Analysis and Design of Parallel Prefix Circuits with Faulty Nodes

Hatem M. El-Boghdadi[1,2], Fazal Noor[1], and Mostafa Mahmoud[1]

[1]Islamic University of Madinah, Faculty of Computer& Information, Saudi Arabia
[2]Cairo University, Computer Engineering Department, Egypt

Abstract
Parallel prefix circuits are circuits that generate the prefix computation of a given input. The prefix computation is used extensively in hardware circuits. Prefix computation has its wide applications in cryptography, fast adders, etc. Any hardware circuit that have adders as one of its components could benefit from such computation. Prefix circuits proposed in literature differ in their performance, cost and size. Usually most circuits use operation nodes with fan-in/fan-out of 2. One classification of prefix circuits is dependent on the width of the circuit: circuits with width equal to the input, and circuits with width less than the input. In this paper, we first perform an analysis of two important classes of parallel prefix circuits. The first class performs well when the input size is of the same width as the circuit. The second class performs well when the width of the circuit is greater than the circuit width. We analyze the two classes in case of existence of faulty nodes. We estimate the time penalty and the number of idle nodes when a node in a certain location in the circuit goes faulty. Then based on the analysis, we propose new designs that can better handle faulty nodes. Finally, we simulate the circuits on FPGAs to assess their performance with faulty nodes.

Key words:
parallel prefix circuits, prefix computation, faulty elements.

1. Introduction

The prefix operation is one of the important operations due to its wide range of applications. Computing the prefix of a certain set of inputs could be done sequentially or in parallel. Parallel prefix computation has gained much interest because the speed it offers for computation as well as the different techniques available. For an associative operation *, the prefix of a set of inputs $x_1, x_2, \ldots, x_n$ is $y_1, y_2, \ldots, y_n$ where $y_i = x_1 * x_2 * \ldots * x_i$.

Prefix computation is very important in many domains. Applications such as encryption, computing biological sequences, and many types of adders especially fast adders are direct applications for prefix computation.

Two main tracks were followed for prefix computation, namely algorithmic computation and combinational circuits. The first track uses the known parallel models for the computation. Models such as parallel random access machine (PRAM) [1], reconfigurable mesh (R-Mesh) [3], hypercubes [2], etc. were used to solve the prefix problem.

The other track uses computational and combinational circuits to solve the prefix problem. The combinational circuit is a directed acyclic graph (DAG) that has *m* inputs and *m* outputs. This DAG is said to be of width *m*. The circuit has several operation nodes and at least one duplication node. The operation node has two inputs and one or two outputs and performs the operation * on the two inputs. The duplication node duplicates the input and does not perform any operation. The width of the circuit is the number of inputs it can accept at a time and the size of the circuit is the number of its operation nodes. The depth of the circuit is the number of levels in the circuit. Figure 1 shows the operation and duplication nodes.

Figure 2 shows an example of the prefix circuit, $L(9)$ [7]. The input is presented at the top of the circuit and after several time units (depth of circuit), the circuit generates the prefix output. Figure 3 shows another prefix circuit, $H(9)$ of waist 1 ($L_t - L_f = 1$) that can perform well if the input size is larger than the width of the circuit. In this case, the input is decomposed into a number of subsets of smaller size. Each subset is then presented to the circuit in consecutive time steps and the circuit generates the output in consecutive time steps as well.

Existing prefix circuits can be categorized under the following categories: circuits with the same width as the input size, circuits with width less than the input size, reconfigurable circuits that have flexible width. In this paper, we consider the second track and target the analysis

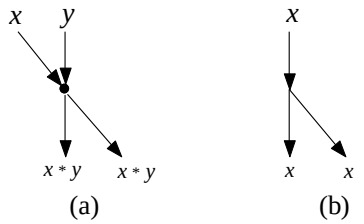

Fig. 1 (a) operation node (b) duplication node

Manuscript received December 5, 2019
Manuscript revised December 20, 2019

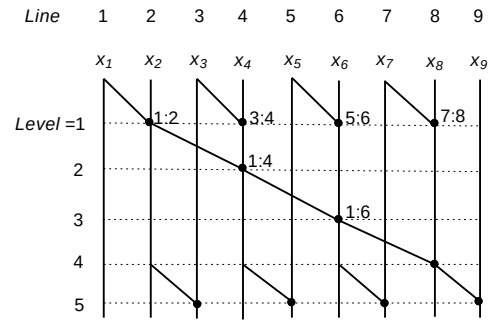

Fig. 2 Example of a prefix circuit, L(9) [7].

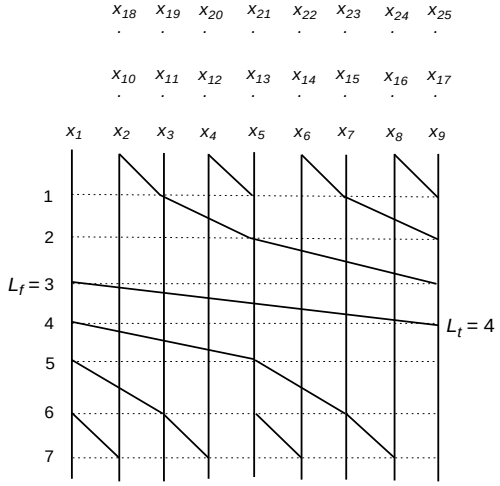

Fig. 3 Prefix circuit of waist 1 [4].

and design of the first two classes of circuits in the existence of faulty nodes. Many circuits were proposed in literature where they differ in performance, size, width and waist.

Previous proposed circuits assume all nodes are active. In this paper, we consider the existence of faulty nodes in prefix circuits and investigate their effect. We consider the circuits that performs well when the input has the same width as the circuit and circuits that perform well when the input has higher width than the circuit. In particular, for the first class we investigate the L-circuit [7] while for the second class we investigate the H-circuit [4]. For these two circuits, we first analyze them when there is a faulty node in one of the lines of the circuit. We show how this node could affect the operation of the circuit. We estimate the number of idles operation nodes. We also investigate the time penalty incurred when using the circuit to compute the prefix computation. We show that the L-circuits can handle the existence of faulty nodes better than the H-circuits.

Then, based on the analysis, we propose parallel prefix circuits that can handle the existence of faulty nodes. For the proposed designs, we estimate how many extra operation nodes needed to deal with a faulty node. Up to our knowledge, this is the first trial to consider analyzing the performance of prefix circuits that has faulty nodes and designing prefix circuits that can better handle faulty nodes. Finally, we simulate the prefix circuits on FPGAs to assess their performance with faulty nodes. We compute the time penalty incurred if the circuit has a faulty node.

The next section presents the analysis of the L-circuit [7] and the H-circuit [4]. Section 3 presents the new proposed circuits. Section 4 summarizes our results and makes some concluding remarks.

2. Prefix Circuits with Faulty Nodes

In this section we consider the analysis two classes of parallel prefix circuits. The first represents circuits that perform well when the input size is of the same width as the circuit width. The second class represents the circuits that perform well when the input size is larger than the input size.

2.1 Analysis of L-circuit with Faulty nodes

In this section we consider a parallel prefix circuit that is depth-size optimal, *L*-circuit. It was proved [7] that *L*-circuit has the smallest depth of other circuits that are depth-size optimal if the fan-out is 2. The *L*-circuit is shown to perform well when the input size is the same as the width of the circuit.

Here we analyze the *L*-circuit in the existence of faulty nodes and investigate the performance penalty of having a faulty node.

Let the $L(m)$ be an *L*- circuit of width m and let $d(L)$ be the depth of the circuit. If an input $x_1, x_2, \ldots, x_m$; of size m is presented to the circuit hen after d time steps the circuit generates the prefix sums of the inputs. Figure 2 shows an *L*-circuit of width 9, $L(9)$, of depth of 5 (The circuit has 9 lines and 5 levels.)

We first show that for $L(m)$, the circuit can be used for prefix computation of size n, $n < m$. Considering the *L*-circuit structure, we find that there are no links connecting two lines i and j, $i < j$, and also the node level in line $j <$ node level of line I, Thus, any line j does not affect the output of line i, $i < j$. The output of line i is only affected by inputs from prior lines and we have the following result.

Lemma 1. *Let L(m) be an L-circuit of width m. L(m) can be used to compute the prefix computation for an input of width n<m.* ■

Now we consider the existence of faulty nodes in *L*-circuit. Let the faulty node be in line $i \leq m$. Since the faulty node in line i contributes to all the outputs in all lines j, $i \leq j \leq m$, then these outputs will be affected. In other words, all lines j, $i \leq j \leq m$, cannot be used to produce an output. However, this has no effect on the outputs produced through all lines $k < i$. Thus, the circuit can generate the prefix computation for inputs of size at most $n \leq i-1$. However, the prefix computation will require $d(L)$ time steps, the depth of the circuit. Thus, we have the following result.

Lemma 2. *Let L(m) be an L-circuit of width m and depth d(L). If there is a faulty node in line i, then L(m) can be used to compute the prefix computation for an input of width h<i using the first h lines in d(L) time steps.* ■

Lemma 2 shows that $L(m)$ with a faulty node still have the same depth to compute prefix computation for an input of size *h<i and* requires $d(L)$ time steps. In an *L*-circuit with no faulty nodes, and for input size of size *h<i*, the *L*-circuit would have a depth that is less than the depth in the case of size of input is *m*. Now, we investigate the penalty of using $L(m)$ with a faulty node at line *i* in computing the prefix for input of size *h<i*.

The $L(m)$ circuit is shown to have a depth, $d(L)= \lceil m/2 \rceil$. If $L(m)$ is used for prefix computation for input of size *h<i*, then this implies a penalty of $\lceil m/2 \rceil - \lceil h/2 \rceil$ time steps, *h<i* and consequently we have the following result.

Corollary 3. *Let L(m) be an L-circuit of width m and depth d(L). If there is a faulty node in line i, then the time penalty of using L(m) to compute the prefix computation for an input of width h<i is* $\left(\left\lceil \frac{m}{2} \right\rceil - \left\lceil \frac{h}{2} \right\rceil\right)$ *time steps.* ■

Now we consider the number of idle operation nodes in the *L*-circuit if there exists a faulty node in line *i*. Recall the structure of the *L*-circuit, starting the third line, lines alternates in having either one operation node or two operation nodes. If there is a faulty node in line *i*, then all nodes in line *j*, $i \leq j \leq m$, are idle nodes. Thus. It is straightforward to show that the total number of idle nodes, *N*, in all lines from line *i* to line *m*, $\left(N = \left\lceil \frac{3}{2}(m-i) \right\rceil\right)$.

Corollary 4. *Let L(m) be an L-circuit of width m and depth d(L). If there is a faulty node in line i, then the number of idle nodes N is* $\left\lceil \frac{3}{2}(m-i) \right\rceil$. ■

Corollary 3 and Corollary 4 show that, if $L(m)$ has a faulty node in line *i*, then the penalty of using the circuit to compute the prefix computation for *input of width h<i is* $\left(\left\lceil \frac{m}{2} \right\rceil - \left\lceil \frac{h}{2} \right\rceil\right)$ *time steps while the number of idle nodes is* $\left\lceil \frac{3}{2}(m-i) \right\rceil$.

3.2 Analysis of the H-circuit

In this section we consider the analysis of the *H*-circuit [4]. The *H*-circuit is a parallel prefix circuit that performs well when the input size *n* is larger than the width of the circuit *m*. In this case, the input is divided into subsets and presented to the circuit in consecutive time steps. The circuit generates the prefix computation in chunks after $d(H)$ time steps in consecutive time steps as well. Figure 3 shows an *H*-circuit of width 9.

We first show that for $H(m)$, the circuit can be used for prefix computation of size *n*, $n < m$. Similar to the *L*-circuit, we find that there are no links connecting two lines *i* and *j*, $i < j$, and also the node level in line *j* < node level of line *i*. Thus, any line *j* does not affect the output of line *i*, $i < j$. The output of line *i* is only affected by inputs from prior lines. The only exception is the link connecting the output of operation node at line *m* and the first duplication node at line 1. However, this link is used to to handle an input size of width $n > m$, where the data is sent from line *m* back to line 1.

Consider the existence of faulty nodes in *H*-circuit. Let the faulty node be in line $i \leq m$. Since the faulty node in line *i* affects all the outputs in all lines *j*, $i \leq j \leq m$, then *H* cannot be used to produce an output in line *j*, $i \leq j \leq m$. Thus, the circuit can generate the prefix computation for inputs of size at most $n \leq i-1$. However, the computation requires $d(H)=2\lceil \log(m-1) \rceil + 1$ [4] time steps. Thus, we have the following result.

Lemma 6. *Let H(m) be an H-circuit of width m and depth d(H). If there is a faulty node in line i, then H(m) can be used to compute the prefix computation for an input of width h<i using the first h lines in d(H)=* $2\lceil \log(m-1) \rceil + 1$ *time steps.* ■

Consequently, we have the following corollary.

Corollary 7. *Let H(m) be an H-circuit of width m and depth d(L). If there is a faulty node in line i, then the time penalty of using L(m) to compute the prefix computation for an input of width h<i is* $(2\lceil \log(\mathrm{m}-1) \rceil - 2\lceil \log(\mathrm{h}-1) \rceil)$ *time steps.* ■

To estimate the number of idle operation nodes in the *H*-circuit in case a faulty node exists in line *i*, we investigate the structure of the H-circuit. Recall the structure of the *H*-circuit, we find that it uses two binary tree-like structures (see Figure 3). If there exists a faulty node in line *i*, then we show that the total number of nodes in line *j*, $i \leq j \leq m$, by estimating the number of idle nodes in both binary trees in addition to a single operation node in level $\log(m-1)+1$ that will be idle as well. For the first binary tree, T_1, that is connected to the inputs of *H*, the total number of nodes is *m*-1. If the line number, *l*, is even number, then there is no nodes in T_1. If the line number, *l*, is odd and equals to 2^h+1, $2 \leq h \leq \log m - 1$, then the number of nodes in line *l* is equal to $\log l$ -1. The rest of the odd line numbers has only one node. Thus, we estimate the total number of idle nodes, $N_I(1)$, if there exists a faulty node in line *i* as $N_I(1)= \sum_{l=i,l\ is\ odd}^{m} 1 + \sum_{l=2^h+1,l\ is\ odd,l>i}^{m} \log l - 1$.

For the second binary tree, T_2, that is connected to the output of *H*, there is a one operation node in each line. Thus, the total number of idle nodes $N_I(2)= \sum_{l=i}^{m} 1= m-i+1$.

Thus the total number of idle nodes if a faulty node exists in line *i* is $N_I= N_I(1)+ N_I(2)+1$. Figure 4 shows an *F*-circuit with idle nodes where the faulty node is in line 5.

Lemma 8. *Let L(m) be an L-circuit of width m and depth d(L). If there is a faulty node in line i, then the number of idle nodes* $N_I(1)+ N_I(2)+1$. ■

The above results show that, if there is a faulty node, both the *L*-circuit and the *H*-circuit can perform the prefix computation for an input of smaller size. However, the *H*-

circuit loses its advantage in computing the prefix for an input of width larger than the width of the circuit. Then we can conclude that the *L*-circuit can better handle faulty nodes.

Corollary 9. *Let L(m) be an L-circuit and H(m) be an H-circuit of the same width m. If there is a faulty node in line i<m, then L can handle the faulty nodes better than H(m).* ■

4. Design of New Prefix circuits

In this section we propose new parallel prefix circuits that can handle faulty nodes better. The proposed designs are based on the analysis done in section 3 for the *L*-circuits and the *H*-circuits.

4.1 Fault tolerant L-circuit

In this section we consider the *L*-circuit that was investigated in section 3.1. The importance of the *L*-circuit is that it belongs to the class of depth-size optimal circuits if the fan-out is 2. The *L*-circuit is shown to have good performance when the input size is the same as the width of the circuit.

Recall the analysis done in section 3.1, if a faulty node exists in line *i*, then *L*-circuit can be used to generate prefix computation for input of size *i*-1. It follows that if the faulty node exists in a line that is closer to line *m*, then the number of idle nodes is smaller. We will use this observation in designing a Fault tolerant *L*-circuit (*FL*-circuit) that can better handle faulty nodes.

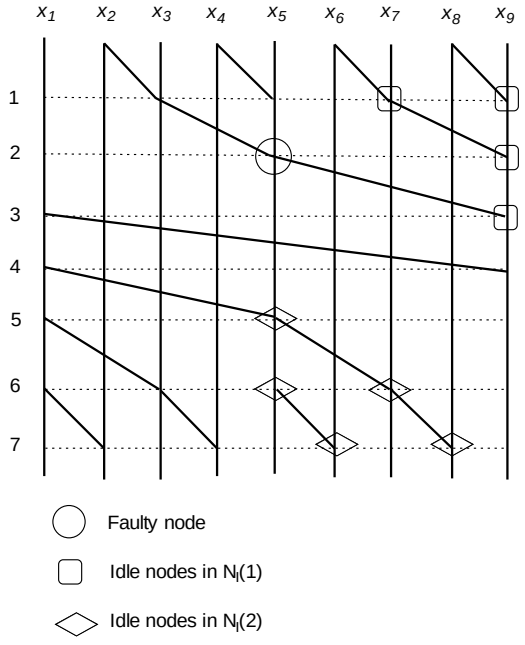

Fig. 4 Idle nodes in H(9).

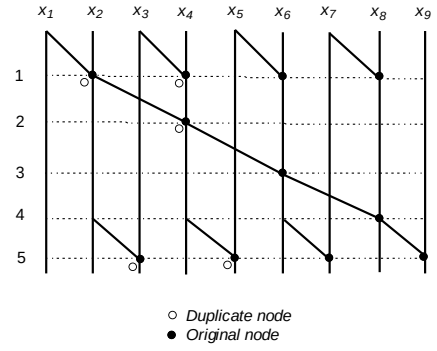

Fig. 5 Prefix circuit FL(9).

Definition 1. A parallel prefix circuit is called an *FL-circuit*, if it has the same structure as *L*-circuit [7] in addition to one duplicate node for each of the original nodes in the first $\left\lceil \frac{m}{2} \right\rceil$ lines. ■

Our proposed design, *FL-circuit*, depends on having a duplicate operation node for each original operation node in the first few lines. A duplicate operation node will be used if the original operation node is faulty. Such duplicate node if used (if a node is faulty), will save a large number of idle operation nodes in later lines. For example, having a duplicate node in line 2 will save the whole circuit if the node in line 2 is faulty. Thus, *FL-circuit* will be equipped with a number of duplicate nodes for the first *m*/2 lines. If any original node in line $i< \left\lceil \frac{m}{2} \right\rceil$ goes faulty, then the duplicate node will be active and the whole circuit operate as originally designed.

If any original node in line $i> \left\lceil \frac{m}{2} \right\rceil$ goes faulty, then the circuit can generate the prefix computation for an input of width $h<i$ (Lemma 2). Figure 5 shows an *FL-circuit* of width 9. Consequently, we have the following theorem.

Theorem 10. *Let FL(m) be an FL-circuit of width m and depth d(FL). If there is a faulty node in line* $i<\left\lceil \frac{m}{2} \right\rceil$ *(resp.* $i\geq\left\lceil \frac{m}{2} \right\rceil$*), then FL(m) can generate prefix computation for input of width m (resp. h<i) in* $\left\lceil \frac{m}{2} \right\rceil$ *time steps. Moreover, FL(m) has* $(\frac{m-3h}{2}+2)$ *duplicate nodes.*

Proof. We first assume that the faulty node is in line $i<\left\lceil \frac{m}{2} \right\rceil$. Since *FL*-circuit has duplicate nodes (one duplicate node for each original node) in the first $\left\lceil \frac{m}{2} \right\rceil$ lines, then any faulty node in the first $\left\lceil \frac{m}{2} \right\rceil$ lines can be replaced by a duplicate node and the *FL*-circuit can work like the original *L*-circuit and can generate prefix computation in $\left\lceil \frac{m}{2} \right\rceil$ time steps. If the faulty node is in line $i\geq\left\lceil \frac{m}{2} \right\rceil$, Then circuit also can work like the original *L*-circuit with a faulty node. By Lemma 2, *FL*-circuit can work like the

original *L*-circuit and can generate prefix computation in $\lceil \frac{m}{2} \rceil$ time steps.

For the number of duplicate nodes of *FL*(*m*), recall the structure of the *L*-circuit, it is straightforward to have $(\frac{m-3h}{2} + 2)$ duplicate nodes. ■

Theorem 10 applies also if there is more than one faulty node in the first $\lceil \frac{m}{2} \rceil$ lines.

4.2 Fault tolerant H-circuit

In this section we consider designing the fault tolerant *H*-circuit. The *H*-circuit belongs to the class of circuits that performs well when the input size is larger than the width of the circuit. We showed that, if there is a faulty node, then *H*-circuit loses its advantage in handling larger-width input. To try keep the advantage of the *H*-circuit in dealing with larger input width, we propose to equip the *H*-circuit with a duplication node for each original operation node. A duplicate operation node will be used if the corresponding original operation node is faulty. Such duplicate node if used will save the operation of the circuit. Thus, if any number of nodes in the *H*-circuit go faulty, then the circuit can still be used to generate the prefix computation as it is originally planned.

Definition 2. A parallel prefix circuit is called an *FH-circuit*, if it has the same structure as *H*-circuit [4] in addition to one duplicate node for each of the original nodes. Since the size, *s*(*H*)= 2*m*-3 [4], then the number of duplicate nodes is 2*m*-3. ■

Theorem 11. *Let FH(m) be an FH-circuit of width m and depth d(FH). If there is a faulty node in line* $1 \leq i \leq m$*, then FL(m) can generate prefix computation similar to the circuit H(m). Moreover, FH(m) has* 2*m* -3 *duplicate nodes.* ■

5 Simulation of Prefix Circuits

In this section, we simulate the *L*-circuit and the *H*-circuit on FPGA using Verilog. The prefix operation was selected to be addition. The target device is Xilinx xc3s500e-5-fg320 running on a clock of 50 MHz. For both the L-circuit and H-circuit, three circuits were simulated of width 5, 9, and 17.

For the L-circuit, L(17), L(9) and L(5) were implemented and their performance were assessed. Then a faulty node is assumed in L(9) at line 6 and the time penalty was computed if L(9) is used for computing the prefix operation for an input of width < 6. Also a faulty node is assumed in L(17) at line 10 and the time penalty was computed if L(17) is used for computing the prefix operation for an input of width <10. Table 1 shows the results of L-circuit simulation. The results shows that if L(9) has a faulty node and is used to compute the prefix for input of width < 6, then the circuit has 66.6% time more than the time in case the L(5) is used. Also, if L(17) has a faulty node and is used to compute the prefix for input of width < 10, then L(17) has 80% time more than the time in case the L(9) is used.

Table 1: Simulation results of L-Circuit

	L-circuit		
	L(17)	L(9)	L(5)
Time Delay	12.3642 ns	6.869 ns	4.1214 ns
Time Penalty	80%	66.6%	-

For the H-circuit, H(17), H(9) and H(5) were implemented and their performance were assessed. Then a faulty node is assumed in H(9) at line 6 and the time penalty was computed if H(9) is used for computing the prefix operation for an input of width < 6. Also a faulty node is assumed in H(17) at line 10 and the time penalty was computed if H(17) is used for computing the prefix operation for an input of width <10. Table 2 shows the results of H-circuit simulation. The results shows that if H(9) has a faulty node and is used to compute the prefix for input of width < 6, then the circuit has 40% time more than the time in case the H(5) is used. Also, if H(17) has a faulty node and is used to compute the prefix for input of width < 10, then H(17) has 28.6% time more than the time in case the H(9) is used.

Table 2: Simulation results of H-Circuit

	H-circuit		
	H(17)	H(9)	H(5)
Time Delay	12.3642 ns	9.6166 ns	6.869 ns
Time Penalty	28.6%	40%	-

6 Conclusions

In this paper, we have investigated different classes of parallel prefix circuits in the existence of faulty nodes in terms of time penalty and number of idle nodes. The analysis shows that some classes of prefix circuits can handle faulty nodes better than others. Also, we have proposed new designs for prefix circuits that can handle faulty nodes. The idea is based on having some duplicate nodes that can be active when some nodes go faulty.

One direction to extend this work is to analyze other classes of circuits. Other directions include proposing other designs that could handle faulty nodes.

Acknowledgment

This work is done under the grant received (82/40) by Deanship of research at Islamic University of Madinah (IUM) for application- based research. We also give special thanks to the administration of IUM for their support in every aspect.

References

[1] Jaja J (1992) An introduction to parallel algorithms. Addison Wesley, Redwood City, CA, USA.

[2] Leighton FT (1992) Introduction to parallel algorithms and architectures: arrays, trees, hypercubes. Morgan Kaufmann, San Mateo

[3] Vaidyanathan R, Trahan JL (2004) Dynamic reconfiguration: architectures and algorithms, vol 13.Springer Series in Computer Science. Kluwer Academic/Plenum Publishers, New York

[4] 4 .Lin Y-C, Hung L-L (2009) Fast problem-size-independent parallel prefix circuits. J Parallel Distrib Comput 69:382–388

[5] Hatem M El-Boghdadi (2013), A class of almost-optimal size-independent parallel prefix circuits. J Parallel Distrib Comput 73:888–894

[6] Hatem M. El-Boghdadi (2015), Dynamic-width reconfigurable parallel prefix circuits. The Journal of Supercomputing 71(4): 1177-1195

[7] Y.-C. Lin, C.-K. Liu, Finding optimal parallel prefix circuits with fan-out 2 in constant time, Inform. Process. Lett. 70 (4) (1999) 191–195.

[8] Y.-C. Lin, C.-C. Shih, Optimal parallel prefix circuits with fan-out at most 4, in: Proc. 2nd IASTED Int. Conf. on Parallel and Distributed Computing and Networks, Brisbane, Australia, 1998, pp. 312–317.

[9] T. D. Sawarkar, L. Chawle and N Narole, " Survey paper on Implementation of FPGA based Parallel Microprogrammed FIR Architecture", International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume 5 Issue 1, January 2016.

Hatem M. El-Boghdadi is a professor of Computer Engineering at Cairo University, Giza, Egypt since 2015. Currently, he is on leave at the Islamic University of Madinah, Saudi Arabia. His research interests include Parallel Architectures and Algorithms, Network-on-Chips, Reconfigurable Computing.

Dr. Fazal Noor received his PhD Engineering from McGill University, Canada 1993. He received his M. Eng ad B. Eng from Concordia University, in 1986 and 1984, respectively. Dr. Noor is working as Associate Professor in the Faculty of Computer Science and Information Systems, Islamic University of Madinah. Dr. Noor has numerous publications in International conferences and journals. His current research interests are in neural networks, optimization algorithms, image recognition, parallel and distributed computing, embedded systems, IoT, robotics.

Dr. Mahmoud, is currently a Professor in computer Sciences department at Islamic university of Madinah KSA, holds Ph.D in artificial intelligent from Cairo university, Dr. Mahmoud has a diverse educational and research background. His research expertise is in the expert systems, multimedia, data mining, ontology, intelligent systems, and knowledge base systems. He has been active in research with over 40 papers published in refereed journals and conferences. He is a senior member of IEEE.