Chapter 09

Structural Transformation based Obfuscation using Pseudo Operation Mixing for Securing Data-intensive IP Cores

Anirban Sengupta, Mahendra Rathor
Computer Science and Engineering
Indian Institute of Technology Indore

The chapter describes a structural transformation based obfuscation approach using pseudo operation mixing for securing data-intensive cores or hardware accelerators. The presented approach is based on pseudo operation mixing algorithm that attains significant structural obscurity in the design to enable un-obviousness without affecting the correct functionality.

The chapter is organized as follows: Section 9.1 discusses about the introduction of the chapter; Section 9.2 describes the structural transformation based obfuscation methodology; Section 9.3 presents POM-SO tool that is capable of performing pseudo operation mixing based structural obfuscation; Section 9.4 presents analysis of case studies in terms of security and design cost, especially focusing on DSP hardware accelerators; Section 9.5 presents conclusion; Section 9.6 provides some exercise and question for readers.

9.1. Introduction

Digital signal processing (DSP) algorithms such a discrete wavelet transform (DWT) and finite impulse response (FIR) filters etc. are highly computational or data intensive (Schneiderman, 2010; Sengupta, 2020). Therefore, it is highly efficient to integrate such data-intensive intellectual property (IP) cores as hardware accelerators in a system-on-chip (SoC). However, participation of offshore foundries in the very large scale integration (VLSI) design process makes the DSP IP cores or SoC designs vulnerable to Trojan insertion threat posed by an adversary present in an untrusted foundry (Zhang and M. Tehranipoor, 2011; Sengupta, 2016; Sengupta, 2017; Sengupta and. Mohanty 2019; Sengupta *et al*., 2017; Sengupta and Rathor; 2019; Chakraborty and Bhunia, 2009). In order to address this hardware threat, (Rathor and Sengupta, 2020) proposed a structural obfuscation methodology which is based on structural transformation of the design during high level synthesis (HLS) process. The structural transformation is performed by mixing pseudo operations during the HLS design process. Mixing of pseudo operations in the design misguides/deludes a potential attacker who targets to reverse engineer the design to understand its true functionality and structure in order to insert Trojan. Thus pseudo operations mixing based structural obfuscation approach (Rathor and Sengupta, 2020)

prevents Trojan horse insertion attack by making the RE arduous, hence ensuring trust in hardware. The pseudo operations mixing based structural obfuscation can widely be applied for all kind of DSP IP cores, regardless of the nature of the DSP application (i.e. operation count and data dependency of operations).

9.2. Structural Transformation based Obfuscation Methodology

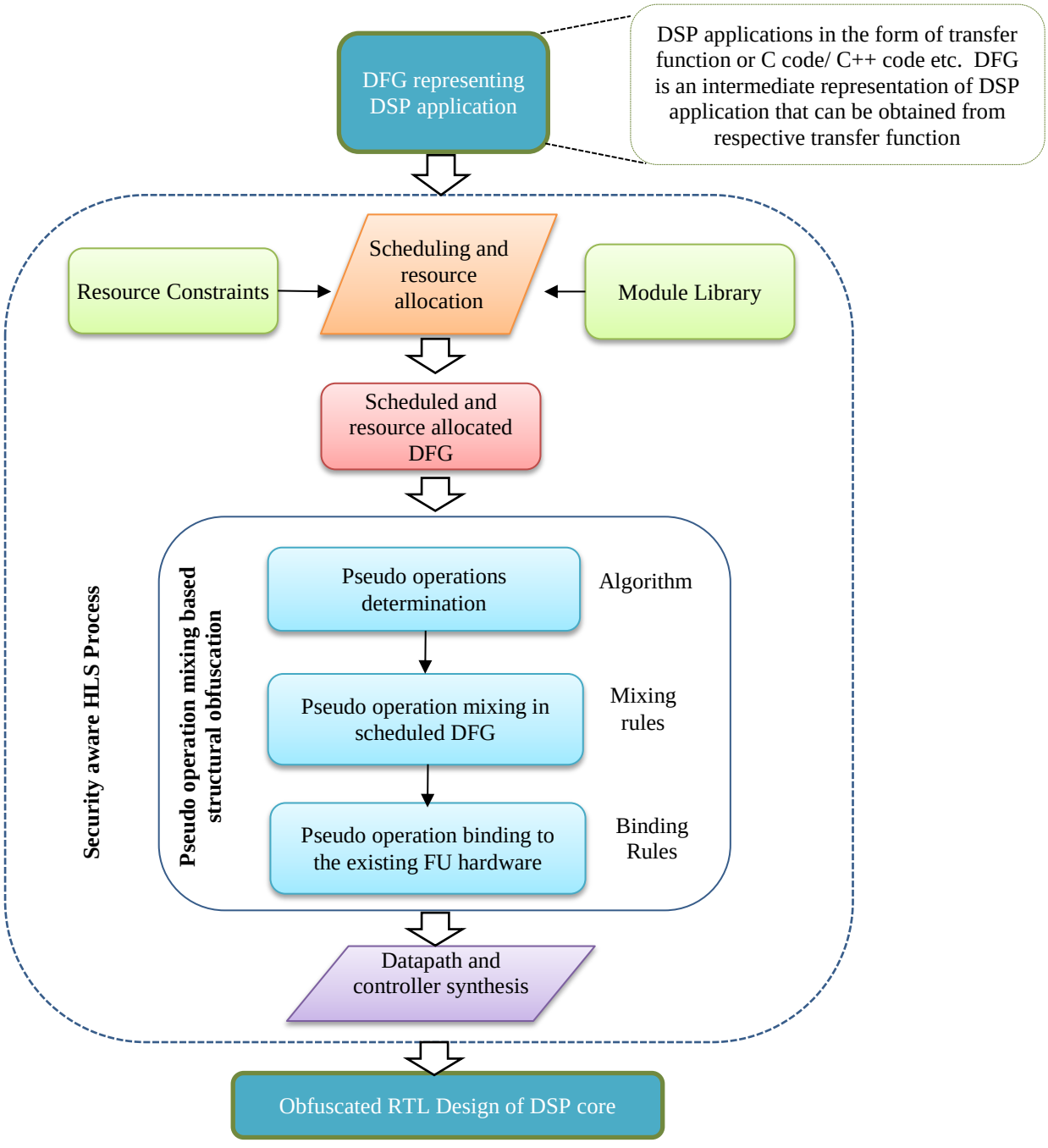

Fig. 1. *Structural transformation based obfuscation approach for securing data-intensive DSP core (Rathor and Sengupta, 2020)*

To ensure security of data-intensive DSP cores against hardware Trojan threat, the structural transformation technique discussed in this chapter is based on mixing of pseudo operations in the intended DSP application during HLS process. Let's discuss the high level perspective followed by in-depth discussion of pseudo operations mixing based structural obfuscation approach (Rathor and Sengupta, 2020).

1. High Level Perspective

Fig. 1 shows the pseudo operations mixing based structural obfuscation approach to generate a structurally obfuscated design of DSP cores. As shown in the figure, the security aware HLS process using pseudo operations mixing based structural obfuscation takes in data flow graph (DFG) of a target DSP application as input and produces a secured register transfer level (RTL) design at output. Further, scheduled and resource allocated DFG is constructed based on resource constraints and module library. Thus obtained scheduled DFG is fed as an input to the pseudo operations mixing based structural obfuscation approach (Rathor and

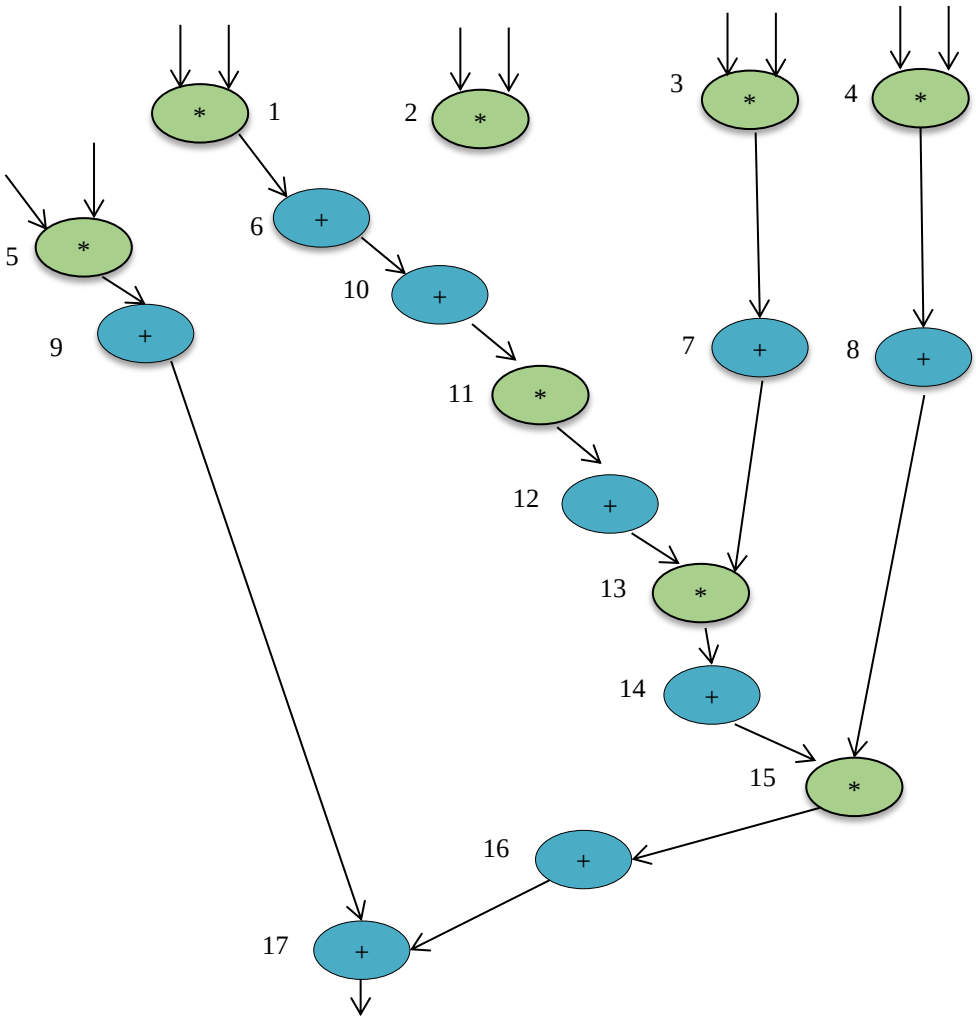

Fig. 2. *DFG of DWT application*

Sengupta, 2020). The structural obfuscation approach executes following steps sequentially in order to produce a structurally obfuscate design: (a) determination of fake or pseudo nodes/operations to be mixed in the scheduled and resource allocated DFG of DSP application (b) insertion of pseudo nodes/operations into the scheduled and resource allocated DFG based on mixing rules (c) binding of pseudo/fake operations (multiplications, additions etc.) to the existing functional unit (multipliers, adders etc.) resources of respective type based on binding rules. Once pseudo operations are mixed and allocated to the existing functional unit (FU) resources, the modified (structurally transformed) scheduled and resource allocated DFG of the DSP application is subjected to the datapath and controller synthesis phase of HLS process. This results into a structurally obfuscated RTL design of a DSP hardware accelerator.

2. Pseudo Operations Mixing based Structural Obfuscation

In-depth discussion of pseudo operations mixing based structural obfuscation approach for securing data-intensive DSP cores is

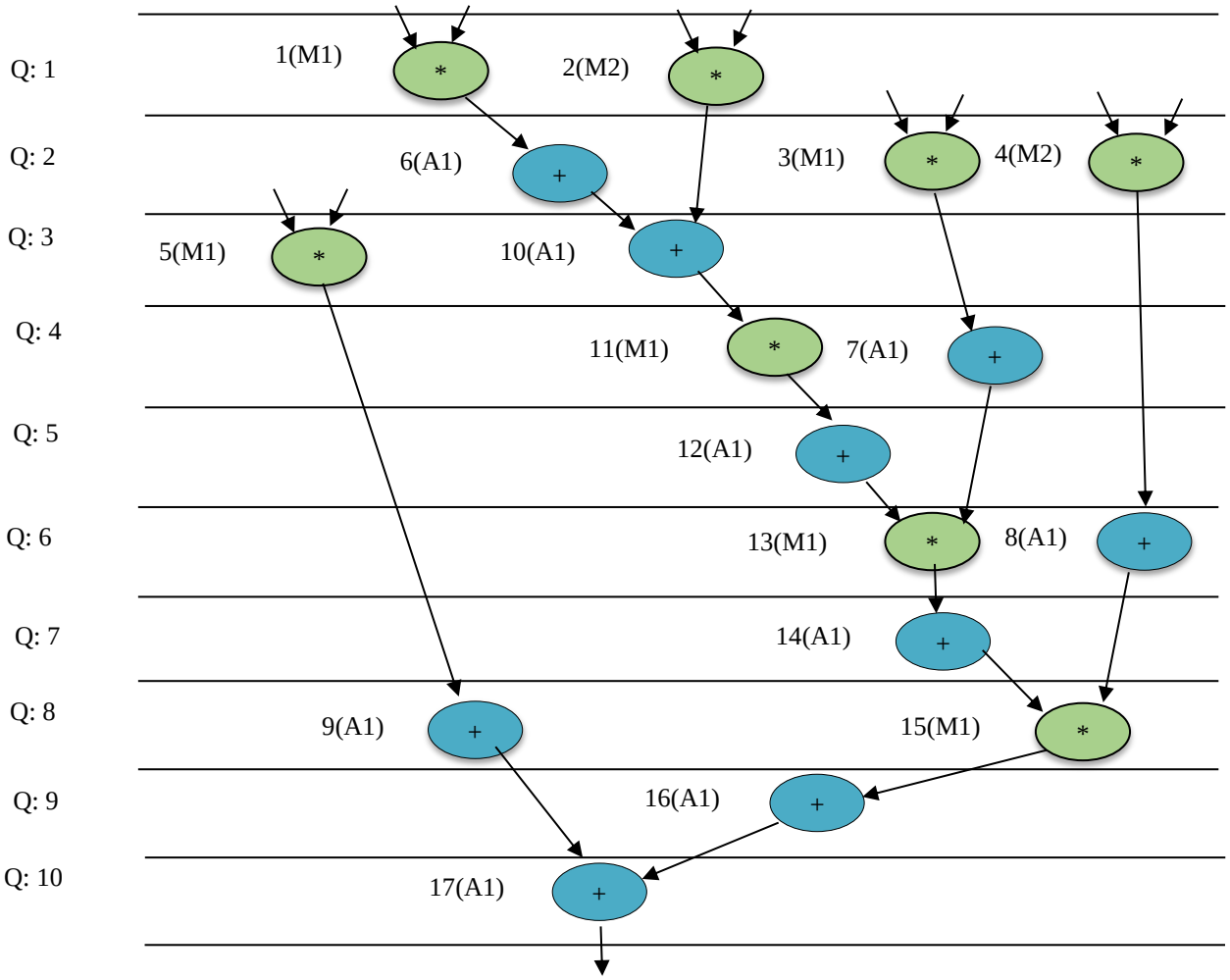

Fig. 3. *Scheduled and resource allocated DFG of DWT application based on 2(*) and 1(+)*

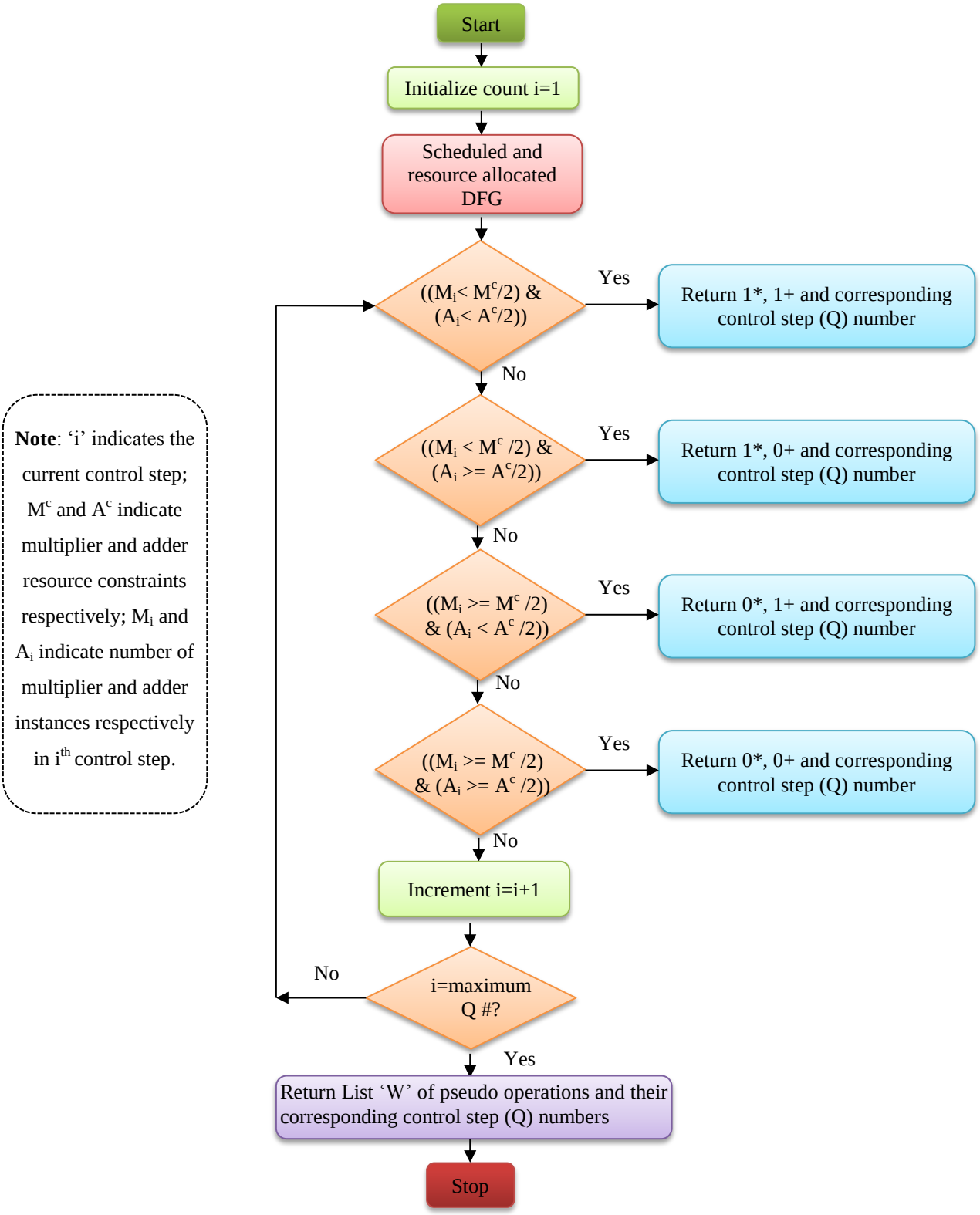

Fig. 4. *Flow chart of determining pseudo nodes/operations to be mixed in scheduled DFG (Rathor and Sengupta, 2020)*

presented in this sub-section (Rathor and Sengupta, 2020). Additionally, the pseudo operations mixing based structural

obfuscation approach has been demonstrated on discrete wavelet transform (DWT) application. The process of designing structurally obfuscated DWT core using the structural obfuscation approach starts with its DFG representation shown in Fig. 2. Further, the DFG of DWT is scheduled and resource allocated based on module library and resource constraints of **two multipliers (*) and one adder (+).** The scheduled and resource allocated DFG of DWT core is shown in Fig. 3. Once the scheduled and resource allocated DFG is obtained, it is subjected to pseudo operations mixing based structural obfuscation approach which is executed in following steps (Rathor and Sengupta, 2020):

(a) Determination of pseudo/fake operations

The scheduled and resource allocated DFG of intended DSP application is the input to this step. The resource constraints adopted during the scheduling and resource allocation governs the process of pseudo nodes insertion in each control step (Q) of scheduled and resource allocated DFG. While determining the pseudo nodes to be mixed in the scheduled and resource allocated DFG, it is ensured that the mixing of pseudo nodes does not result into the designer's resource constraints violation. The flow chart/ algorithm for determining pseudo nodes/operations to be mixed in the scheduled and resource allocated DFG is shown in Fig. 4. As shown in the flow chart, each control step is checked to determine whether the insertion of pseudo multiplication operation or pseudo addition operation in the control step is possible or not. The algorithm runs for all the control steps. **The insertion of pseudo operations can only be possible when all FU resources are not utilized to their maximum capacity (constraints) in each control step**. There are some control steps where either all instances of an FU resource are free or only some instances are utilized. These unutilized FU instances in a control step are leveraged for executing pseudo operation of respective type. Although multiple pseudo operations of a specific type can be inserted in a potential control step, however the algorithm presented in the flow chart ensures that maximum one addition / one multiplication is inserted. The output of this algorithm of determining pseudo

Table 1. *List 'W' of pseudo operations and corresponding control step number*

Pseudo addition operations	1 '+'	0 '+'	0 '+'	0 '+'	0 '+'	0 '+'	0 '+'	0 '+'	0 '+'	0 '+'
Pseudo multiplication operations	0 '*'	0 '*'	0 '*'	0 '*'	1 '*'	0 '*'	1 '*'	0 '*'	1 '*'	1 '*'
Control step (Q) number	1	2	3	4	5	6	7	8	9	10

operations is a list ‘W’ comprising of the pseudo operations to be mixed in the scheduled and resource allocated DFG and corresponding control step number.

Upon applying the algorithm of pseudo operations determination on scheduled and resource allocated DFG of DWT, the resultant list ‘W’ of pseudo operations and corresponding control step number is shown in Table 1.

(b) Mixing of pseudo operations into the scheduled and resource allocated DFG

The list of pseudo operations and corresponding Q number obtained from previous step is exploited to insert pseudo multiplication and additions operation in the scheduled and resource allocated DFG. The mixing of pseudo operations with original operations of scheduled and resource allocated DFG is performed based on following rules (Rathor and Sengupta, 2020):

(i) Pseudo operations corresponding to the first control step in the list W use any inputs of those original operations

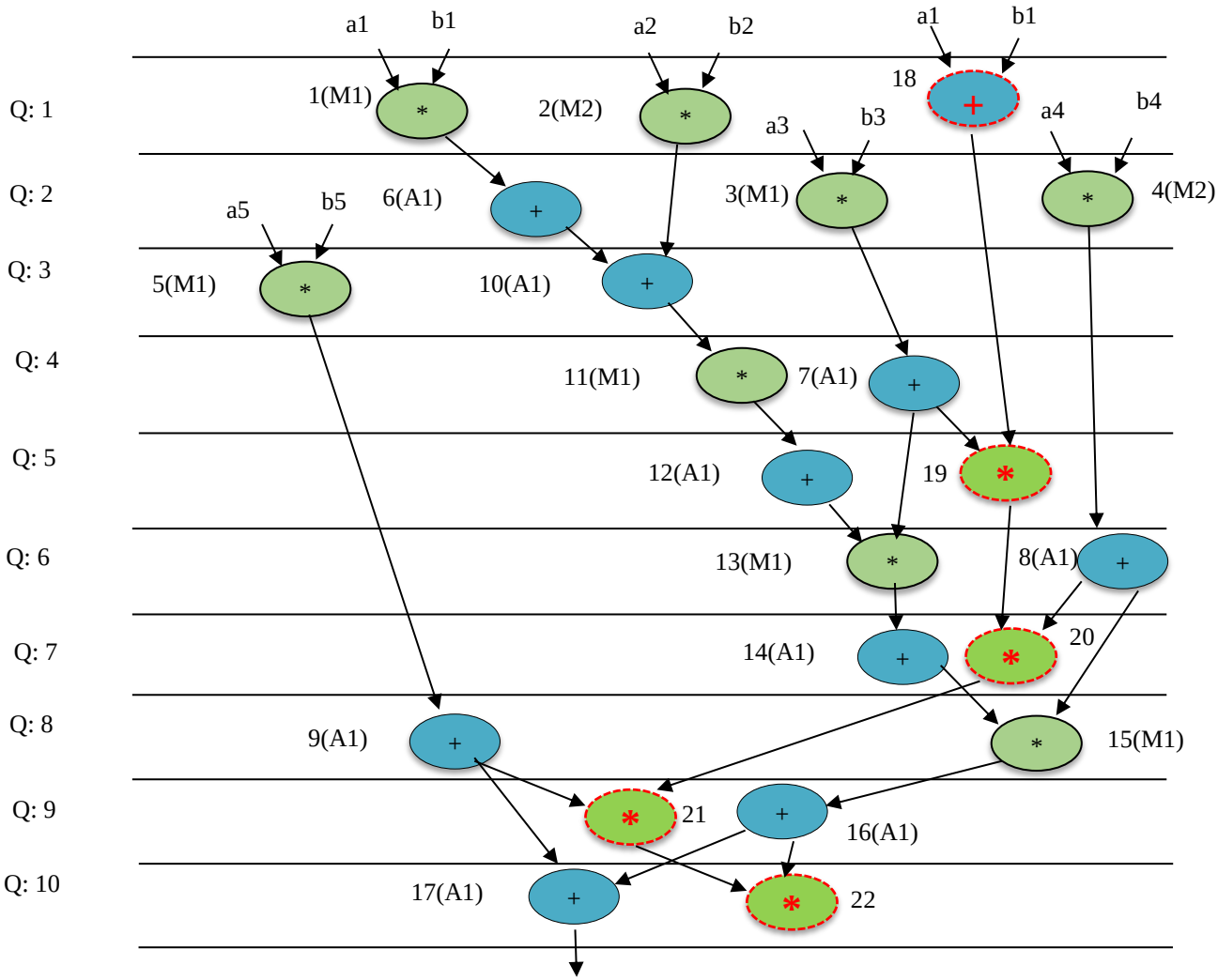

Fig. 5. *Scheduled and resource allocated DFG of DWT application post mixing pseudo operations*

which do not have predecessor operations (i.e. primary inputs).

(ii) Pseudo operations corresponding to the remaining control step in the list W uses one input from the pseudo operation and another input from any original operation located in the preceding control step.

Based on the above mentioned rules, the mixing of pseudo addition operations and pseudo multiplication operations (highlighted in Table 1) in the scheduled and resource allocated DFG of DWT core is shown in Fig. 5. As shown in the figure, operation number 18, 19, 20, 21 and 22 (highlighted in red) are the pseudo operations which have been mixed among the original operations. The mixing has been performed in such a manner that the adversary cannot distinguish the gates corresponding to the pseudo operations amongst gates corresponding to the original operations during reverse engineering of the design.

(c) Binding of pseudo operations to the existing functional unit (FU) resources

So far, we have seen how the pseudo operations are mixed into scheduled and resource allocated DFG. This step shows how the

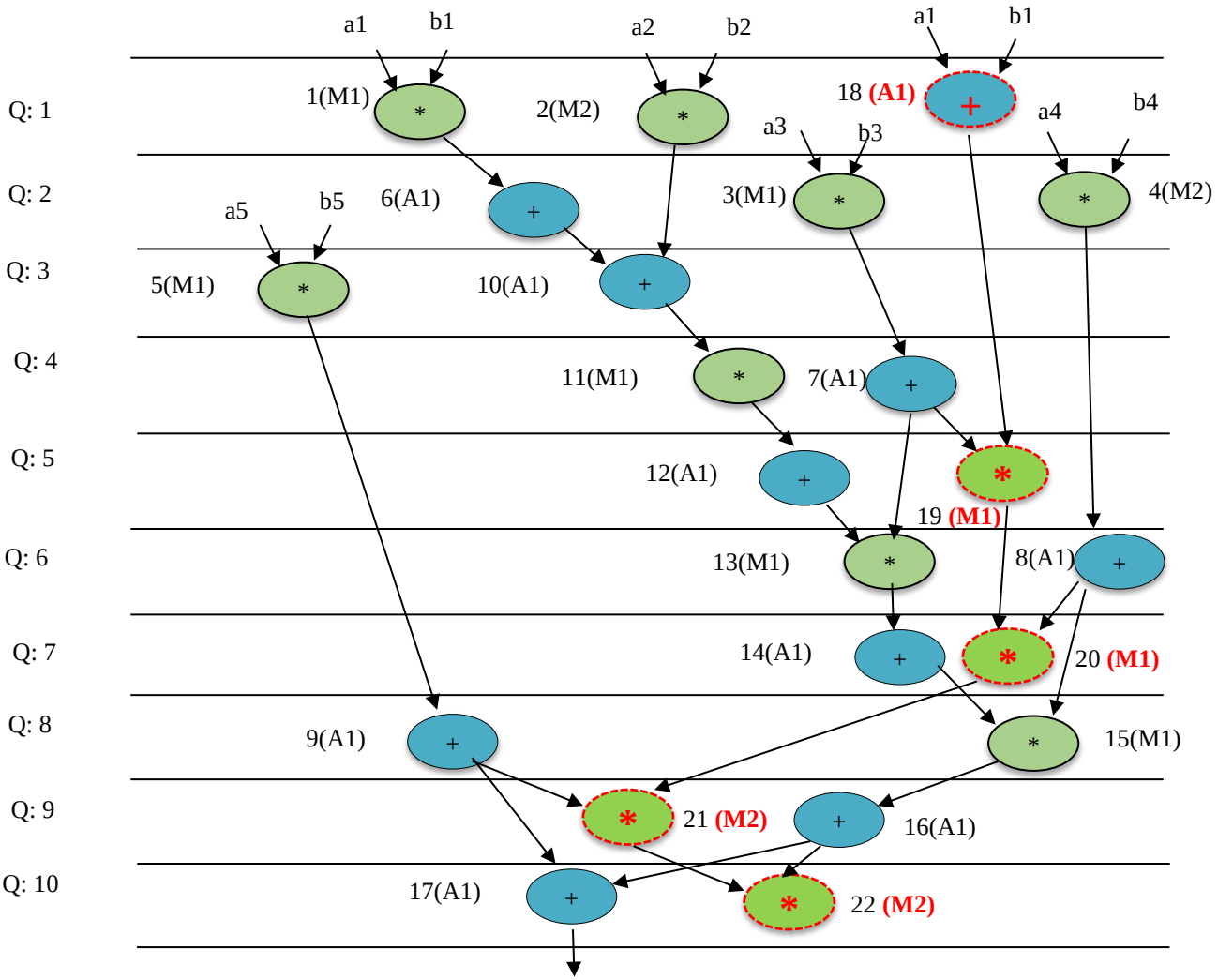

Fig. 6. *Structurally transformed scheduled and resource allocated DFG of DWT application post mixing and binding of pseudo operations*

binding of pseudo operations, to the already available FUs of respective type (multiplier/adder) in scheduled and resource allocated DFG, is performed. Here, it needs to ensure that the binding of pseudo operations to the available FUs of respective type should lead to minimal interconnect hardware overhead post-synthesizing the RTL datapath. The overhead aware binding rules of pseudo operations, to bind them with the already available FUs of respective type, are as follows (Rathor and Sengupta, 2020):

(i) The corresponding multiplexer/demultiplexer size for all instances of each FU resource type (used for FU sharing while synthesizing RTL datapath) is determined in advance.

(ii) A pseudo operation is assigned to that instance of corresponding FU type whose respective multiplexer/demultiplexer has maximum number of free (unused) inputs/outputs.

(iii) If un-used inputs/output are not available in the multiplexer/demultiplexer of corresponding FU type, then the pseudo operation is assigned to that instance of corresponding FU type which has been exploited least number of times in the scheduled and resource allocated DFG.

Based on the above mentioned rules, the binding of pseudo addition operations and pseudo multiplication operations, to the already available adder and multiplier resources respectively in the scheduled and resource allocated DFG of DWT is performed. Let's see the binding of pseudo multiplication operations based on binding rules. While binding pseudo multiplication operation # 19 in control step Q5, the available choices are multiplier M1 and M2. Before binding, the number of instances and required multiplexer size for M1 and M2 are determined, as highlighted in the blue column of Table 2. As shown in the table, the required multiplexer size for M1 is 8:1. However, only six inputs are currently in use. Two inputs are still free to offer two times more sharing of multiplier resource M1. In contrast, the required multiplexer size for M2 is 2:1 and both inputs are engaged. Therefore according to the binding rule, the pseudo multiplication operation # 19 is assigned to multiplier M1. Similarly, pseudo

Table 2. *Multiplexer size determination pre and post performing pseudo operations mixing based structural obfuscation*

FU resources	**Before mixing and binding pseudo operations**		**Post mixing and binding pseudo operations**	
	Number of instances	Required multiplexer size	Number of instances	Required multiplexer size
Adder (A1)	9	16:1	10	16:1
Multiplier (M1)	6	8:1	8	8:1
Multiplier (M2)	2	2:1	4	4:1

multiplication operation # 20 is also assigned to multiplier M1 because of availability of un-used input in the corresponding Mux. Once binding of pseudo multiplication operations #19 and # 20 is accomplished, the corresponding multiplexer of multiplier M1 is not left with any unused input. And, the corresponding multiplexer of multiplier M2 has also not unused inputs. Therefore the next pseudo multiplication operation # 21 is assigned to that multiplier instance which has been used least number of times (minimum multiplexer size) in the scheduled and allocated DFG. Therefore multiplication operation # 21 is assigned to multiplier

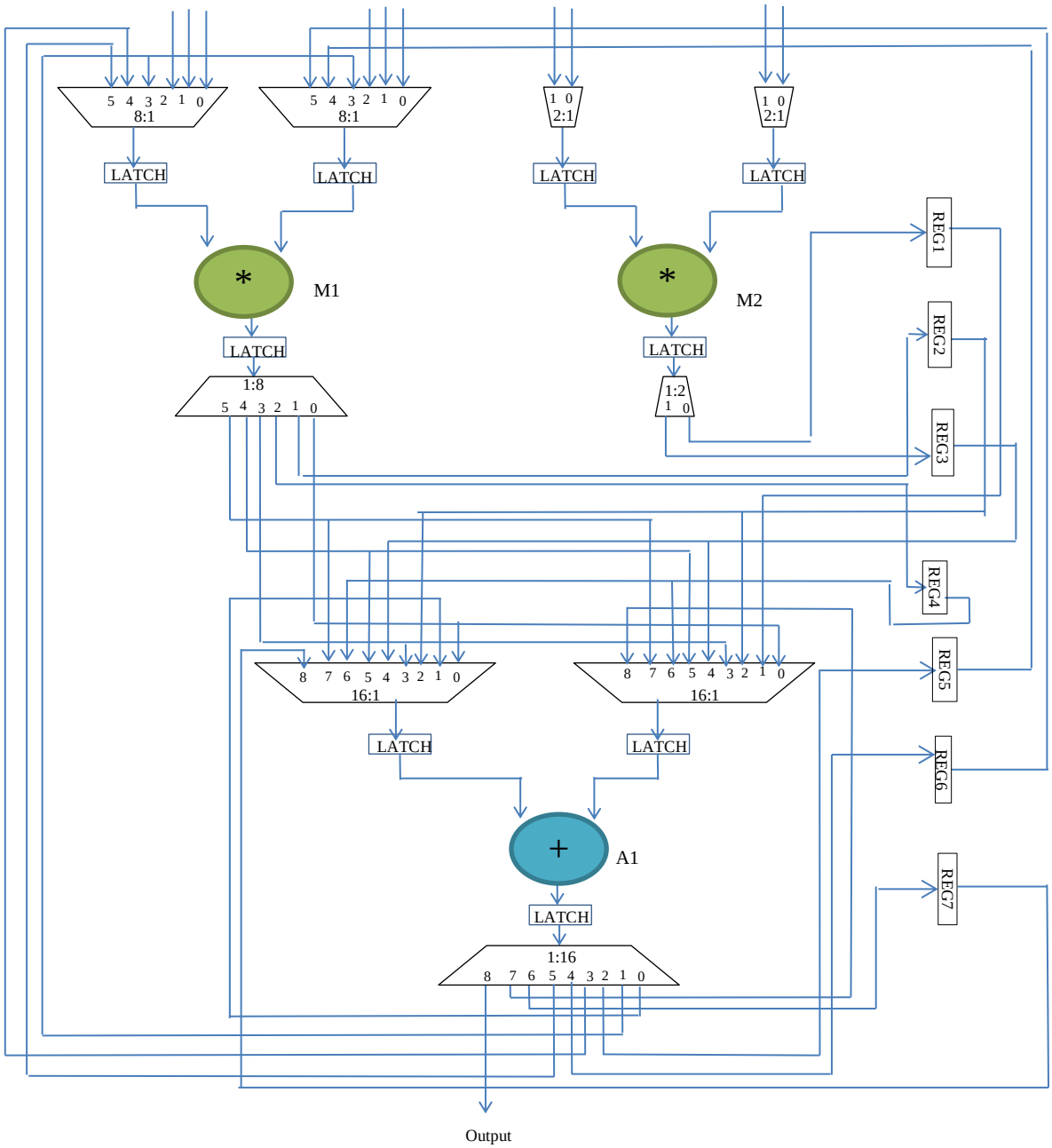

Fig. 7. *RTL datapath before structural transformation based obfuscation*

M2 according to the binding rules. Similarly, binding of remaining pseudo operations is performed. Post-binding of pseudo operations with the respective existing FU resources, the structurally transformed scheduled and resource allocated DFG of DWT is shown in Fig. 6. Table 2 highlights the number of instances and required multiplexer size for FU resources, pre and post performing pseudo operations mixing based structural obfuscation. As shown in the table, binding of pseudo operations to the resource M1 and A1 does not increase their respective multiplexer size. However, binding of pseudo operations to the resource

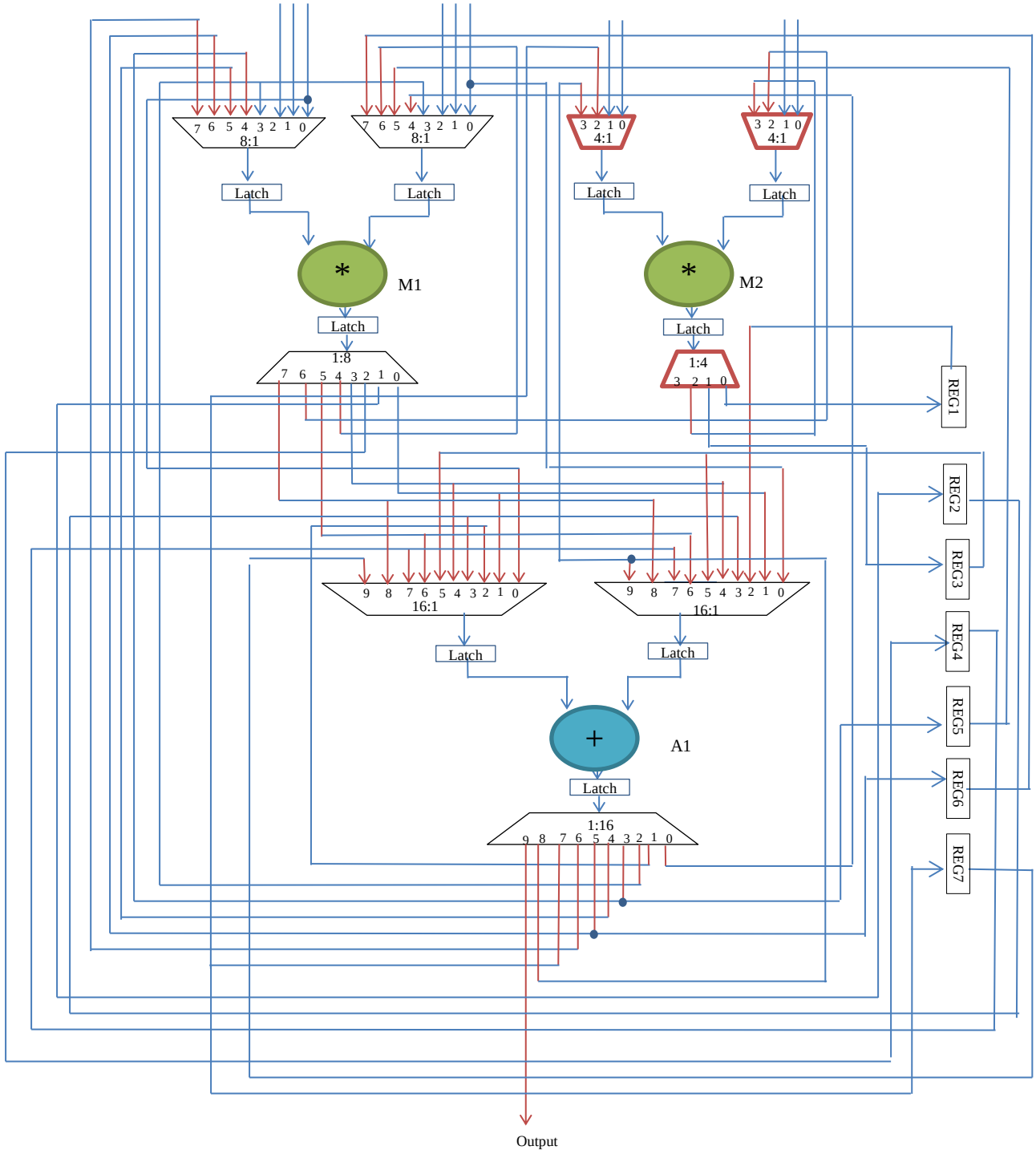

Fig. 8. *Structurally obfuscated RTL datapath using pseudo operations mixing based structural transformation*

M2 increases its size from 2:1 to 4:1. This result into a slight design cost overhead due to extra interconnect hardware. However in many cases, the available interconnect hardware is capable to associate pseudo operations with the existing respective FU resource without incurring interconnect hardware overhead (or without increasing multiplexer size). Hence, this approach offers the structural obfuscation security at minimal area overhead.

RTL Datapath synthesis: The RTL datapath post performing pseudo operations mixing based structural obfuscation is synthesised from structurally transformed scheduled and allocated DFG. For the sake of comparison, the RTL datapath of DWT application pre and post-structural obfuscation are shown in Fig. 7 and Fig. 8 respectively. The unsecured RTL datapath (pre-obfuscation) is obtained from scheduled and allocated DFG shown in Fig. 3. The secured (structurally obfuscated) RTL datapath is obtained from structurally transformed scheduled and allocated DFG, as shown in Fig. 6. In the structurally obfuscated RTL datapath shown in Fig. 8, the changes due to pseudo operations mixing based obfuscation are highlighted in red colour.

9.3. Pseudo Operations Mixing based Structural Obfuscation Tool

Authors have developed a ***POM-SO tool*** (pseudo operation mixing based structural obfuscation tool) to simulate and analyse the

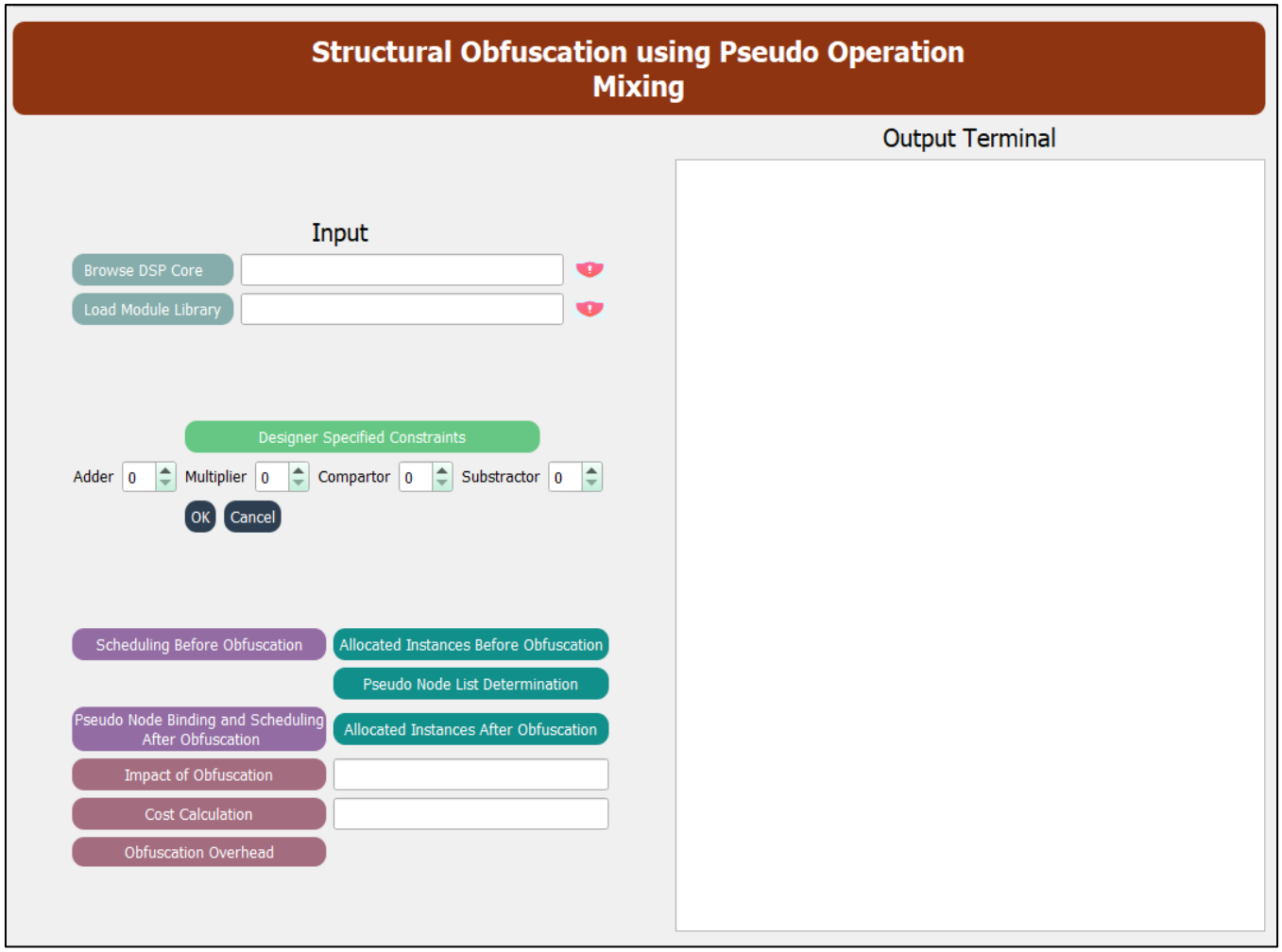

Fig. 9. *Snapshot of GUI of POM-SO tool*

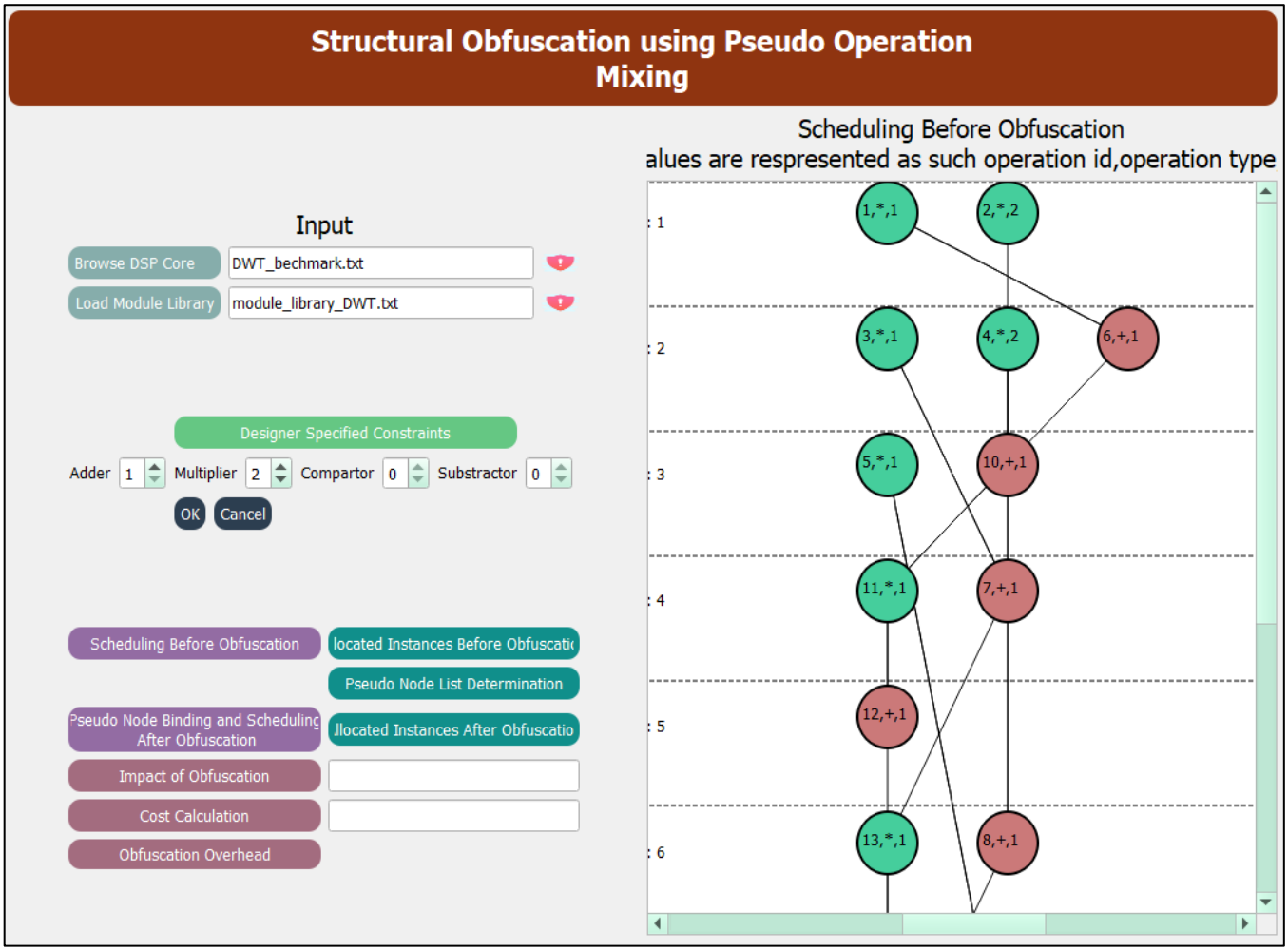

Fig. 10. *Snapshot of scheduled and resource allocated DWT application*

pseudo operation mixing based obfuscation approach for securing DSP hardware accelerators. This tool provides a friendly graphical interface to users. A snapshot of the graphical user interface (GUI) of the tool is shown in Fig. 9. The left portion of the tool shows the panel for providing required inputs to the tool and right portion shows the panel to see the intermediate and final outputs of the pseudo operation mixing based structural obfuscation approach. The POM-SO tool accepts the DSP application input in the form CDFG along with module library and resource constraints. The tool shows intermediate steps of pseudo operation mixing and finally generated structurally transformed scheduled and resource allocated DFG at the output.

Let's generate all the intermediate and final output of the pseudo operation mixing based structural obfuscation approach for DWT core using the POM-SO tool. We will provide the same inputs used during the demonstration of DWT core discussed in section 9.2. Here, we can match the output generated with the tool and that obtained in the demonstration. First of all, input DFG of DWT core, resource constraints of 1 adder and 2 multipliers and module library are fed to the tool as shown in Fig. 10. Upon clicking on the button "Scheduling Before Obfuscation", the scheduled and resource allocated DFG becomes available on to the

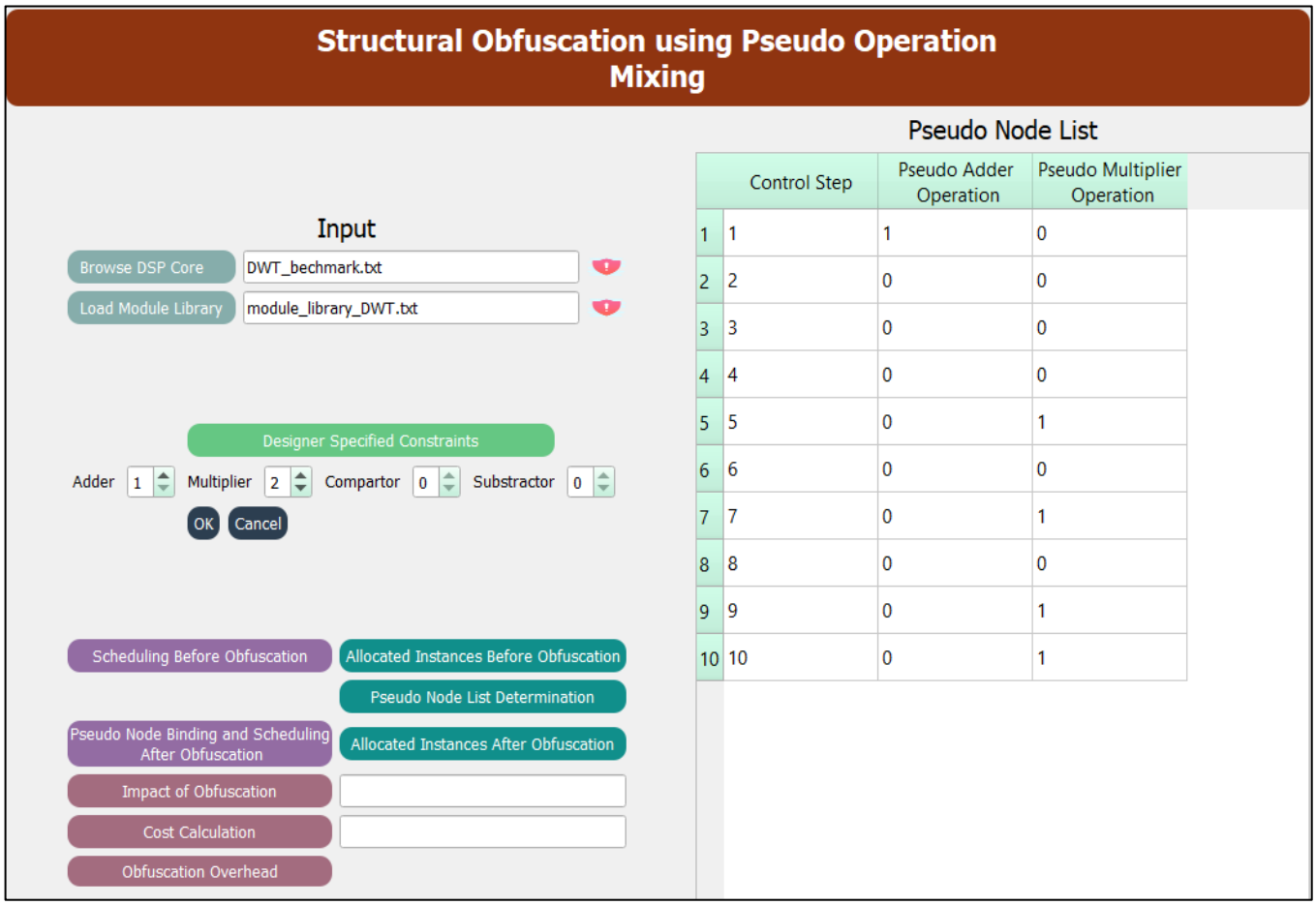

Fig. 11. *Snapshot post determining the list of pseudo operations*

output terminal. Here, only an excerpt of the graph (upto six control steps) is shown in Fig. 10. Further, upon clicking on the button "Pseudo Node List Determination", the list of pseudo operations to be mixed and corresponding control step number becomes available onto the output terminal as shown in Fig. 11. The structurally transformed scheduled and resource allocated DFG post mixing pseudo operations can also be seen at output terminal by clicking on the respective button. Fig. 12 shows the structurally transformed scheduled and resource allocated DFG of DWT core.

Further, Fig. 13 shows the number of instances of FU resources pre and post performing pseudo operations mixing based structural obfuscation. The tool produces the desired outputs that match with the demonstration on DWT core discussed in section 9.2. This tool is useful for case studies of various kind of DSP hardware accelerator applications such as finite impulse response (FIR) filter, infinite impulse response (IIR) filter, discrete wavelet transform (DWT) etc. In addition, the tool evaluates and shows the strength of obfuscation and design cost overhead post performing the structural obfuscation.

9.4. Analysis on Case Studies

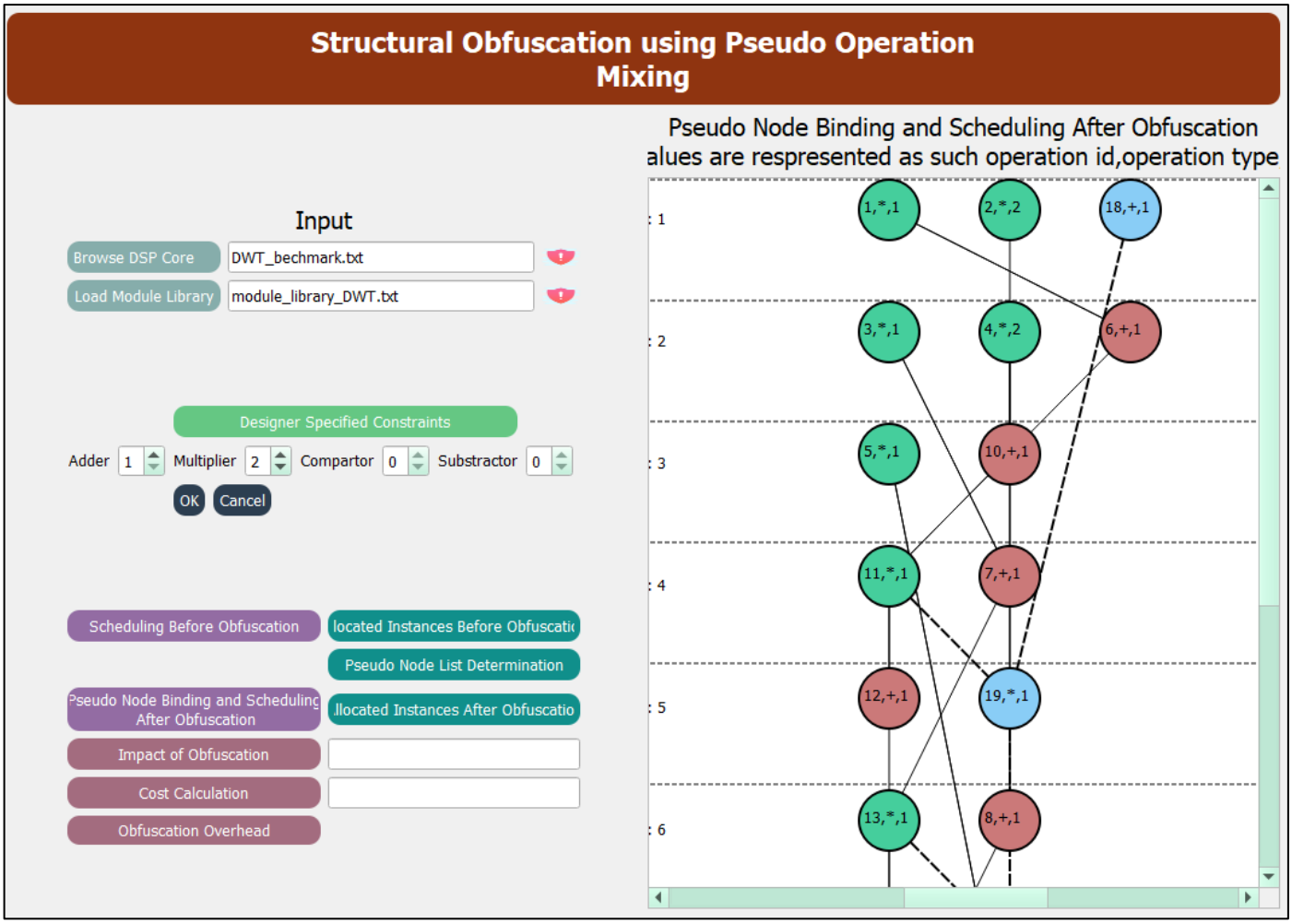

Fig. 12. *Snapshot of structurally transformed scheduled and resource allocated DWT application*

This section analyses the security due to pseudo operations mixing based structural obfuscation and its impact on design cost. The cast study in terms of security and design cost analysis has been performed on various DSP applications. Further, the security due to pseudo operations mixing based structural obfuscation (Rathor and Sengupta, 2020) has been compared with a contemporary structural obfuscation approach (Sengupta and Rathor, 2019).

*1. **Security Analysis** (Rathor and Sengupta, 2020)*

The pseudo operation mixing based structural obfuscation approach deludes an adversary using pseudo operations mixed into the design, therefore renders the design architecture arduous to be reverse engineered by the adversary. Thus the pseudo operation mixing based obfuscation approach is very useful in preventing Trojan insertion by the adversary. The RTL datapath of DSP applications is affected due to pseudo operation mixing based structural obfuscation in terms of more number of times sharing of available FU resources, among original and pseudo operations, using multiplexers and demultiplexers. Further, the mixing of

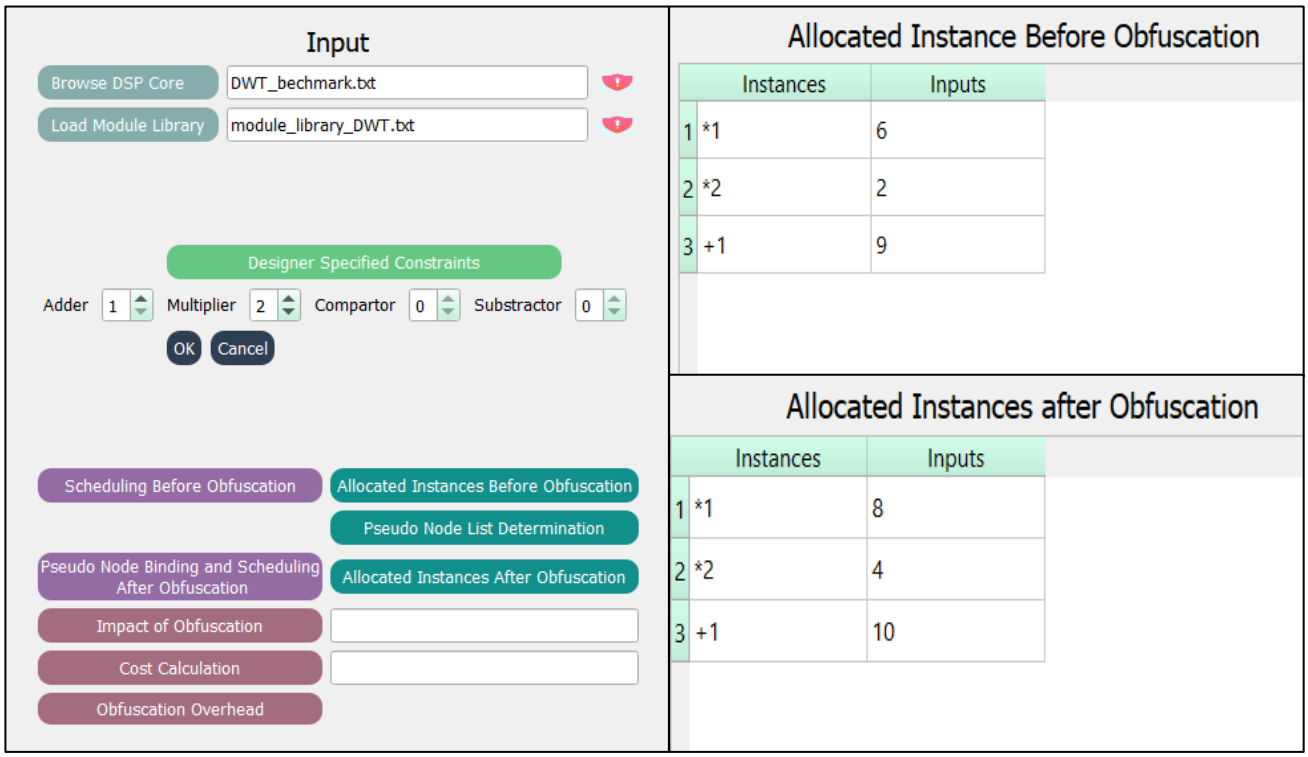

Fig. 13. *Snapshot showing impact on resource instances post structural obfuscation*

pseudo operations hugely obscures the gate level netlist obtained post logic synthesis. This is because, the mixing of pseudo operations affects the large percentage of gates. Hence the % gate count affected due to obfuscation is the measure of strength of structural obfuscation. The formula for evaluating strength of structural obfuscation (SOB) in terms of % gate count affected is as follows:

$$\%SOB = \frac{AG}{BG} \times 100 \qquad (1)$$

Where, AG indicates total affected gate count (with respect to baseline) post structural obfuscation and BG indicates total gate count of baseline (un-obfuscated) design. Where, total affected gate count (AG) is calculated as follows:

$$AG = G^{AR} + G^{C} \qquad (2)$$

Where, G^{AR} indicates gate count of affected resources (such as affected multiplexers and demultiplexers) post obfuscation and G^{C} indicates change in gate count post obfuscation (i.e. difference of gate count of baseline and obfuscated design). Table 3 shows the gate count of baseline design, total affected gate count (calculated using (2)) due to obfuscation and strength of obfuscation (calculated using (1)) in terms of % gates affected with respect to baseline. As evident from the table, high value of strength of obfuscation is achieved using pseudo operations mixing based structural obfuscation. Further, the strength of obfuscation using pseudo operations mixing based structural obfuscation (Rathor and Sengupta, 2020) has been analysed in terms of comparison with the contemporary approach (Sengupta and Rathor, 2019) as shown in Table 3. Since the contemporary approach (Sengupta and

Table 3. *Security analysis in terms of strength of structural obfuscation and comparison with (Sengupta and Rathor, 2019)*

DSP applications	IIR	Mesa Horner	JPEG	MPEG	DWT
Gate Count (baseline)	3648	4192	29856	8272	6288
Affected Gate Count (Rathor and Sengupta, 2020)	3168	3168	8256	4752	5472
Strength of obfuscation (Rathor and Sengupta, 2020)	86.84%	75.57%	27.65%	57.44%	93.89%
Strength of obfuscation (Sengupta and Rathor, 2019)	26.3%	NA	NA	NA	NA

Note: NA indicates that the obfuscation approach is "not applicable"

Rathor, 2019) is based on integration of two RTL datapath of such DSP applications which have some similarity in their algorithmic description, therefore it is applicable to limited number of DSP applications. Thus it is not widely applicable obfuscation approach. However, the pseudo operations mixing based structural obfuscation (Rathor and Sengupta, 2020) can be applied extensively.

2. Design Cost Analysis (Rathor and Sengupta, 2020)

This sub-section discusses the impact of employing pseudo operation mixing based structural obfuscation on design cost. Following equation is used to evaluate the design cost:

$$C_d(U_i) = \rho_1 \frac{L_d}{L_m} + \rho_2 \frac{A_d}{A_m} \qquad (3)$$

Where, $C_d(U_i)$ is the design cost towards resource constraints U_i, further L_d and L_m are the design latency at specified resource constraints and maximum design latency respectively, A_d and A_m are the design area at specified resource constraints and maximum area respectively and ρ_1, ρ_2 are the weights which are fixed at 0.5.

The design cost analysis of pseudo operation mixing based obfuscation approach has been performed in terms of design cost comparison with the baseline/un-obfuscated version. Fig. 14 shows the design cost analysis. It is observed from the figure that the design cost overhead due to pseudo operation mixing is zero for most of the DSP application. The underlying reason is that the binding of pseudo operations with the available respective FU resources does not result into additional interconnect-hardware

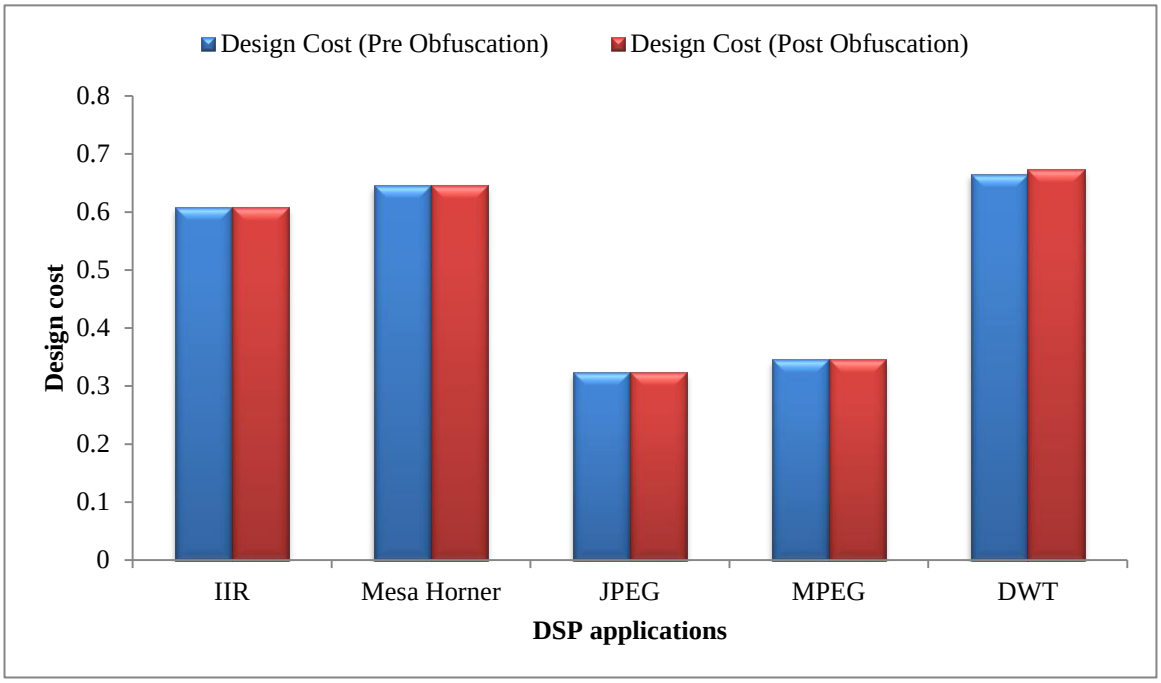

Fig. 14. *Design cost analysis with respect to baseline or un-obfuscated version (Rathor and Sengupta, 2020)*

(multiplexers and demultiplexers). However in some cases, the size of multiplexers and demultiplexers may need to be augmented to accommodate pseudo operations. Nonetheless, the binding rules discussed in this chapter aims to minimize the interconnect-hardware overhead. Hence the pseudo operation mixing based obfuscation approach leads to minimal design cost overhead.

9.5. Conclusion

Employing security during the design process of data-intensive IP cores is required to ensure trust in hardware. This chapter discussed a pseudo operations mixing based structural transformation approach which obfuscates the designs of data-intensive DSP cores in order to prevent against Trojan (malicious logic) insertion threat. The robustness of this approach lies in the facts that the applicability of pseudo operations mixing based structural transformation approach has extensive coverage of target DSP applications, regardless of the nature of the application. Additionally, the pseudo operations mixing based structural obfuscation approach provides high security at minimal overhead.

At the end of this chapter, following concepts are communicated to the readers:

- Algorithm of determining pseudo operations to be inserted in a scheduled and resource allocated DFG
- Mixing rules of pseudo operations into the scheduled and resource allocated DFG
- Binding rules of pseudo operations with the existing FU resources in the scheduled and resource allocated DFG
- Demonstration of pseudo operations mixing based structural obfuscation on DWT core
- Case studies in terms of security and design cost analysis

9.6. Questions and Exercise

1. What is pseudo operation determination algorithm?
2. Describe the security aware HLS follow.
3. State the algorithm of pseudo operation mixing in scheduled DFG.
4. State the binding rules of pseudo operation binding.
5. What is the role of list ‘W’ of pseudo operations?
6. How does pseudo operation mixing algorithm achieve structural obfuscation?
7. Perform pseudo operation mixing algorithm for DCT core to achieve structural obfuscation.
8. How is the multiplexer size determined after resource sharing?
9. What is the input/output of POM-SO tool used for structural obfuscation?
10. What is strength of structural obfuscation?
11. How is the design cost analysed for a structurally obfuscated design?

References

A. Sengupta (2017), 'Hardware Security of CE Devices [Hardware Matters],'*IEEE Consumer Electronics Mag,* vol. 6(1), pp. 130-133.

A. Sengupta (2016), 'Intellectual Property Cores: Protection designs for CE products,' *IEEE Consumer Electronics Mag*, vol. 5, no. 1, pp. 83-88.

A. Sengupta, S. P. Mohanty (2019), 'IP core and integrated circuit protection using robust watermarking', *Book: IP Core Protection and Hardware-Assisted Security for Consumer Electronics*, e-ISBN: 9781785618000, pp. 123-170.

R. Schneiderman (2010), 'DSPs evolving in consumer electronics applications, '*IEEE Signal Process. Mag*., vol. 27(3), pp. 6–10.

A. Sengupta (2020), 'Frontiers in Securing IP Cores - Forensic detective control and obfuscation techniques', *The Institute of Engineering and Technology (IET)*, ISBN-10: 1-83953-031-6, ISBN-13: 978-1-83953-031-9.

X. Zhang and M. Tehranipoor (2011), 'Case study: Detecting hardware Trojans in third-party digital IP cores,' *IEEE International Symposium on Hardware-Oriented Security and Trust*, San Diego CA, pp. 67-70.

A. Sengupta, D. Roy, S. P. Mohanty and P. Corcoran (2017), "DSP design protection in CE through algorithmic transformation based structural obfuscation," *IEEE Transactions on Consumer Electronics*, vol. 63, no. 4, pp. 467-476.

M. Rathor and A. Sengupta (2020), 'Obfuscating DSP Hardware Accelerators in CE Systems Using Pseudo Operations Mixing,' axy, pp.

RS Chakraborty, and S. Bhunia (2009), 'Security against hardware Trojan through a novel application of design obfuscation,' in *proc. International Conference on Computer-Aided Design, ACM*, pp. 113-116.

A. Sengupta and M. Rathor (2019), 'Protecting DSP Kernels Using Robust Hologram-Based Obfuscation,' *IEEE Transactions on Consumer Electronics*, vol. 65, no. 1, pp. 99-108.