fig. 1(b) Temporal/Spatial Diagram of Combinational Logic Circuit

In the feasibility of wave-pipelining using lookup table (LUT)-based FPGAs is studied through the implementation of the Guild multiplier Using a two phase clocking scheme, the WP circuit is found to be operating ten times faster than the speed predicted by the timing analyzer tool (i.e., based on alone). External clocks are used for the input and output registers, surrounding the multiplier. The skew for the clock to the output register is manually adjusted to lie in the stable period, by observing the output of the multiplier in Cathode Ray Oscilloscope (CRO). The use of external clocks in limits the WP circuit to be operated at a lower frequency than what would be possible if the clock had been generated within the FPGA. The frequency of the externally fed clock signal is restricted by the printed circuit board in which the FPGA is mounted and the input–output (I/O) pad delays.

A. WP Clock

In medium- and high-density FPGAs, the interconnect delays become comparable to those of the active device blocks in Configurable Logic Blocks (CLB) such as Look Up Tables (LUT) and flip-flops. For example, for the XC4003E-1 device, the LUT delay is typically 1.3 ns, and the interconnect delay between LUTs can be varied over a range of 0.7–7 ns. A clock signal can be generated by interconnecting the output of a number of LUTs in cascade to the input of the first LUT in the chain. The highest frequency is obtained when a single LUT is used as shown in Fig. 2. Low values of interconnect delays such as 0.7 ns cannot be achieved if a particular LUT has high fan-out. High fan-out is desirable since the clock may have to be applied to a number of blocks. On implementation of the clock in an FPGA, it is observed that an interconnect delay of 1.4 ns can be achieved with the least number of interconnect resources and reasonably large fan outs. With this delay, a clock period of 5.4 ns is obtained. The clock generated has a frequency very much above what can be fed through the I/O pads using the demo board. An alternative scheme for generation of high frequency clock is through multiplication of an external clock using Delay Locked Loops (DLLs). This feature is not available in XC4000 and Spartan family of FPGAs, whereas Spartan II and Virtex FPGAs contain DLLs. In these devices, however, the maximum multiplication factor can be only 16. However, the clock generation scheme described above has the advantage of changing the clock periods in smaller steps. Moreover, the clock frequency can also be altered through programming by controlling the number of LUTs in the forward path of the clock generator. To study the drift in the clock frequency, the clock signal is divided by a large number and observed in the CRO. The clock waveform was found to be stable. Moreover, the Static Timing analysis (STA) input of the clock generator enables the clock signal to be reset periodically to minimize the drift, if any.

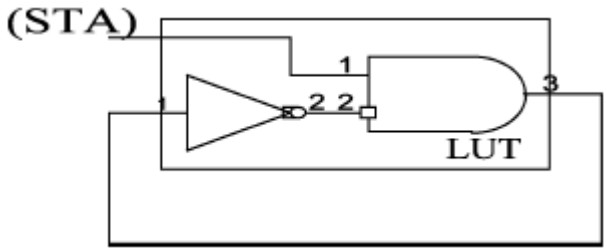


Fig. 2. Clock Circuit

B. Implementation of the WP Counters

The basic clock signal may be used as the least significant address input for the RAM in the WP system. Counters may be used to generate the higher order addresses for the RAM. In XC4003E-1 device, the minimum write cycle time for the flip-flop in the CLBs is 6 ns. Hence, for clock signals of period 6 ns or more, the counter can be implemented using the flip-flops in the CLBs. For clock periods less than 6 ns, the counter may be realized using the LUTs in the CLBs. Lookup table with feedback between output and input functions as a latch. Using these latches, a novel WP counter is proposed in this paper. A 2-bit WP counter is shown in Fig. 3. In this counter, the interconnect delay between the output of the first LUT and the input to the second LUT is made equal to the interconnect delay at the feedback paths of both of the LUTs. The AND and XOR functions, indicated in Fig. 3, are implemented using a single LUT and the STA input is used for starting the counter. The above technique can be extended for the design of higher order WP counters.

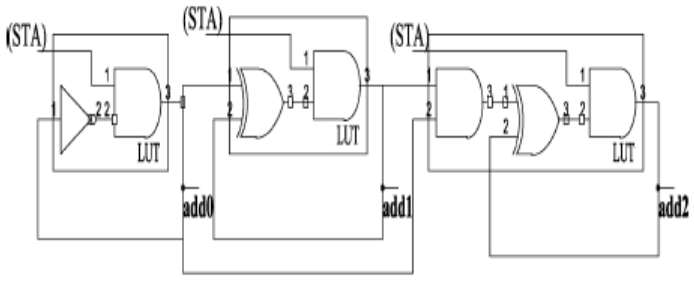
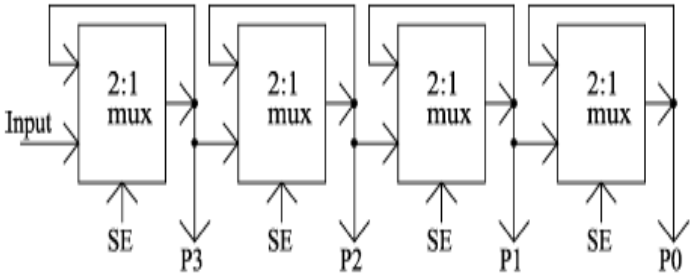


Fig. 3 WP Counter

C. WP Shift Registers

The circuit diagram of a 4- bit WP shift register is shown in Fig. 4. The multiplexers in Fig. 4 operate in two modes. When shift enable is 1, the data output from each multiplexer is shifted into the multiplexer on the right. When shift enable is zero, the data output of each of the multiplexers at the 1–0 transition is latched onto the same multiplexer. If the interconnect delay between the multiplexers is adjusted to be 1.4 ns and if the LUT delay is assumed to be 1.3 ns, then the WP shift register is equivalent to a conventional shift register with shift clock period of 2.7 ns.



SE – Shift Enable

Fig. 4. 4-Bit WP Shift Register

IV. PC-BASED TESTING SCHEME FOR WP CIRCUITS

The operation of simple circuits such as a clock can be verified using CRO after suitable frequency division. For testing more complex circuits, a PC based testing scheme is developed using the General Purpose I/O (GPIO) PC add-on card. The GPIO card is assumed to be used for writing the test data into the input RAM and reading the output from the output register. In order to ensure that the speed of the I/O card does not restrict the maximum operating frequency of the WP circuit, it is required to carry out the read/write (r/w) operation by the add-on card at a rate different from the rate at which the WP circuit processes the data. This is achieved by connecting the address and data bus of the input RAM as shown in Fig. 5. The address multiplexer is implemented in the same FPGA in which the WP circuit is implemented. The start and r/w signals are controlled by the PC add-on card. After the design is downloaded into the FPGA, a testing routine executed from the PC applies the test inputs, collects the results, compares them with the expected results, and reports the discrepancies if any.

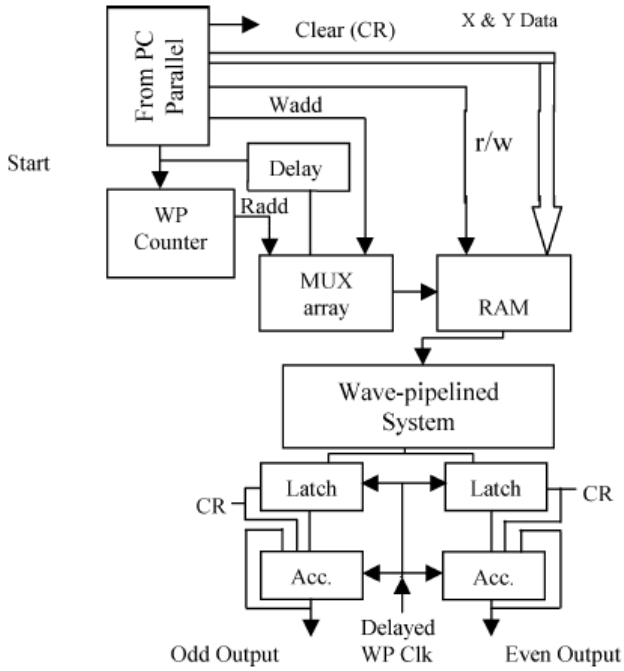


Fig. 5 PC- Based Testing Skill for WP Circuits

V. DESIGN AND IMPLEMENTATION OF FPGA-BASED WP DSP CIRCUITS

AWP circuit may be implemented using the layout editor by manually choosing the LUTs required, specifying the function to be performed by each LUT, the inputs and outputs to be interconnected, and the composition of the interconnect for each interconnect. Alternatively, to specify the design, the Hardware Description Language (HDL) entry may be used and the exact location of the CLBs to be used for realizing the different child modules in the HDL program may be specified through the user constraints file. Moreover, after the placement and routing step by the Computer-Aided Design (CAD) tool, the delays have to be examined using the layout editor and altered, if required, to equalize the delays. Using the second approach and the design techniques given in Section IV, WP multipliers multiply accumulators, and serial/parallel convolvers are designed and implemented. Some details of the design and implementation results are presented in this section.

A. WP Serial / Parallel Convolver

As a case study for investigating the feasibility of operating the WP circuits faster than the pipelined circuits, a one-dimensional (1-D) WP 8-tap convolver using 8-bit serial/parallel WP multipliers is implemented on Xilinx XC4006E. In the serial/parallel convolver, the product bits arrive serially, and accumulators are implemented using WP shift registers. The categorical matching proposed in [21] is used to equalize the path delays. To adjust the clock skew of the output RAM/register, for a known sequence of test inputs, the

simulated output of the convolver is read and compared with the expected result. The clock skew is adjusted until the correct results are obtained. The results obtained through simulation are compared for convolvers with and without wave-pipelining. The WP convolver is found to be faster by a factor of two compared with the convolver using pipelining. This is because, for the pipelined circuit, the minimum clock period is 6 nos due to the limit posed by the flip-flops. The WP convolver requires 73% more CLBs. To confirm whether the simulation results match with the actual results obtained from the device, the design is downloaded to the FPGA and tested using the PC based testing scheme described here. From the test results, it is found that the convolver does not work correctly. This has indicated the inadequacy of the simulation for testing the WP circuits. This may be caused by the significant difference between the routing delays reported by the layout editor and the actual delays.

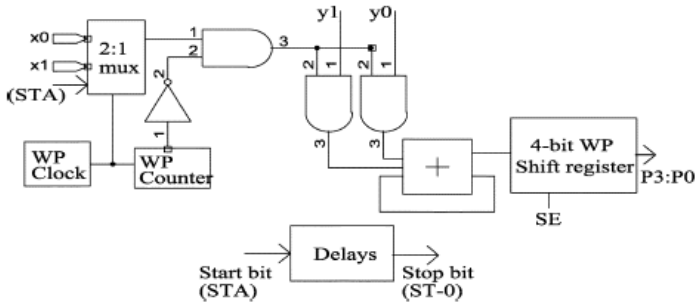


Fig. 6. WP 2x2 Serial Parallel Multiplier

VI. OPTIMIZATION SCHEMES FOR FPGA BASED PIPELINED AND WP MULTIPLIERS

A. Optimally Synthesized Pipelined Array Multiplier

The objective of the synthesis technique proposed in this section is to ensure that all of the four inputs of the LUTs in FPGAs are effectively engaged. Optimization on the use of four input LUTs is considered, as FPGAs from popular FPGA design houses such as Xilinx and Altera contain four input LUTs. Let us consider the optimization of a 4x4 array multiplier given in Figure 7 and the stages involving half-adders utilize the LUTs inefficiently. These stages may be modified to engage all of the four inputs of the LUT, as follows: Stage1 may be modified to compute the partial products due to the two least significant multiplier bits. The last stages may be reduced to stages by replacing the half-adders with suitable functional blocks and feeding the sum and carry outputs from one stage to another properly. The resulting pipelined array multiplier is referred to as Optimally Synthesized Pipelined Array Multiplier (OSPAM) and is shown in fig. It consists of five stages of combinational logic blocks. These stages use the functional blocks M0, M2, M5, M6, M7, M8, M9, and M10. The inputs and outputs of each of these blocks are given in Fig. Using these, logic equations for the various blocks can be written. The latency of the multiplier is reduced from eight clock cycles to five clock cycles. In general, multiplication can be achieved using registers with the latency of clock cycles. This scheme results in 25% lower latency and requires 25% lower area for implementation. This scheme is applicable for both pipelined and WP array multipliers.

B. Observations and Results

The pipelined array multiplier implemented using the optimization schemes proposed in Sections VII-A and VII-F is denoted as OPARAM. The conventional pipelined array multiplier, the Guild multiplier, OSPAM, and OPARAM are implemented on the XC4010E-1 device. The different characteristics of 4x4 multipliers such as number of CLBs required fmax maximum operating frequency and latency in nanoseconds are evaluated for all the above multipliers and are tabulated in Table .The WPARAM of different sizes are implemented and tested using both simulator- and PC-based testing schemes. WPARAM of size 4x 4 and 2x 6 are found to be satisfactory at a clock rate of 185 MHz. However, WPARAM of size 4 6 and 6x 6 are found to be satisfactory only at a lower frequency. From these results, it may be noted that for a given operating frequency, the maximum size of a multiplier that can be designed using the wave-pipelining technique is limited. However, OPARAM of larger sizes can be directly implemented and operated with the clock period of 6 ns. WPARAM of larger sizes have to be operated with a lower clock frequency. For higher order multipliers, OPARAM is to be preferred, if higher speeds are required and any clock frequency lower than 166 MHz can be used. If power dissipation is the primary concern, one may go for WPARAM instead of OPARAM. The effectiveness of the structure organizer utility is studied by implementing different types of 8x8 multipliers on a Spartan XCS30-3VQ100 device (a Xilinx Spartan family device with speed grade 3), and the results are given in Table. From Table, it may be noted that the structure organizer improves the speed of the multipliers by a factor of 1.1 to 1.4 times compared with that achieved by using commercially available P&R tools. For larger size multipliers and in designs engaging a majority of CLBs in an FPGA, the routing complexity is bound to decrease the speed of the multipliers further when they are designed with commercially available P&R tools. However, the structure organizer makes the design operate at maximum rate, irrespective of the routing complexity.

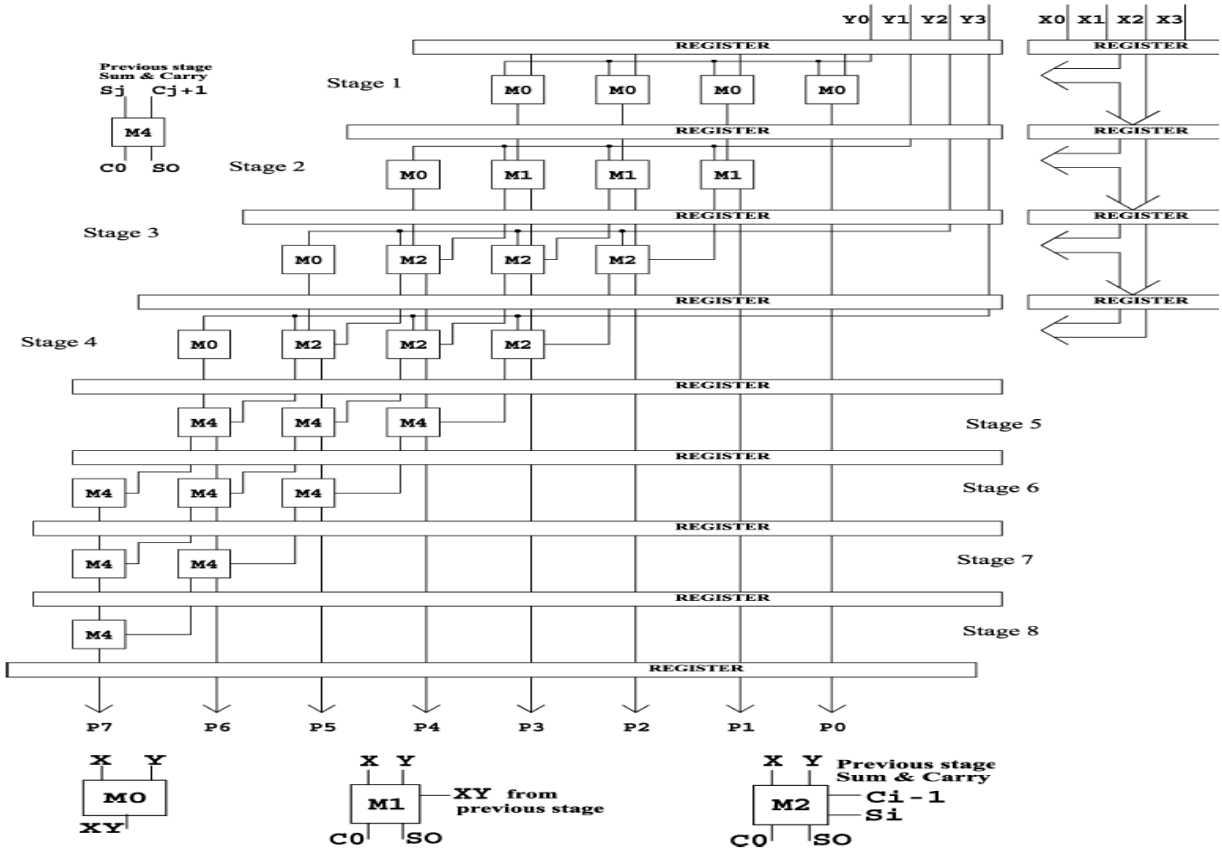


Fig. 7. Optimally Synthesized Pipelined 4x4 Array Multiplier

TABLE I. IMPLEMENTATION RESULTS OF PIPELINED AND WAVE-PIPELINED ARRAY MULTIPLIERS

Name of the Multiplier	Area in CLBs	Fmax (MHz)	Latency (ns)
Pipelined Array Multiplier– 4x4	48	128	70.3
Guild Multiplier- 4x4	44	117	68.4
OSPAM- 4X4	32	131	45.8
OPARAM- 4X4	32	166	36.0
WPARAM- 2X4	15	370	10.8
WPARAM- 2X6	30	185	27.0
WPARAM- 4X4	32	185	32.4

VII. CONCLUSION

The FPGA based WP multipliers have established that they are superior to the pipelined multipliers in both speed and power for small operand sizes. Higher order multipliers also dissipate lesser power but operate at lower speeds. The synthesis technique proposed for both WP and pipelined array multipliers reduces the area and latency by 25%. Optimization schemes, proposed in this article is to enable the increase in the speed of the conventional pipelined multipliers by 10% to 40% and have led to a reduction in the area by 40% and increase in speed by 58% for the filters using the interleaving. Application of these techniques for WP systems has enabled partial automation of the design technique and the filters using WP multipliers result in 60% reduction in the delay–register product compared with those using in pipelined multipliers.

REFERENCES

[1] Hirak Kumar Maity, M B Sarkar and A. Chakrobarty, "Wave Pipelining: An Analysis for High Performance Digital Circuits", International Journal of Electronic Engineering Research, Vol. 1(3), (2009). pp. 269-278

[2] Suryanarayana B. Tatapudi and José G. Delgado-Frias, "A High Performance Hybrid Wave-Pipelined Multiplier", VLSI, Proceedings. IEEE Computer Society Annual Symposium, (2005), pp. 282- 283

[3] Charanjit Singh, Balwinder Singh, "Design of High Performance Modified Wave pipelined DAA Filter with Critical Path Approach", International Journal of Electrical and Electronics Engg., Vol. I (2), (2011), pp. 28-32

[4] Terrence Mak et al, "Wave-pipelined intra-chip signaling for on-FPGA communications", INTEGRATION, the VLSI Journal, Vol. 43, (2010), pp. 188-201

[5] Lakshminarayanan and B. Venkataramani, "Optimization techniques for FPGA-based wave-pipelined DSP blocks," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol. 13(7), (2005), pp. 783-793

[6] A. Draper, J. R. Beveridge, A. P. W. Bohm, C. Ross, and M. Chawathe, "Accelerated image processing on FPGAs," IEEE Transactions on Image Processing, Vol. 12, No. 12, (2003), pp. 1543-1551

[7] P.Chaitanyakumari and R.Nagendra, "Design of 32 bit Parallel Prefix Adders", IOSR Journal of Electronics and Communication Engineering Volume 6, Issue 1 (2013), PP. 01-06

[8] V.KrishnaKumari and Y.SriChakrapani, "Designing and Characterization of koggestone, Sparse Kogge stone, Spanning tree and Brentkung Adders", Int. Journal of Modern Engg. Research, Vol. 3 (4), (2013) pp. 2266-2270

[9] Seetharaman, B. Venkataramani, and G. Lakshminarayanan, "Design and FPGA implementation of self tuned wave-pipelined filters", IETE Journal of Research, Vol. 52(4), (2006), pp. 281-286

[10] Nyathi and J. G. Delgado-Frias, "A hybrid wave pipelined network router," *IEEE Trans. Circuits Syst. I: Fundam. Theory Appl.*, Vol. 49, No. 12, (2002), pp. 1764-1772

[11] G. Seetharaman and B. Venkataramani, "SOC implementation of wave-pipelined circuits," in Proceedings of IEEE International Conference on Field-Programmable Technology in Japan, (2007), pp. 9-16

[12] T. Mak, C. D'Alessandro, P. Sedcole, P. Cheung, A. Yakovlev, W. Luk, "Global interconnections in FPGAs: modeling and performance analysis", in: Proceedings of ACM Int. Workshop on System Level Interconnect Prediction, (2008), pp. 51-58

[13] Joshi, G. Lopez, J. Davis, "Design and optimization of on-chip interconnects using wave-pipelined multiplexed routing", IEEE Trans. VLSI Systems Vol. 15 (9), (2007), pp. 990-1002

[14] V.V. Deodhar, J.A. Davis, "Optimization of throughput performance for low power VLSI interconnects", IEEE Trans. VLSI System, Vol. 13, (2005), pp. 308-318

[15] S.-J. Lee, K. Kim, H. Kim, N. Cho, H.-J. Yoo, "Adaptive network-on-chip with wave-front train serialization scheme", Symposium on VLSI Circuits Digest of Technical Papers, (2005), pp. 104-107.

[16] Hedayati, "The new era of programmable systems", *Xcell Journal*, No. 42, (2002), pp. 7-9