B. TECH. PROJECT REPORT

On

IP Core Protection using Structural Obfuscation by mixing Pseudo Nodes

BY

Rishabh Verma, 160001049

DISCIPLINE OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY INDORE

NOVEMBER 2019

IP Core Protection using Structural Obfuscation by mixing Pseudo Nodes

A PROJECT REPORT

Submitted in partial fulfillment of the

requirements for the award of the degrees

of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

Submitted by:

Rishabh Verma, 160001049

Guided by:

Dr. Anirban Sengupta

INDIAN INSTITUTE OF TECHNOLOGY INDORE

NOVEMBER 2019

CANDIDATE'S DECLARATION

I hereby declare that the project entitled **"IP Core Protection using Structural Obfuscation by Pseudo Operation Mixing"** submitted in partial fulfillment for the award of the degree of Bachelor of Technology in 'Computer Science Engineering' completed under the supervision of **Dr Anirban Sengupta, Associate Professor, Computer Science and Engineering,** IIT Indore is an authentic work.

Further, I declare that I have not submitted this work for the award of any other degree elsewhere.

Rishabh Verma (160001049)

CERTIFICATE by BTP Guide

It is certified that the above statement made by the students is correct to the best of our knowledge.

Dr. Anirban Sengupta

Associate Professor

Computer Science and Engineering

IIT Indore

Preface

This report on “IP Core Protection using Structural Obfuscation by mixing Pseudo Nodes " is prepared under the guidance of Dr Anirban Sengupta.

We have tried to present the detailed concept of structural obfuscation. Through this report the explanation and all the transformation for obfuscation is shown diagrammatically. As well as the algorithm for obfuscating the design. For better understanding of my concept one example of obfuscation is added with all the steps from base case design i.e. design which is neither obfuscated nor secure. To conclude the report comparison of cost of obfuscated design with non-obfuscated is made along with strength of the achieved obfuscation.

Rishabh Verma, 160001049
B.Tech. IV Year
Discipline of Computer Science and Engineering
IIT Indore

Acknowledgements

We wish to thank Dr. Anirban Sengupta and Mr. Mahindra Rathore PhD Student for their kind support and valuable guidance.

It is their help and support, due to which we became able to complete the design and technical report.

Without their support this report would not have been possible. It was only possible because of their enthusiasm, beforehand schedule, knowledge and sincerity towards me and my work to produce the better result. We wouldn't have achieved my goals without their encouragement at each step.

Rishabh Verma, 160001049s
B.Tech. IV Year
Discipline of Computer Science and Engineering
IIT Indore

Abstract

Digital Signal Processing (DSP) kernels based intellectual property (IP) cores are important components of modern consumer electronics (CE). However, hardware threats such as reverse engineering and Trojan insertion have raised serious concerns about the protection of DSP kernels. Obfuscation is one of the protection mechanisms that make DSP design architecture un-obvious to interpret, thereby making it harder to reverse engineer. To tackle the problem of reverse engineering and trojan insertion. We proposed the approach of mixing pseudo operation in data flow graph of IP Core in order to obfuscate the data flow graph (DFG). By inserting and mixing pseudo operation we are obfuscating the DFG structurally thus making it unobvious to adversary making hardware attacks more complex against mentioned hardware attacks. In proposed approach insertion and hardware binding of pseudo operations is performed in such a manner that it results into minimal overhead. Proposed approach has been implemented and applied on various IP core; experimental results indicate that proposed approach yields minimal overhead with stronger obfuscation.

Table of Content

List of Figures

List of Tables

1. Introduction

With the mounting popularity of the reusable intellectual property (IP) cores, security threats such as reverse engineering, piracy and hardware Trojan insertion have become a serious problem for electronic designs due to globalization of supply chain. The global semiconductor supply chain for SoC/IC design is highly susceptible to hardware attacks, such as trojans and IP piracy. However, globalization in the semiconductor supply chain is inevitable due to requirements of maximizing design productivity and minimizing design cycle time. Further, keeping the entire design and manufacturing process of IC design in house increases the nonrecurring cost significantly, especially for mass-production commercial developments. It is estimated that 10% of the globally sold electronic products are counterfeited that leads to ~$100 billion of revenue loss. To tackle reverse engineering and trojan insertion obfuscation is used.

Obfuscation is a process of transforming an original design into its functionally equivalent form that significantly enhances the reverse engineering complexity.

Obfuscation classification:

1. **Structural Obfuscation:** is a technique by which the structure of an electronic hardware is modified intentionally to conceal its design, which makes it significantly more difficult to reverse-engineer. In other words, structural obfuscation modifies the design in such a way that the resulting architecture becomes un-obvious to an adversary.
2. **Functional Obfuscation:** This kind of obfuscation is performed by locking the functionality of the design using some key-gates. Without application of correct key sequence, the correct functionality cannot be revealed.

2. Prior Related Work

Works towards the protection of IP cores against hardware threats have contributed to two major classes of techniques such as watermarking [1]-[4] and obfuscation [5], [6], [7], [9], [10]. Watermarking is a passive mode of IP core protection where a secret mark is implanted into the design. It aims to protect the ownership of the design. Obfuscation aims to prevent reverse engineering attacks by obscuring a design. Watermarking approach cannot protect against reverse engineering-based threats [1]. [1] aims to protect IP core that are combinational circuit based. However, it is incapable to protect DSP kernels. Further, [2]-[4] target DSP kernels, however they do not handle protection against reverse engineering-based threats such as Trojan insertion. Additionally, [5], [6], [7] uses high level transformations (HLTs) based obfuscation to protect DSP kernels against RE threats. However, these approaches are HLT based which cannot be universally applied on all type of DSP kernels. Further, the protection achieved by HLT methods does not yield a robust obfuscation always. Finally, they may also incur design overhead after obfuscation. [9], [11] protect an IP core of combinational circuit type via logic obfuscation by inserting key gates into design. However, it suffers from several limitations: (a) in the context of DSP kernels, insertion of several key-gates may result into significant area overhead, (b) these approaches do not handle DSP kernels in their work, (c) the key based logic obfuscation approaches are vulnerable to several other forms of threats such as Boolean satisfiability (SAT) attack, removal attack, key sensitization attack etc.

Structural obfuscation methods do not suffer from above limitations. Our proposed work presents a minimal overhead obfuscation for DSP kernels that eliminates limitations of prior work.

3. Proposed Methodology

3.1 Overview

The proposed algorithm approach Fig 3.1 takes IP core as data flow graph along with the resource constraints and module library. Then it scheduled the data flow graph based on the input constraints provided and the scheduled DFG is passed to the pseudo operation determination algorithm (Algorithm 1), pseudo node determination algorithm based on the resource's constraints find out control steps which can accommodate pseudo nodes and returns table representing it. This table along with scheduled graph is passed to Pseudo operation mixing step. In this step pseudo operations are inserted and mixed into the scheduled graph based on the table returned from previous step. After processing this step obfuscated data flow graph is passed to hardware binding step after this step obfuscated design is forwarded to Datapath and controller synthesis step and finally obfuscated design of DSP core is obtained.

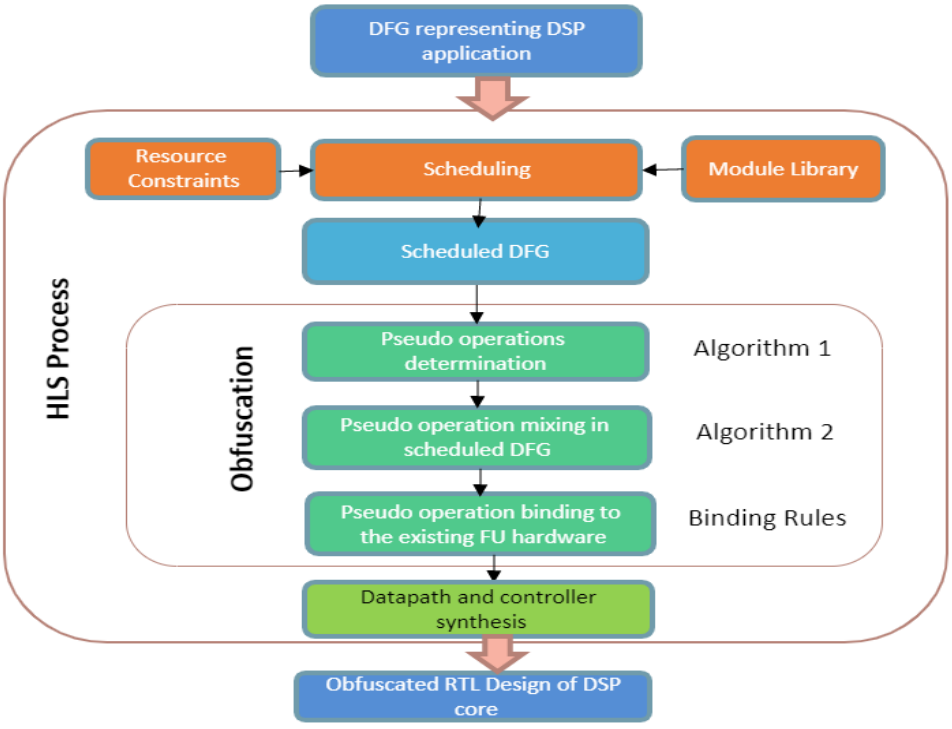

Figure 3.1: Design Flow of proposed algorithm

3.2 Details of the proposed algorithm

Algorithm 1: Pseudo Operation Determination

Inputs*: Initial scheduled DFG, Resource constraints*

Algorithm:

In each control step:

For each Resource type (Ex- adder, multiplier)

If (count of allocated resource < resource constraint/2)

Put 1 in table corresponding to resource type and control step

Else

Put 0 in table corresponding to resource type and control step

Output: *Table T1 with row representing control step and column as resources type*

Example:

Output Table

Control Step	Adder	Multiplier
1	1	1
2	1	0
3	0	1

Here in control step 1 under column adder signifies that pseudo operation of type adder can be inserted into control step 1 while 0 in control step 3 represent no pseudo operation of type adder can be inserted into this step.

Table 3.1

Table returned by Algorithm 1 along with the scheduled graph is passed to the Algorithm 2, which insert and mix the pseudo operations to the scheduled graph.

Algorithm 2: Pseudo Operation mixing in CDFG

Inputs: Initial scheduled DFG, table of CS numbers and their corresponding pseudo nodes

Algorithm: Select Control Step one by one from the table:

Insert the corresponding nodes (operations) in the selected control step

To insert pseudo operations in 1st control step

Assign primary inputs of any original operation in the same control step to the pseudo nodes

To insert pseudo operations in control other than 1st control step and no pseudo operations have been inserted into any of prior control steps

Assign output of any original operation in the previous control step to the inputs of pseudo operation

For remaining pseudo operations in the table

For each pseudo operation

Assign 1st input from the output of previous inserted pseudo node in the prior CS

Assign 2nd input from the output of any original operation in the previous CS

Output: Scheduled DFG with inserted pseudo-graph

The returned DFG from this algorithm is passed to pseudo operation binding phase. In this phase hardware is allocated to the inserted pseudo operations.

Rule for Pseudo Operations Binding

Inputs: CDFG after insertion of a pseudo nodes

for each control step:

for each type pseudo operation

Allocate the operation to the instance which have maximum number of unallocated inputs (unallocated inputs are calculated by taking difference of upper nearest power of 2 for count of allocated operation to resource instance)

if no inputs of any of the unallocated hardware are free then operation is allocated it to the resource instance which is least used.

end for

end for

Output: Obfuscated DFG is obtained

4. Demonstration of proposed approach on DWT DFG

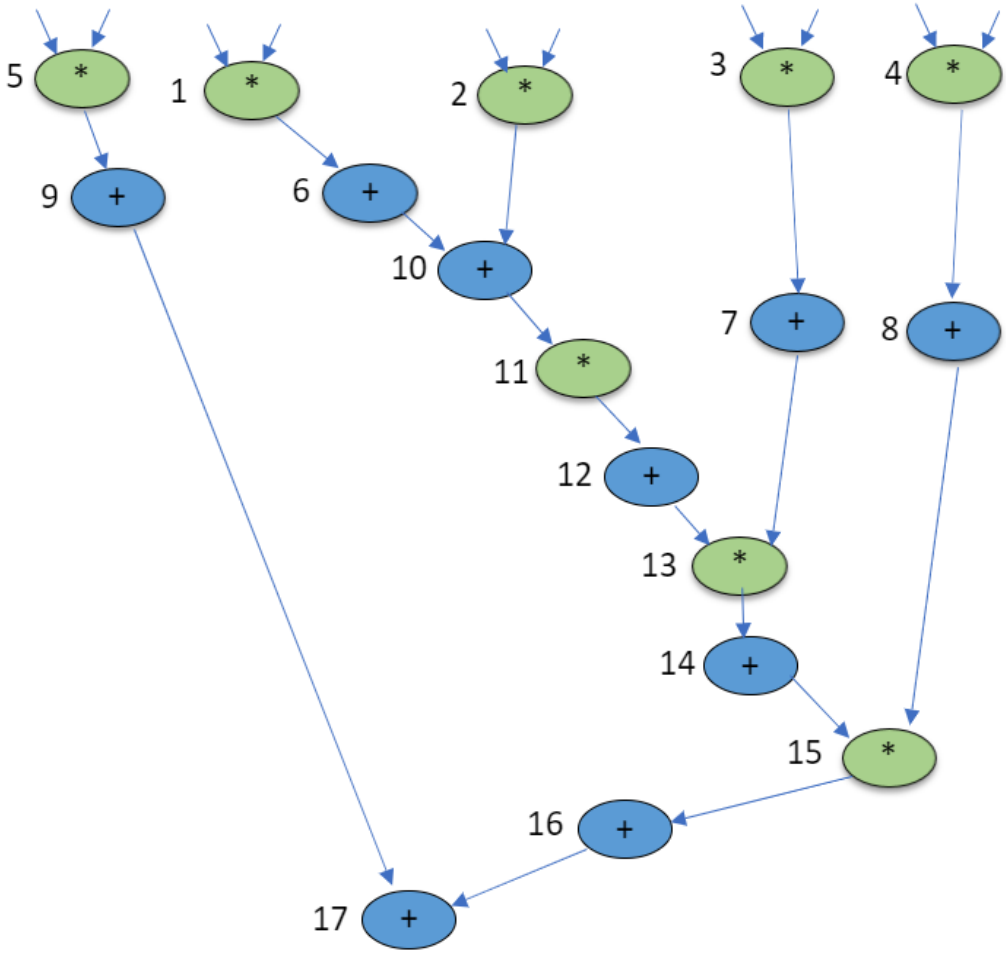

Fig 4.1: Input DFG along with constraints of 2 multiplier,1 Adder

The proposed approach takes input DFG of Discreate Wavelet Transform function shown in Fig 4.1 along with resource constraints of 2 multiplier, 1 adder and module library then it applies list scheduling algorithm based on the input constraints, which allocates control step to each node shown in Fig 4.2.After the scheduling CDFG in Fig 4.2 is passed to pseudo node determination phase of obfuscation in this step Algorithm 1 is applied to obtain table 3.3 which represents number and type of pseudo operation that can be inserted in each control step.

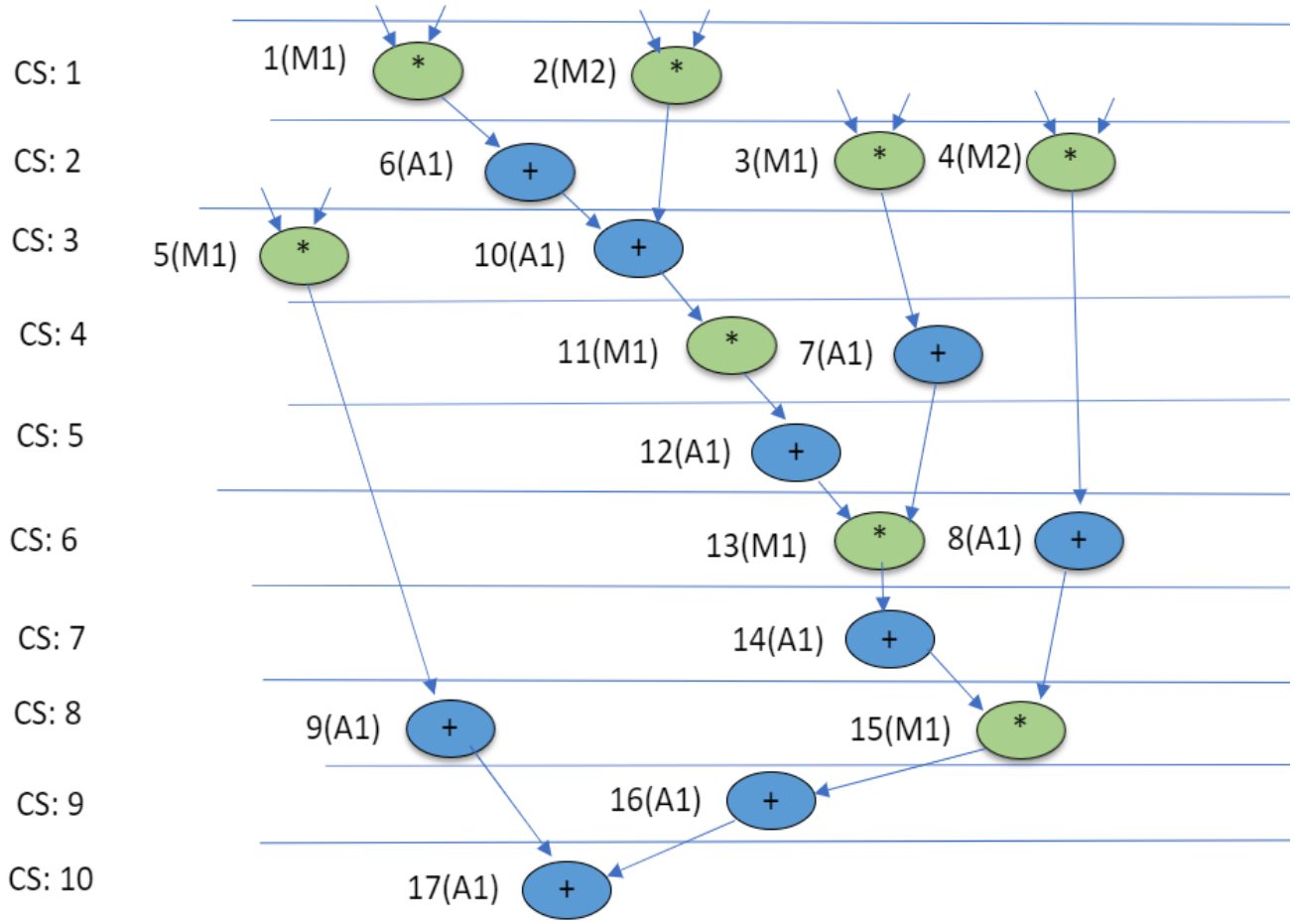

Fig.4.2: DFG After scheduling based on the resource constraints provided

Resource Instances	Input	Mux Size
A1	9	16
M1	6	8
M2	2	2

Table 3.2: Allocated Resource Instances Before Obfuscation

Control Step	Adder	Multiplier
1	1	0
2	0	0
3	0	0
4	0	0
5	0	1
6	0	0
7	0	1
8	0	0
9	0	1
10	0	1

Table 3.3: Table returned by Algorithm 1

After applying pseudo operation determination step. The second phase is applied that is pseudo operation mixing step. In this step algorithm 2 is applied which inserts and mix the pseudo operation in CDFG.As a result graph shown in fig 4.3 is obtained. Finally, binding rules are applied and obfuscated design is obtained shown in Fig 4.4.

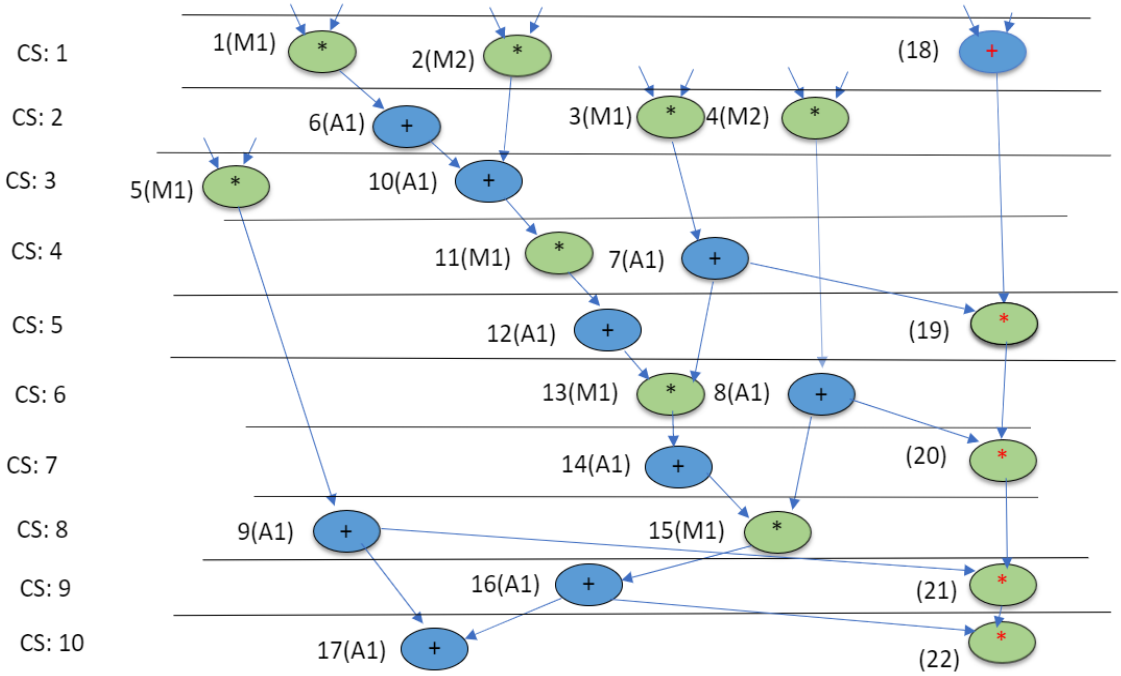

Fig 4.3: Scheduled graph obtained from Algorithm 2

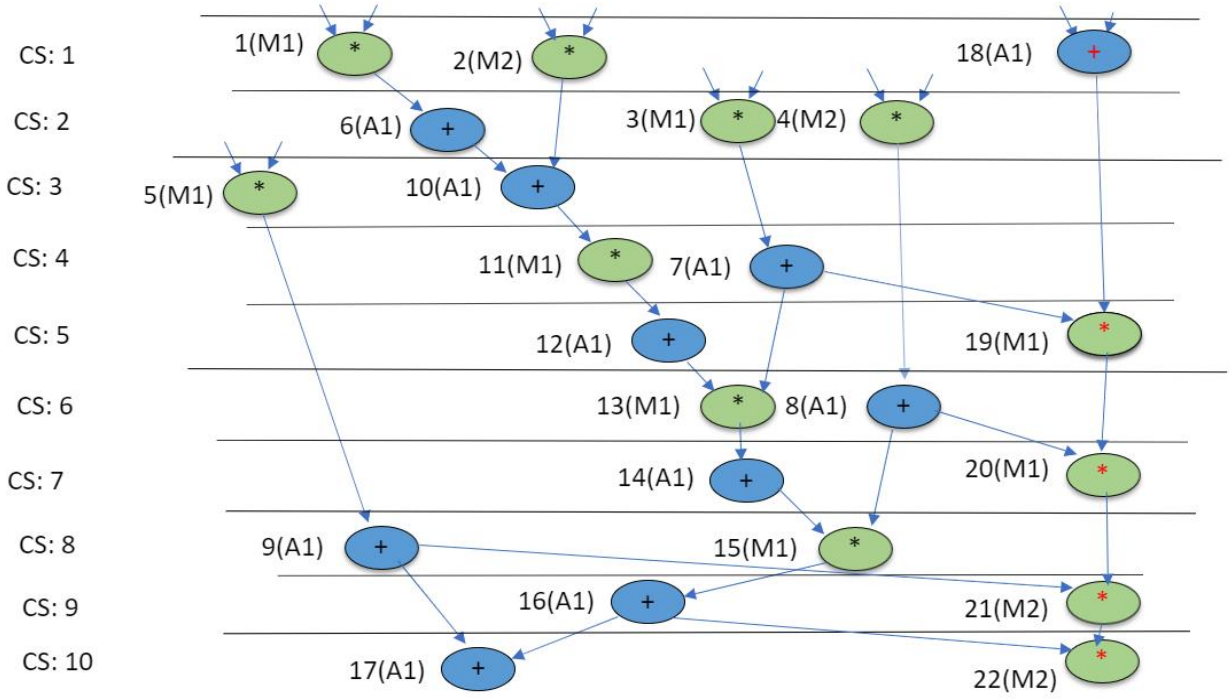

Fig 4.4: Obfuscated DFG obtained after allocating hardware allocation

Resource Instances	Inputs	Mux Size
A1	10	16
M1	8	8
M2	4	4

Table 3.4: Allocated Resource Instances after Obfuscation

Table 3.4 shows the allocated resource instance after applying proposed approach.

Pre-Obfuscation Datapath

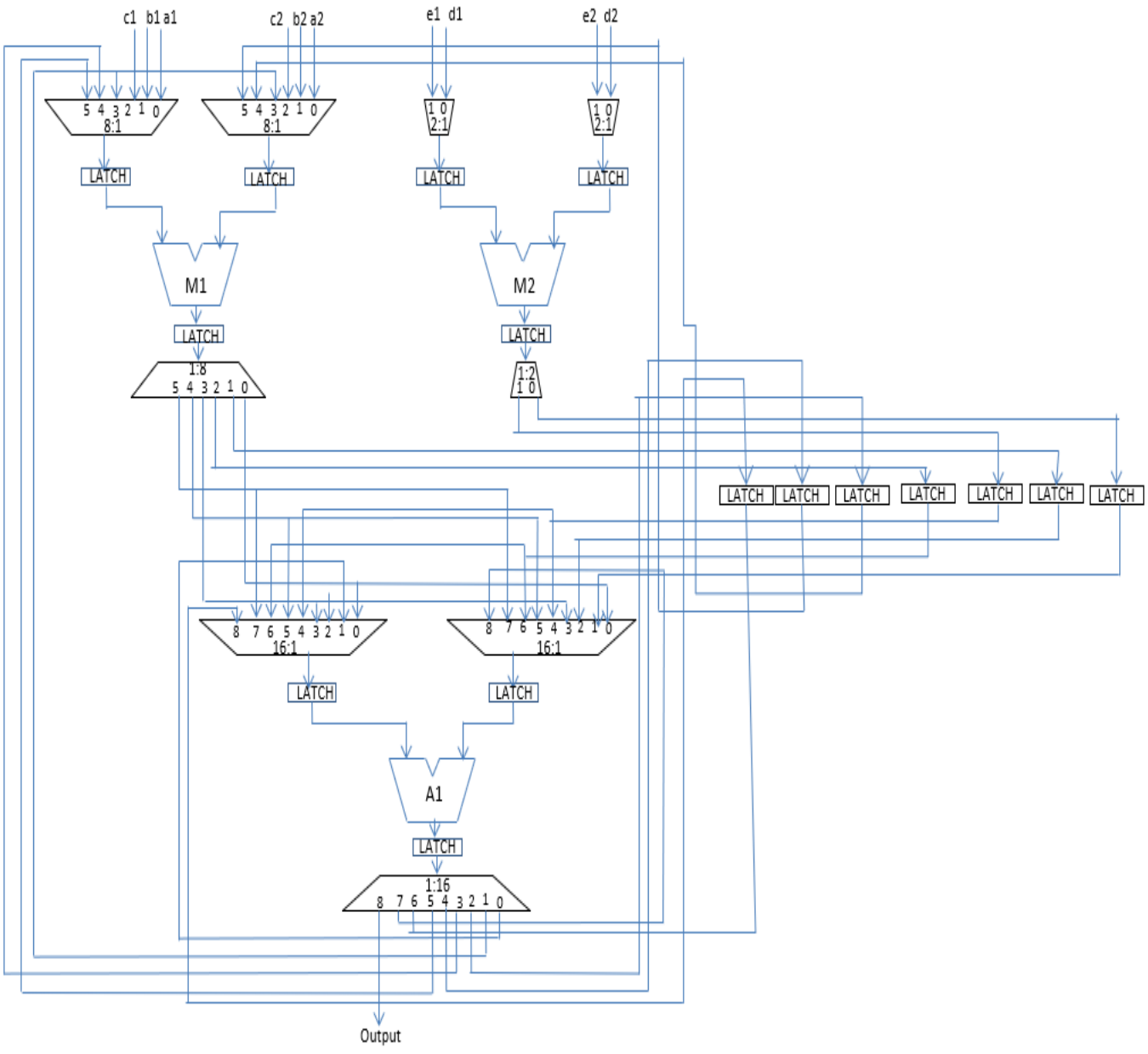

Fig 4.5

Post Obfuscation Datapath

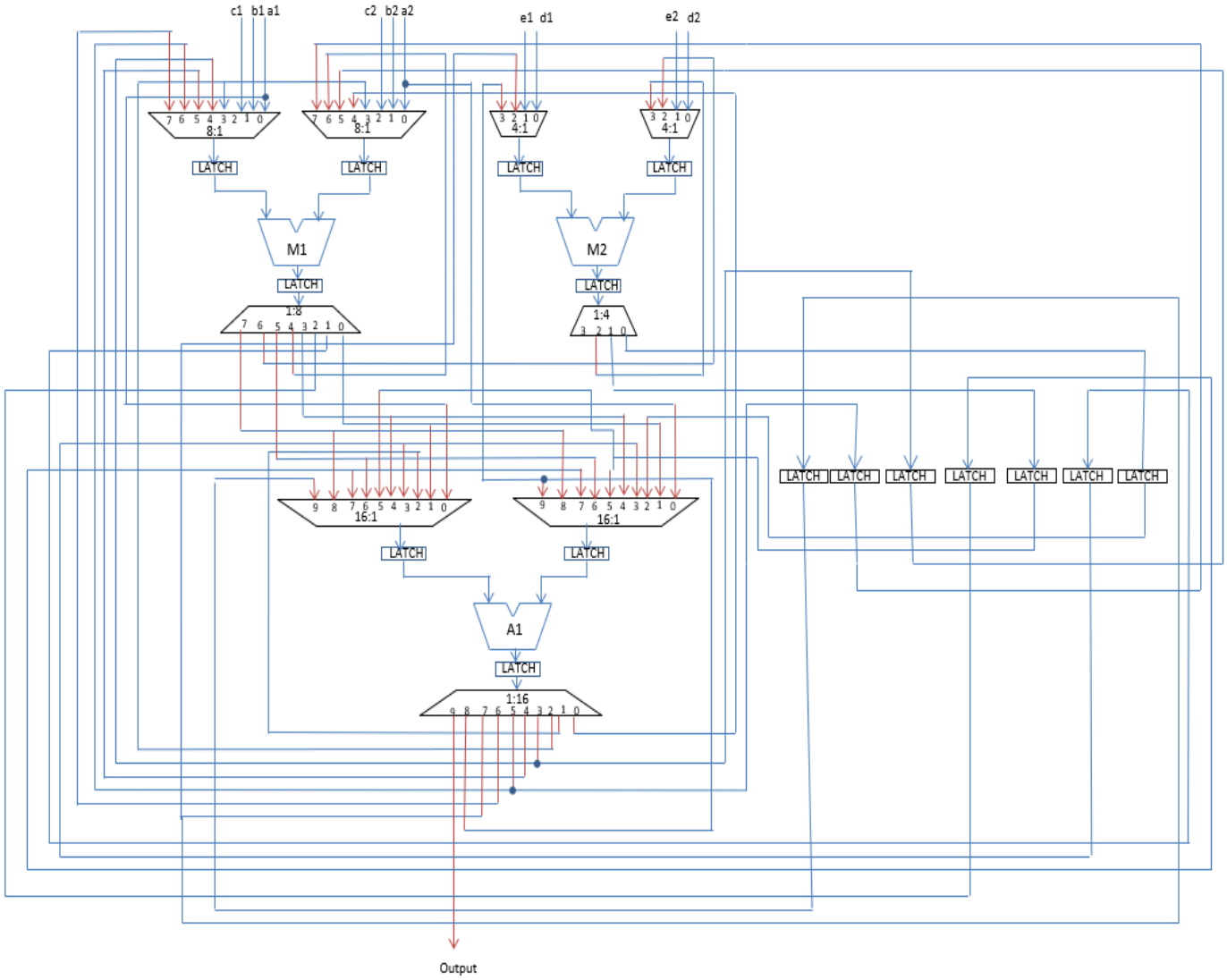

Fig 4.6

Red connections are the affected connections due to structural obfuscation.

Implementation:

The proposed approach is implemented in C++ and executed on Intel Core-i5-3210M CPU @ 2.3 GHz 2.4GHz. And Obtained Obfuscated DWT DFG is implemented on Altera Quartus II

5. Results

Synthesis Report

Pre-Obfuscation

Family	Cyclone II
Total logic elements	968 / 14448(7%)
Total combinational functions	966 / 14448(7%)
Dedicated logic registers	34 / 14,448 (< 1 %)
Device	EP2C15AF484C6

Post Obfuscation

Family	Cyclone II
Total logic elements	1,130 / 14448(8%)
Total combinational functions	1,125 / 14448(8%)
Dedicated logic registers	36 / 14,448 (< 1 %)
Device	EP2C15AF484C6

DSP Benchmark	Pre-Obfuscation Delay(ps)	Post Obfuscation Delay(ps)	Delay Overhead(ps)
IIR	2533.01	2533.01	0%
Mesa Horner	2191.54	2191.54	0%
JPeg	2477.02	2477.02	0%
MPeg	2278.29	2278.29	0%
DWT	2849.25	2874.75	0.89%
FIR_TAP_24	9612.90	9612.90	0%
Feedback point	3409.95	3409.95	0%

Table 4.2: Delay Pre/Post Obfuscation

$$Delay = \sum_{1}^{CS} (\max(delay\ of\ operator) + \max(delay\ of\ operator\ mux) + \max(delay\ of\ operator\ demux))$$

Table 4.2,4.3 shows comparison of delay and latency of design before obfuscation and after obfuscation.

DSP Benchmark	Pre-Obfuscation Area(us)	Post Obfuscation Area(us)	Area Overhead
IIR	121.21	121.21	0%
Mesa Horner	198.62	198.62	0%
JPeg	1339.50	1339.50	0%
MPeg	323.66	323.66	0%
DWT	213.96	213.96	0.89%
FIR_TAP_24	213.22	213.22	0%
Feedback point	262.00	262.00	0%

Table 4.3: Area overhead pre/post obfuscation

$$\text{Area} = \sum Area\ of\ operator + Area\ of\ Mux + Area\ of\ demux$$

DSP Benchmark	Pre-Obfuscation Gate Count	Post Obfuscation Gate Count
IIR	3648	3648
Mesa Horner	4192	4192
JPeg	29856	29856
MPeg	8272	8272
DWT	6288	6720
FIR_TAP_24	17152	17152
Feedback point	13408	13408

Table 4.4: Gate Count Pre/Post Obfuscation

$$\boldsymbol{Cost} = \mathrm{w1}\frac{L}{L_{max}} + w2\frac{A}{A_{max}}$$

- A and L is area and delay of design solution
- A_{max} , L_{max} indicate maximum values of area and delay of design solution in the design space
- w1 & w2 are the user defined weight. Both values are chosen to be 0.5 to assign equal preference

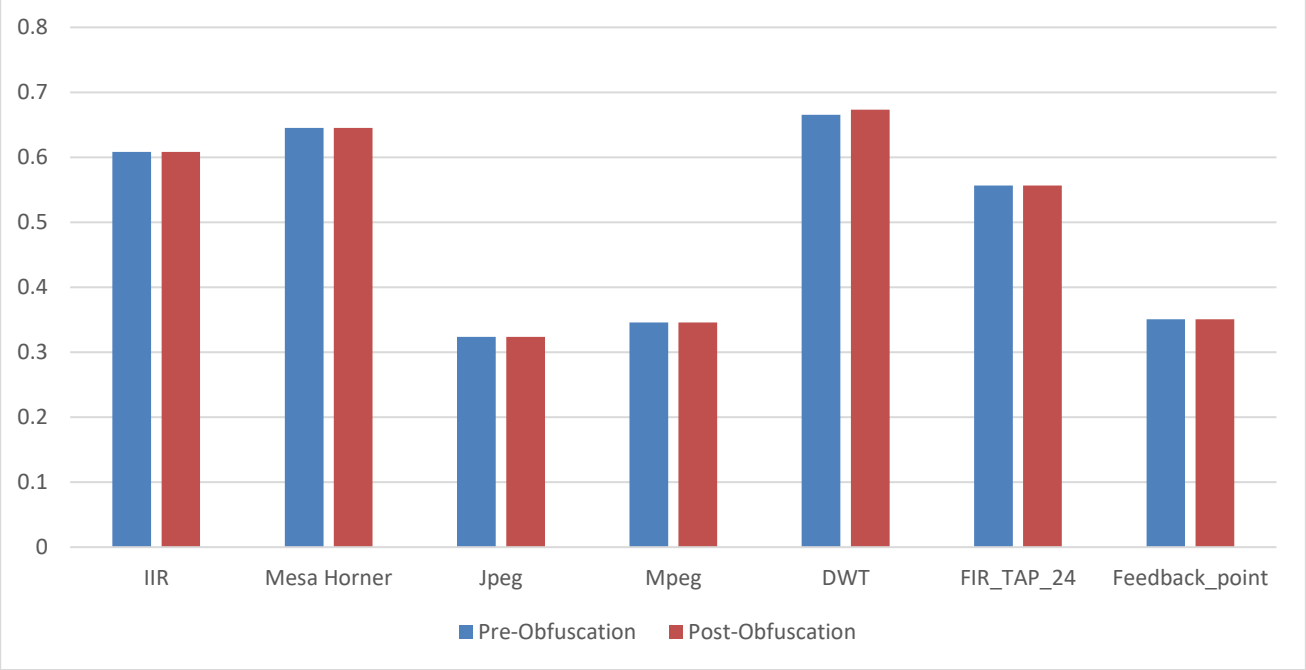

Fig 5.1: Pre/Post Obfuscation Cost Comparisons

$$Strength\ of\ obfuscation(RTL)\ (\%) = \frac{Number\ of\ affected\ inputs/outputs\ of\ mux/demux}{Total\ number\ of\ inputs/outputs\ of\ mux/demux} * 100\%$$

DSP Benchmark	Strength
IIR	56.25 %
Mesa Horner	44.4%
JPeg	9.84%
MPeg	30.55%
DWT	57.14%
FIR_TAP_24	40.62%
Feedback point	10.71%

Table 4.1: Security Strength (RTL)

$$Strength\ of\ obfuscation(Gate\ Level)\ (\%) = \frac{Gate\ count\ in\ affected\ Components}{Total\ number\ gate\ count} * 100\%$$

DSP Benchmark	Gate Count	Affected Gate Count	Strength of obfuscation
IIR	3648	3168	86.84%
Mesa Horner	4192	3168	75.57%
FFT	10752	3168	29.46%
JPeg	29856	8256	27.65%
MPeg	8272	4752	57.44%
DWT	6288	5472	87.02%
FIR_TAP_24	17152	16672	97.20%
Feedback point	13408	4752	35.44%

Table 4.5: Strength of Obfuscation at Gate level

6. Conclusion

Protection of IP cores is of prominence these days because of globalization of design supply chain. A designer at times may have to compromise with other design metrics such as area and delay during incorporating protection feature within an IP. This has always been a challenge for a designer to generate a highly secured design while keeping design overhead as minimal as possible. This paper illustrated how proposed work obtain obfuscated design with minimal overhead. Through our work we tried to bring new innovation in this field. This kind of work has never been proposed in the literature before. We have achieved very less overhead in terms of cost with results in the strong strength of obfuscation.

7. References

1. D. Kirovski, Y.-Y. Hwang, M. Potkonjak, and J. Cong, “Intellectual property protection by watermarking combinational logic synthesis solutions,” in Proc. Int. Conf. Comput.-Aided Design, Nov. 1998, pp. 194–198.

2. A. B. Kahng et al., “Watermarking techniques for intellectual property protection,” in Proc. 35th Annu. Design Autom. Conf., Jun. 1998, pp. 776–781.

3. D. Roy and A. Sengupta, “Low Overhead Symmetrical Protection of Reusable IP Core Using Robust Fingerprinting and Watermarking During High Level Synthesis,” Future Gener. Comput. Syst., vol. 71, no. C, pp. 89–101, Jun. 2017.

4. A. Sengupta and S. Bhadauria, “Exploring Low Cost Optimal Watermark for Reusable IP cores During High Level Synthesis,” IEEE Access, vol. 4, pp. 2198–2215, 2016.

5. A. Sengupta, D. Roy, S.P. Mohanty, and P. Corcoran, “DSP Design Protection in CE through Algorithmic Transformation based Structural Obfuscation,” IEEE Trans. on Consum. Electron., Vol. 63, no. 4, pp. 467 – 476, Nov. 2017.

6. Y. Lao and K. K. Parhi, “Obfuscating DSP Circuits via High-Level Transformations,” IEEE Trans. on Very Large Scale Integr. (VLSI) Syst., vol. 23, no. 5, pp. 819–830, May 2015.

7. A. Sengupta and D. Roy, “Protecting an intellectual property core during architectural synthesis using high-level transformation based obfuscation,” IET Electronics Letters, Vol: 53, Issue: 13, pp. 849 – 851, June 2017.

9. J. Rajendran, Y. Pino, O. Sinanoglu, and R. Karri, “Security analysis of logic obfuscation,” in DAC, 2012, San Francisco, June 2012, pp. 83–89.

10. J. Zhang, “A Practical Logic Obfuscation Technique for Hardware Security,” IEEE Trans. on Very Large Scale Integr. (VLSI) Syst., vol. 24, no. 3, pp. 1193–1197, 2015.

11. 2016Matrix Technologies, Hologram Features. [Online]. Available: http://www.matrixtechnologies.in/hologram-features.html, last accessed on April 2018.