Synthesization of Low Power Digital Signal Processor Architecture

[1]P. Krishna Kishore, [2]Y. Naveen Kumar, [3]P. Sreenivasulu, [4]P. Khesshawa Kumar

[1,2,3,4]Final year, B. Tech, Department of Electronics & Communications,
Geethanjali Institute of Science & Technology, Nellore.

***Abstract*: A Wireless Sensor Networks spatially distributed autonomous sensors to monitor physical or environmental conditions, such as temperature, sound, pressure, etc. Radio communication exhibits the highest energy consumption in wireless sensor nodes. This paper describes the design of the newly proposed folded-tree architecture for on-the-node data processing in wireless sensor networks, using parallel prefix operations and data locality in hardware.**

***Keywords*: Wireless sensor nodes, Folded-tree, Parallel prefix operations.**

I. INTRODUCTION

Each such sensor network node has typically several parts: a radio transceiver with an internal antenna or connection to an external antenna, a microcontroller, an electronic circuit for interfacing with the sensor and an energy source, usually a battery or an embedded form of energy harvesting. A sensor node might vary in size from that of a shoebox down to the size of a grain of dust, although functioning motes of genuine microscopic dimensions have yet to be created. The cost of sensor nodes is similarly variable, ranging from a few to hundreds of dollars, depending on the complexity of the individual sensor nodes.

MOTIVATION

To reduce power consumption and radiation effects in conventional Digital Signal Processor architecture and WSN.A low cost implementation of Digital Signal Processor for WSN. To reduce the complexity of system.

1.2 EXISTING SYSTEM

In existing method binary tree is used. The disadvantages of this method is at a time only one node act as root nodes, other node act as leaves. So at a time only one data is send. Hence the power as well as energy is increased. Time requirement is high and interconnection is high. The proposed approach gives the limited power and energy. The time requirement is low as well as interconnection path is increased. So Folded tree architecture is proposed to send the data in the way of wireless communication technique.

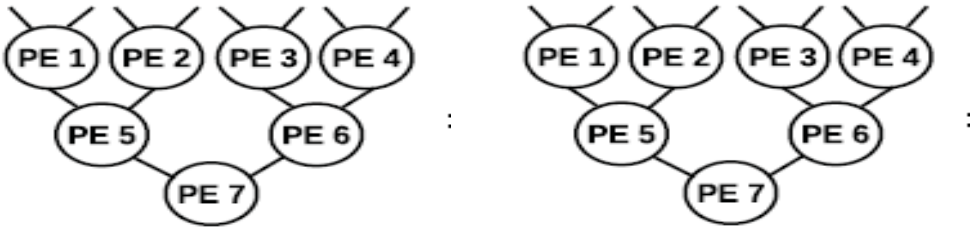


Fig 1.1: Binary tree

II. PROPOSED SYSTEM

Data communication must be traded for on-the-node processing which in turn can convert the many sensor readings into a few useful data values. Previously, we have shown how parallel prefix computations can be a common denominator of many WSN data processing algorithms. The goal of this paper is to design an ultralow-energy WSN digital signal processor by further exploiting this and other characteristics unique to WSNs.

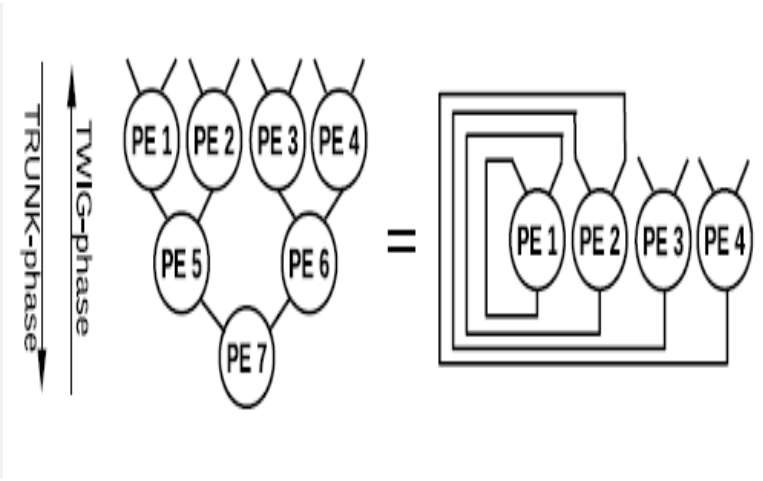


Fig2.1: Binary tree reuse as Folded tree

A. On-The-Node Data Aggregation

Common on-the-node operations performed on input data collected directly from the nodes sensors or through in-the-network aggregation include filtering, fitting, sorting, and searching.

We published earlier that these types of algorithms can be expressed in terms of parallel prefix operations as a common denominator. The tree-based data flow will, therefore, be executed on a data path of programmable PEs, which provides this flexibility together with prefix concept.

Parallel prefix operations

In the parallel prefix operations the addition of two inputs A and B in this case consists of three stages: a bitwise propagate-generate (PG) logic stage, a group PG logic stage, and a sum-stage. The outputs of the bitwise PG stage ($P_i = A_i \oplus B_i$ and $G_i = A_i \cdot B_i$) are fed as (P_i , G_i)-pairs to the group PG logic stage, which implements the following
expression:

$(P_i , G_i) \circ (P_{i+1}, G_{i+1}) = (P_i \cdot P_{i+1}, G_i + P_i \cdot G_{i+1})$ (1). It can be shown this operator has an identify element I =(1, 0) and is associative.

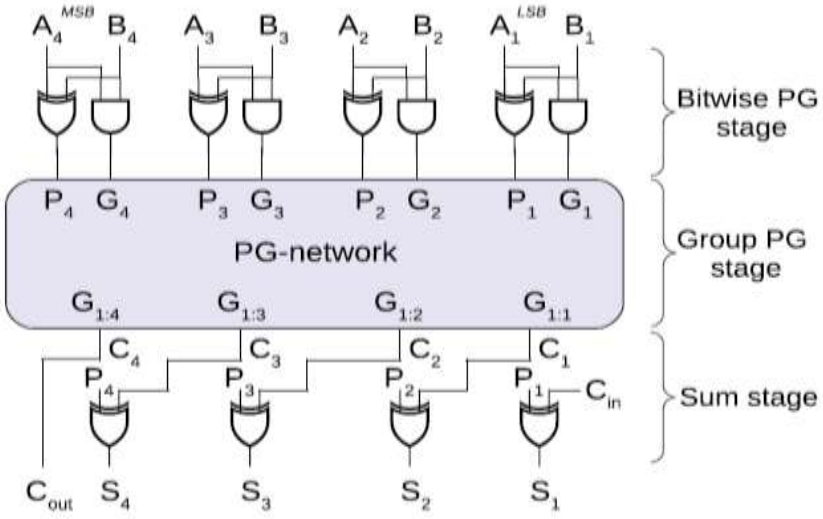


Fig 2.2: Addition with propagate-generate (PG) logic

For example, the binary numbers A = “0110” and B = “1010” are added together. The bitwise PG logic of LSB-firstnoted A = {0110} and B = {0101} returns the PG-pairs for these values, namely, (P, G) = {(0, 0); (0, 1); (1, 0); (1, 0)}. Using these pairs as input for the group PG-network, defined by the operator from (1) to calculate the prefix operation, results in the carry-array G = {0, 1, 0, 0}. In fact, it contains all the carries of the addition, hence the name carry look ahead. Combined with the corresponding propagate values Pi , this yields the sum S = {1011}.

Given a binary closed and associative operator with identity element I and an ordered set of n elements , the reduced-prefix set is the ordered set , while the all-prefix set is the ordered set , of which the last element is called the prefix element. The output of this parallel prefix PG-network is called the all-prefix set defined next.
.Blelloch's procedure to calculate the prefix-operations on a binary tree requires two phases. These are Trunk phase and Twig phase shown in below figure 2.3.

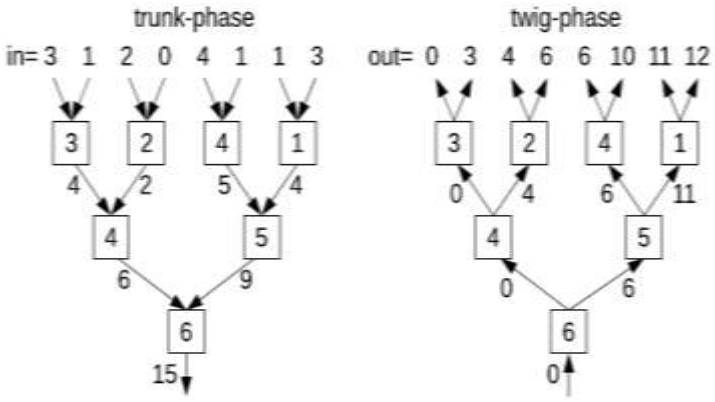


Fig 2.3: Trunk phase and Twig phase

In the trunk-phase, the left value L is saved locally as Lsave and it is added to the right value R, which is passed on toward the root. This continues until the parallel-prefix element 15 is found at the root. Note that each time, a store-and-calculate operation is executed. Then the twig-phase starts, during which data moves in the opposite direction, from the root to the leaves. Now the incoming value, beginning with the sum identity element 0 at the root, is passed to the left child, while it is also added to the previously saved Lsave and passed to the right child.

B. Programming using folded Tree

Generic patterns:

There are generic patterns that can be used to advantage for modelling business. These include entity types for Party (with included Person and Organization), Product type, Product instance, Activity type, Activity instance, Contact, Geo graphic area and Site. A model which explicitly includes versions of these entity classes will be both reasonably robust and reasonably easy to understand.
More abstract models are suitable for general purpose tools, and consist of variations on Thing type, with all actual data being instances of these. Such abstract models are on one hand more difficult to manage, since they are not very expressive of real world things, but on the other hand they have a much wider applicability, especially if they are accompanied by a standardized dictionary. More concrete and specific data models will risk having to change as the scope or environment changes.

Approach to generic data modelling:

One approach to generic data modelling has the following characteristics:

- A generic data model shall consist of generic entity types, such as 'individual thing', 'class', 'relationship', and possibly a number of their subtypes.
- Every individual thing is an instance of a generic entity called 'individual thing' or one of its subtypes.
- Every individual thing is explicitly classified by a kind of thing ('class') using an explicit classification relationship.
- The classes used for that classification are separately defined as standard instances of the entity 'class' or one of its subtypes, such as 'class of relationship'. These standard classes are usually called 'reference data'. This means that domain specific knowledge is captured in those standard instances and not as entity types. For example, concepts such as car, wheel, building, ship, and also temperature, length, etc. are standard instances. But also, standard types of relationship, such as 'is composed of' and 'is involved in' can be defined as standard instances.

This way of modelling allows the addition of standard classes and standard relation types as data (instances), which makes the data model flexible and prevents data model changes when the scope of the application changes.

Generic data model rules:

A generic data model obeys the following rules:

1. Candidate attributes are treated as representing relationships to other entity types.
2. Entity types are represented, and are named after, the underlying nature of a thing, not the role it plays in a particular context. Entity types are chosen. Thus a result of this principle, any occurrence of an entity type will belong to it from the time it is created to the time it is destroyed, not just whilst it is of interest. This is important when managing the underlying data, rather than the views on it used by applications. We call entity types that conform to this principle generic entity types.
3. Entities have a local identifier within a database or exchange file. These should be artificial and managed to be unique. Relationships are not used as part of the local identifier.
4. Activities, relationships and event-effects are represented by entity types (not attributes).

5. Entity types are part of a sub-type/super-type hierarchy of entity types, in order to define a universal context for the model. As types of relationships are also entity types, they are also arranged in a sub-type/super-type hierarchy of types of relationship.
6. Types of relationships are defined on a high (generic) level, being the highest level where the type of relationship is still valid. For example, a composition relationship (indicated by the phrase: 'is composed of') is defined as a relationship between an 'individual thing' and another 'individual thing' (and not just between e.g. an order and an order line). This generic level means that the type of relation may in principle be applied between any individual thing and any other individual thing. Additional constraints are defined in the 'reference data', being standard instances of relationships between kinds of things.

III. SOFTWARE REQUIREMENTS

A. XILINX

Migrating Projects from Previous ISE Software Releases

When you open a project file from a previous release, the ISE® software prompts you to migrate your project. If you click Backup and Migrate or Migrate Only, the software automatically converts your project file to the current release. If you click Cancel, the software does not convert your project and, instead, opens Project Navigator with no project loaded.

Note: After you convert your project, you cannot open it in previous versions of the ISE software, such as the ISE 11 software. However, you can optionally create a backup of the original project as part of project migration, as described below.

To Migrate a Project

1. In the ISE 12 Project Navigator, select File > Open Project.
2. In the Open Project dialog box, select the .xise file to migrate.

Note You may need to change the extension in the Files of type field to display .npl (ISE 5 and ISE 6 software) or.ise (ISE 7 through ISE 10 software) project files.

3. In the dialog box that appears, select Backup and Migrate or Migrate Only.
4. The ISE software automatically converts your project to an ISE 12 project.

Note If you chose to Backup and Migrate, a backup of the original project is created at project_name_ise12migration.zip.

5. Implement the design using the new version of the software.

Note Implementation status is not maintained after migration.

B. Creating a Project:

Venture Navigator permits you to deal with your FPGA and CPLD plans utilizing an ISE® venture, which contains all the source records and settings particular to your outline. To begin with, you must make a task and after that, include source documents, and set procedure properties. After you make an undertaking, you can run procedures to execute, compel, and break down your configuration. Venture Navigator gives a wizard to offer you some assistance with creating an undertaking as takes after.

Note: If you incline toward, you can make an undertaking utilizing the New Project dialog box rather than the New Project Wizard. To utilize the New Project dialog box, deselect the Use New Project wizard alternative in the ISE General page of the Preferences dialog box.

To Create a Project

1. Select File > New Project to launch the New Project Wizard.
2. In the Create New Project page, set the name, location, and project type, and click Next.
3. For EDIF or NGC/NGO projects only: In the Import EDIF/NGC Project page, select the input and constraint file for the project, and click Next.
4. In the Project Settings page, set the device and project properties, and click Next.
5. In the Project Summary page, review the information, and click Finish to create the project

Project Navigator creates the project file (project_name.xise) in the directory you specified.

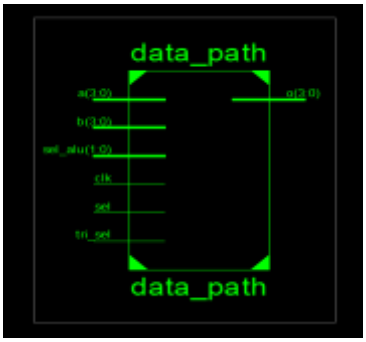
C. LANGUAGE VERILOG

In the semiconductor and electronic outline industry, Verilog is an equipment portrayal language(HDL) used to show electronic frameworks. Verilog HDL, not to be mistaken for VHDL (a contending dialect), is most generally utilized as a part of the outline, confirmation, and usage of digital rationale chips at the register-exchange level of reflection. It is likewise utilized as a part of the confirmation of analog and blended sign circuits.

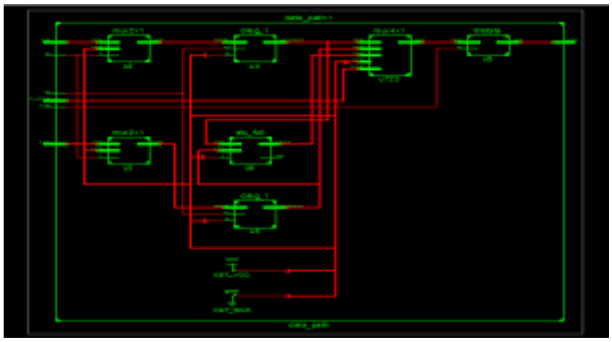
IV. RESULTS AND ANALYSIS

A. PROPAGATE GENERATE(PG)NETWORK

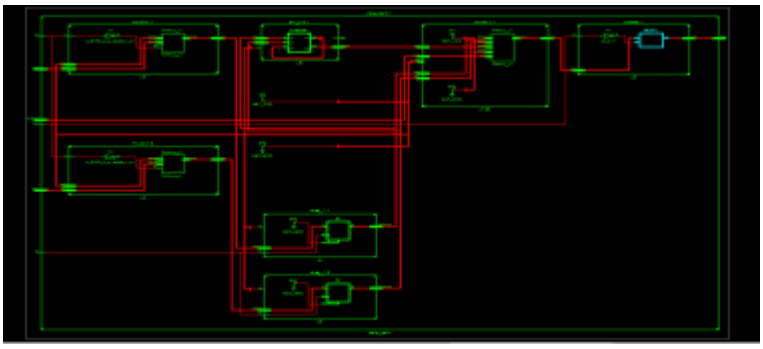
RTL schematic of PG-Network



Hierarchy 1 for RTL schematic of PG- Network



Hierarchy 2 for RTL schematic of PG-network



Hierarchy 3 for RTL schematic of PG-Network

Fig 3.1: RTL Schematics of Propagate Generate network

The above figures shows the RTL schematics of PG network. The hierarchy 1 of PG-network consisting of two inputs (a & b) and one output (o) and also it having the clock & selection inputs. Here tristate buffer logic was used in Group PG state. In this PG-network we used multiplexer (2x1 & 4x1) operations & full adders. The clear connection of this PG-network is shown in above Hierarchy 2&3.

Technology schematic of PG-network

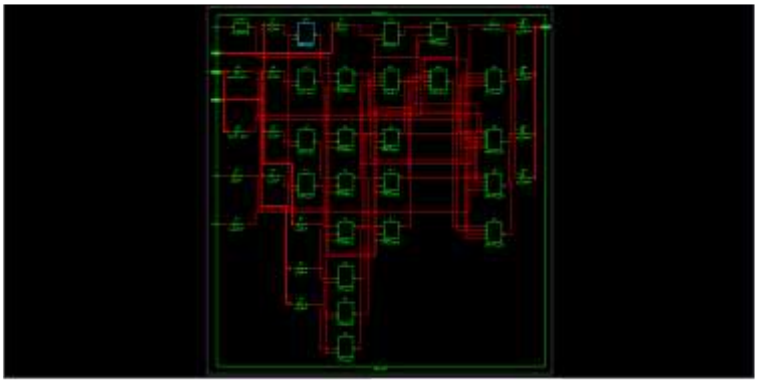


Fig 3.2: Technology schematic of PG-network

The technology schematic of PG-network having many look up tables (LUTs).These LUTs are used to performed the logic operations.

Simulation result of PG-network

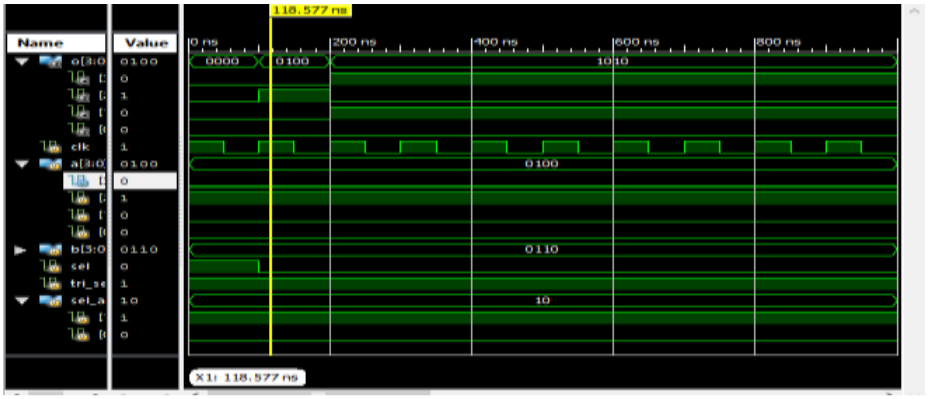
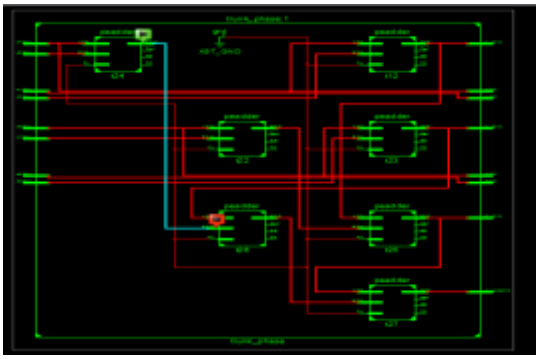


Fig 5.3: Simulation result of PG-network

The simulation result shows inputs and outputs in the form of wave forms. For example "a & b" considered as 0100 & 0110 respectively, then the out will be 0100. It is shown in above figure.

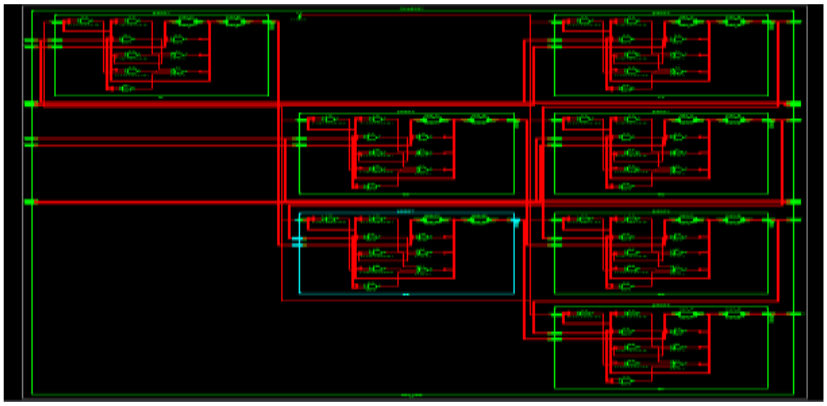
B. TRUNK PHASE

RTL Schematic of Trunk phase



Hierarchy 1 for RTL schematic of Trunk phase

Hierarchy 2 for RTL schematic of Trunk phase



Hierarchy 3 for RTL schematic of Trunk phase

Fig 3.4: RTL Schematic of Trunk phase

The above figure shows the RTL schematics of Trunk phase. In this trunk phase if you want to perform 8 input binary operation we require 7 Prefix Element (PE) adders. The connection between these PE adders shown in Hierarchy 2 & 3.

Technology Schematic of Trunk phase

Fig 3.5: Technology schematic of trunk phase

The technology schematic of Trunk phase having many look up tables (LUTs).These LUTs are used to performed the logic operations.

Simulation result of Trunk phase

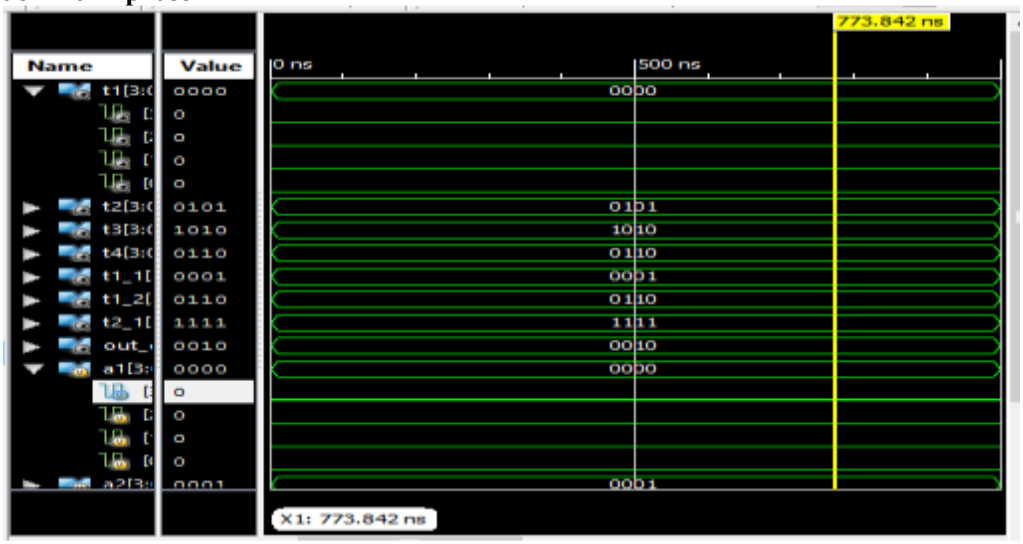
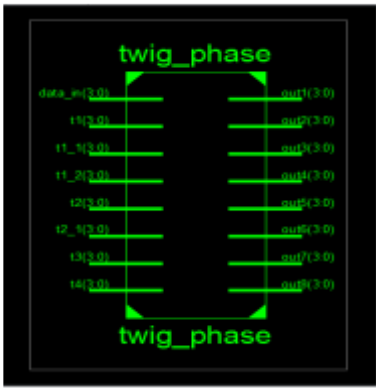


Fig 3.6: Simulation result of trunk phase

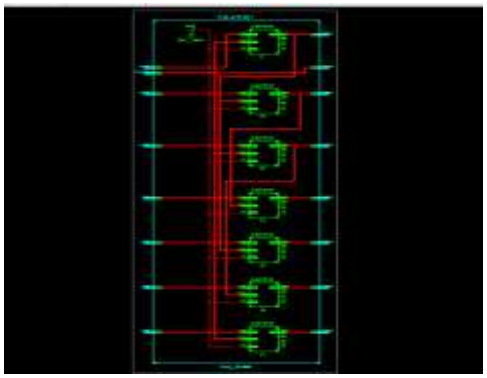
In the trunk phase initially 8 inputs are applied 4 PEs each prefix element having two inputs. The output of each prefix element connected to input of next prefix element. The operation of PE adder is first save left side value then perform addition operation with right side value and give the output at root of the prefix element.

C. TWIG PHASE

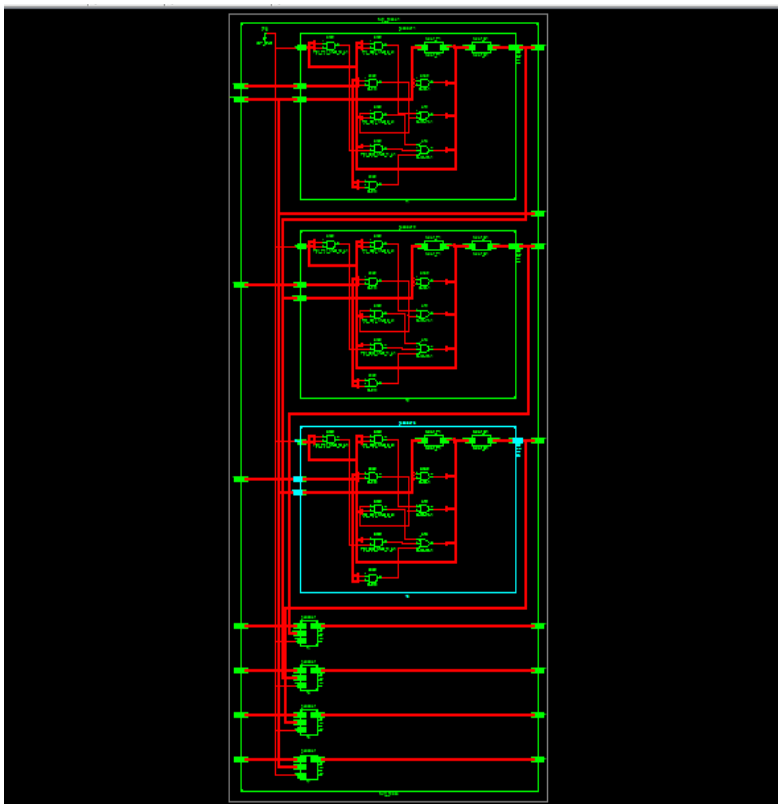
RTL schematic of Twig phase



Hierarchy 1 for RTL schematic of Twig phase



Hierarchy 2 for RTL schematic of Twig phase



Hierarchy 3 for RTL schematic of Twig phase

Fig 3.7: RTL schematic of twig phase

The above figures shows the RTL schematics of Twig phase. In this twig phase If you want to perform 8 input binary operation we require 7 Prefix Element (PE) adders. The connection between these PE adders shown in Hierarchy 2 & 3.

Technology schematic of Twig phase

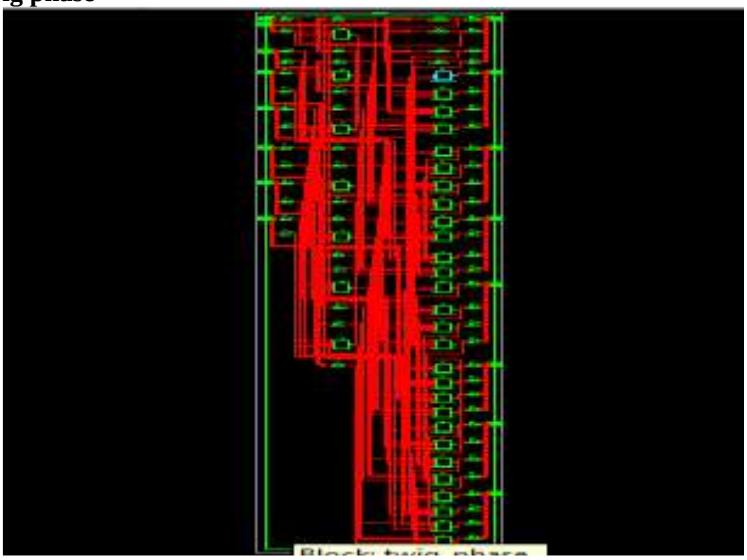


Fig 5.8: Technology schematic of twig phase

The technology schematic of Twig phase having many look up tables (LUTs).These LUTs are used to performed the logic operations.

Simulation result of Twig phase

Fig 3.9: Simulation result of twig phase

The incoming value, beginning with the sum identity element 0 at the root, is passed to the left child, while it is also added to the previously saved Lsave and passed to the right child this process continued up to displayed inputs. The example of twig phase output shown in above wave forms.

Result: **https://youtu.be/CYW-eXi0QXk**

V. ADVANTAGES AND APPLICATIONS

ADVANTAGES

- **Low cost:** The utility of the networks depends on high density of nodes. In order to make large scale deployments economically feasible, nodes must be of very low cost.
- **Small size:** For the same reasons, the size of modules must be of small size so that the network is unobtrusive.
- **Low power:** For large networks with many nodes, battery replacement is very difficult, expensive or even impossible. Nodes must have efficient energy so that it can function for long periods without running out of power.
- It avoids plenty of Wiring.
- It is flexible to undergo physical partitions.
- Suitable to non-reachable places such as over the sea, mountains, rural areas or deep forests.

- It might accommodate new devices at any time.

APPLICATIONS

Applications of wireless sensor network are in:

- **Security and Surveillance:** Security and surveillance are important applications of wireless sensor networks. Sensors can be used to improve the safety of roads by providing warnings of approaching cars at intersections. Image or video sensors can be very useful in identifying and tracking moving entities.

- **Industrial Monitoring**: Wireless sensors can be used to monitor manufacturing processes and conditions of industrial equipment to alert for imminent failures. This reduces cost for service and maintenance, increase machine up-time, improve customer satisfaction and even save lives.

- **Agriculture:** Wireless sensor network allow users to make precise monitoring of crops at the time of its growth. Hence, farmer can immediately know the state of the item at all its stages which will ease the decision process regarding the time of harvest.

- **Smart Home Monitoring:** Wireless sensors embedded within everyday object forming a wireless sensor network is used to monitor the activities performed in a smart home.

- **Environmental Monitoring:** Sensors can be used to monitor air quality and track environmental pollutants, wildfires or any other natural or manmade disasters. Sensors can also monitor biological or chemical hazards to provide early warnings.

VI. CONCLUSION

This paper presented the folded tree architecture of a digital signal processor for WSN applications. This design used to exploit the many data processing algorithms by using parallel prefix operations.

1) It is used to limiting the data set by pre-processing;
2) The reuse of the binary tree as a folded tree; and
3) The combination of data flow and control flow elements to introduce a local distribution.

The future Scope of this project is the end of architecture router is included. It is used to reduce the delay as well as congestion.

Bibliography

1). P. Sanders and J. Traff, “Parallel prefix (scan) algorithms for MPI,” in Proc. Recent Adv. Parallel Virtual Mach. Message Pass. Interf., 2006, pp. 49–57.

2). G. Blelloch, “Scans as primitive parallel operations,” IEEE Trans. Comput., vol. 38, no. 11, pp. 1526–1538, Nov. 1989.

3). N. Weste and D. Harris, CMOS VLSI Design: A Circuits and Systems Perspective. Reading, MA, USA, Addison Wesley, 2010.

3). J. Backus, “Can programming be liberated from the von neumann style?” in Proc. ACM Turing Award Lect., 1977, pp. 1–29.