Module-9

Security of Functionally Obfuscated DSP cores

Dr Anirban Sengupta

Professor, CSE,

IIT Indore, India

Outline

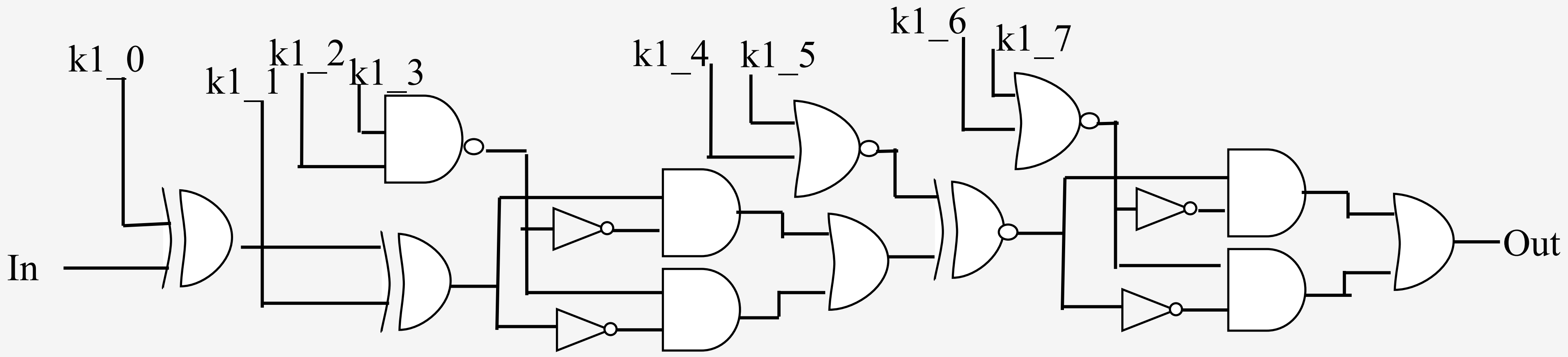
- ❑ Introduction
- ❑ Functional Obfuscation of DSP cores
- ❑ Attack scenarios and Threat models of Functional Obfuscation
- ❑ Security of Functionally Obfuscated DSP core against Removal Attack using SHA-512 based Key Encryption Hardware
- ❑ Security against Removal Attack using ARL Hardware
- ❑ Analysis on Case Studies

Introduction

- It is wide acknowledged that the IPs can be infringed/abused by attackers through different forms of threat model including reverse engineering (RE).
- An IP core once designed (till netlist), can be easily used in different process technologies to convert into a hard IP format.
- This also entices an attacker to steal the IP netlist and use it without authorization.
- Thus there need to be active form of protection needed to secure these IP cores against standard threats.
- The most common form of active protection is called functional obfuscation.

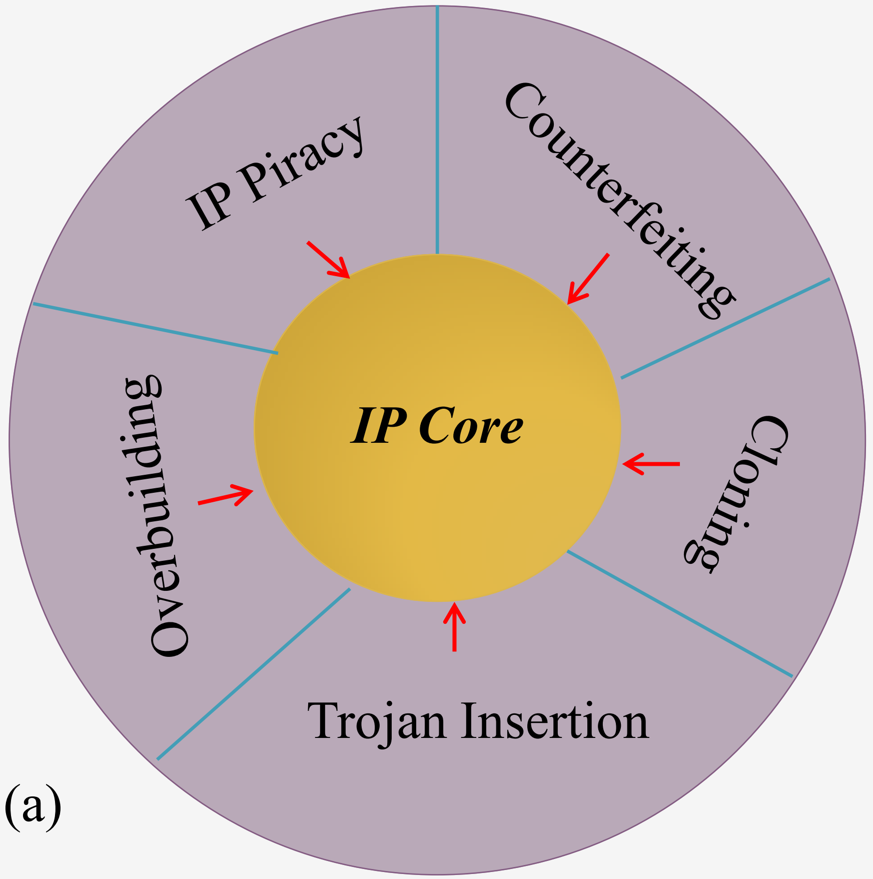
Functional Obfuscation of DSP cores

- The functional obfuscation of DSP cores is performed by inserting a certain number of IP core locking blocks (ILBs) at appropriate locations into the design .
- The number of ILBs inserted is determined using a rule based on some parameters. The structure of an ILB is constituted using a combination of basic XOR, XNOR, AND, NAND, OR, NOT gates. A sample ILB structure is shown in the Figure.
- The internal gates in an ILB are interconnected in such a fashion that the higher security through functional obfuscation could be achieved.

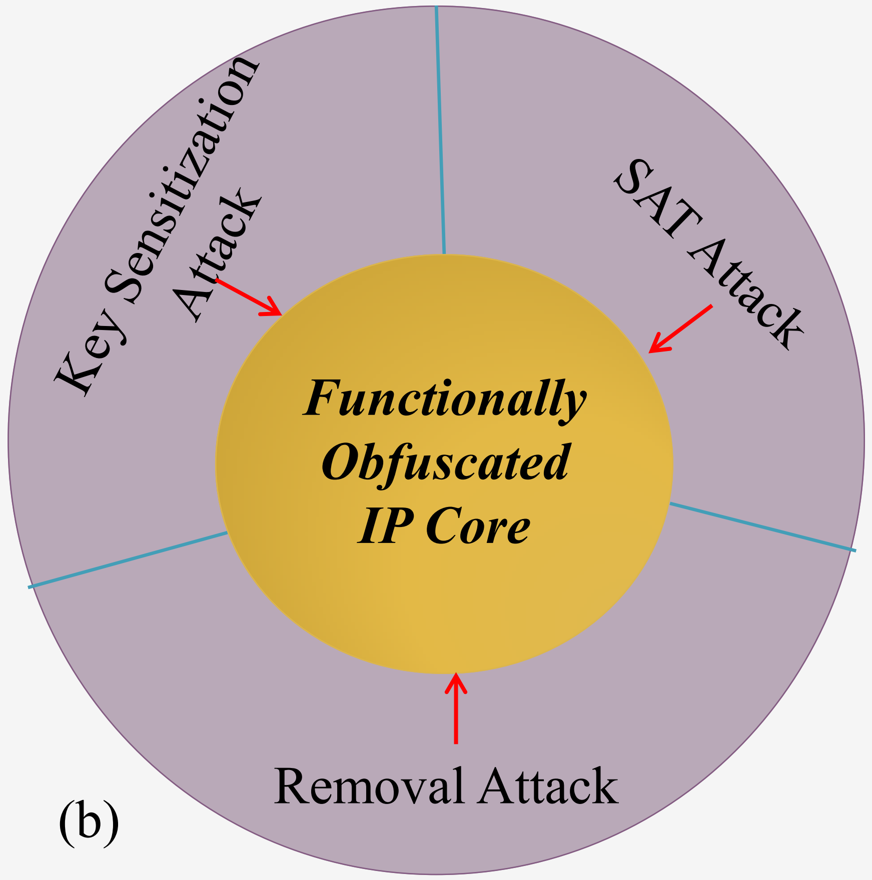


A sample ILB structure (requires an 8-bit key 'k1' to activate)

Attack Scenarios and Threat models of Functional Obfuscation



Attack scenarios for an un-obfuscated IP core



Attack scenarios for a functionally obfuscated IP core

The Vulnerability of Functional Obfuscation towards Different Attacks

❑ ***Key sensitization attack:***

- An attacker can extract keys of a locked (functionally obfuscated) netlist through key sensitization attack.
- To mount this attack, the attacker also requires a functional IC (can be obtained from open market) along with the obfuscated netlist (may be obtained through RE).

➢ **Can sensitize the key through following possibilities :**

- The attacker can sensitize a key-bit at primary output by controlling the primary inputs of the design. To do so, the attacker needs to identify the input pattern that can sensitize the correct key-bits to the primary output.
- The Attacker tries to find isolated key-gates in the obfuscated netlist. This is because the key-bits are easy to sensitize through isolated key-gates. (If a key-gate is not connected to other key gates (through any path) in the obfuscated design, then it is termed as an isolated key-gate).
- If the attacker finds a sequence of key gates in the obfuscated design, then he/she can substitute it by a single gate. This leads to reduction in key-bits. Thus, run of key-gates makes obtaining key-bits easier for an attacker.
- In the path of sensitization of a key-bit, the effect of other key-bits can be nullified by muting the relevant key-gates.

❑*SAT attack:*

- To mount this attack, an attacker needs a locked netlist and an activated/ functional IC.
- The SAT attack algorithm first generates distinguished input-output (IO) pairs using a SAT formula.
- These distinguished IO pairs are exploited to eliminate wrong key combinations. A subset of wrong key combinations can be eliminated using each distinguished IO pair.
- The SAT attack algorithm generates the distinguished IO pair iteratively as long as the elimination of all the wrong keys is accomplished

❑Removal attack:

- In this attack, an attacker is assumed to have access to the obfuscated netlist.
- The attacker attempts to eliminate the additionally inserted key-gates (IP core locking logic) by detecting those using sophisticated algorithms/tools

Security of Functionally Obfuscated DSP core against Removal Attack

- The security against removal attack can be provided by making the ILBs structurally reconfigurable
- That is the gate structures of the ILB architecture can be configured according to the values at the key bits.
- In other words, the ILB gate structure used (in the obfuscated netlist) is never fixed to make it undetectable to an attacker.
- Different ILB gate structures are used based on the reconfiguration to confuse the attacker and thwart removal attack.
- This is possible because various structures of ILBs can be generated using different combinations of same basic gates (XOR, XNOR, AND, NAND, OR, NOT gates).
- Yet, each ILB structure is capable to provide similar security strength.

Security using custom SHA-512 based ILBs-Key generation Hardware

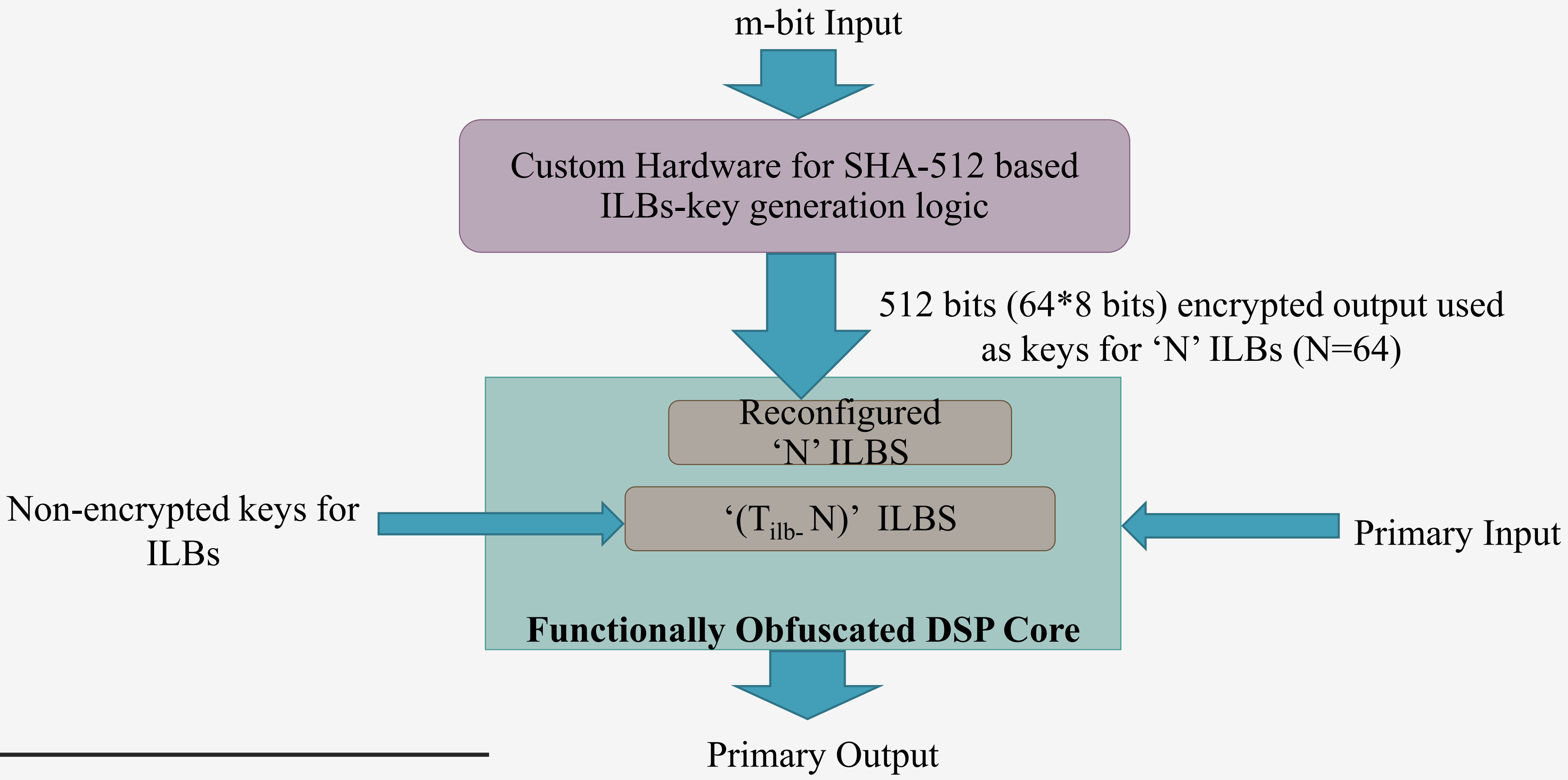
❖ Features of this approach

- The ILBs structures are reconfigured based on output of custom SHA- 512 based ILBs-key generation logic. Thus, detection of ILBs in the complex DSP netlist becomes difficult because its architecture is not known to the attacker in advance.
- Post synthesis, the ILBs gate structure change and get camouflaged (unrecognizable) the form of basic gates in the entire design netlist. Thus, the removal attack on ILBs becomes extremely difficult.
- Additionally, since authors have designed a custom SHA-512 based ILBs-key generation hardware (not publicly available), therefore post synthesis, its detection in the design netlist (comprising of DSP core and ILBs) is highly challenging.
- The crypto hash function (SHA-512) provides some strong security features such as collision resistance, uniformity , deterministic .

❖ Features of this approach

- ➢ To know the 512-bit hash-digest, an attacker needs to find 1024 bits of plaintext input of SHA-512 algorithm.
- ➢ Therefore, for an attacker to know the architecture of 64 reconfigured ILBs, 512 bits of ILBs keys are required to be decoded from 1024 bits of plaintext input of SHA-512
- ➢ To reconfigure up to 64 ILBs, one SHA-512 based key generation block needs to be integrated with a functionally obfuscated DSP design.
- ➢ It incurs considerably low area overhead than four instances of AES-128 required to structurally reconfigure the same number of ILBs

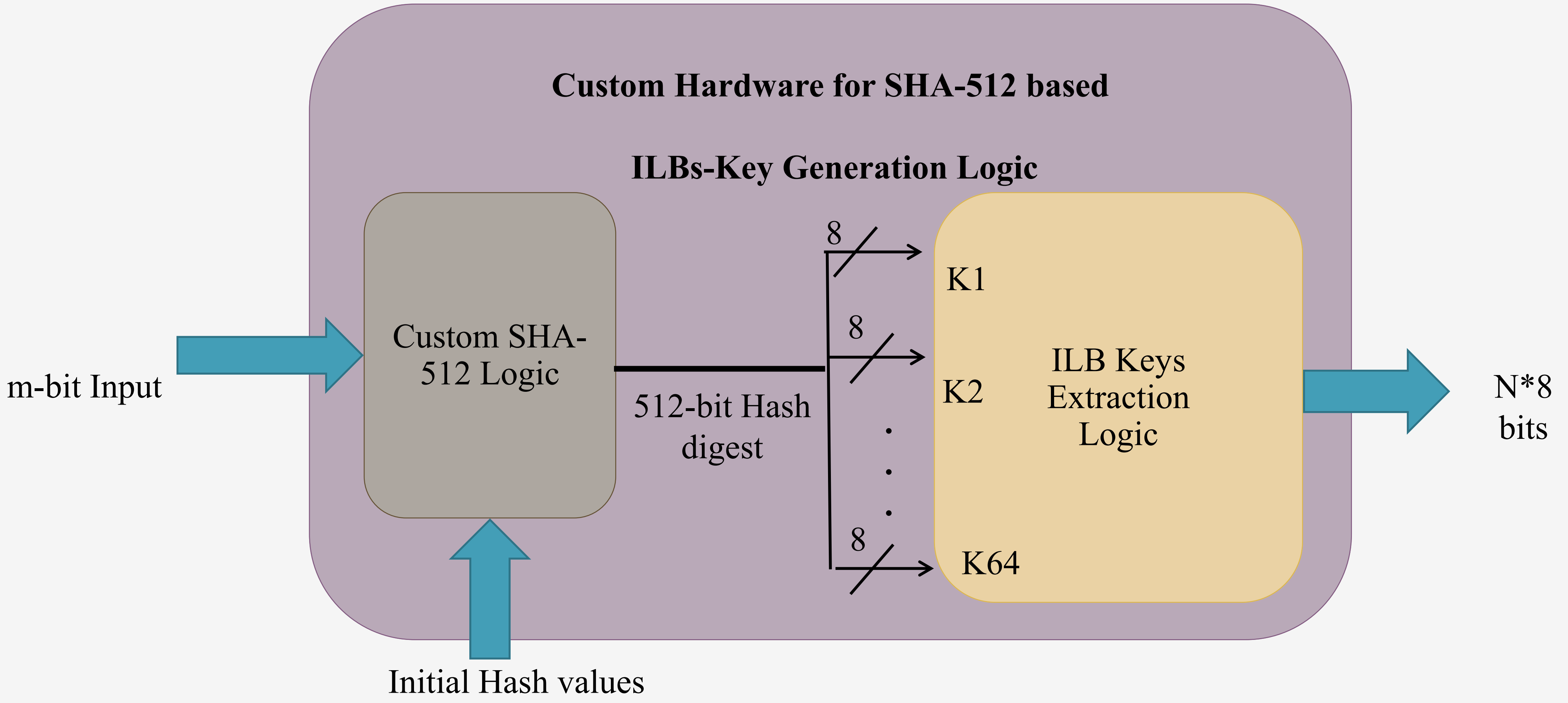
Overview Protection scenario using SHA-512 based ILBs-key generation hardware


m-bit Input
Custom Hardware for SHA-512 based ILBs-key generation logic
512 bits (64*8 bits) encrypted output used as keys for 'N' ILBs (N=64)
Reconfigured 'N' ILBS
'(T_{ilb-} N)' ILBS
Non-encrypted keys for ILBs
Primary Input
Functionally Obfuscated DSP Core
Primary Output

Overview

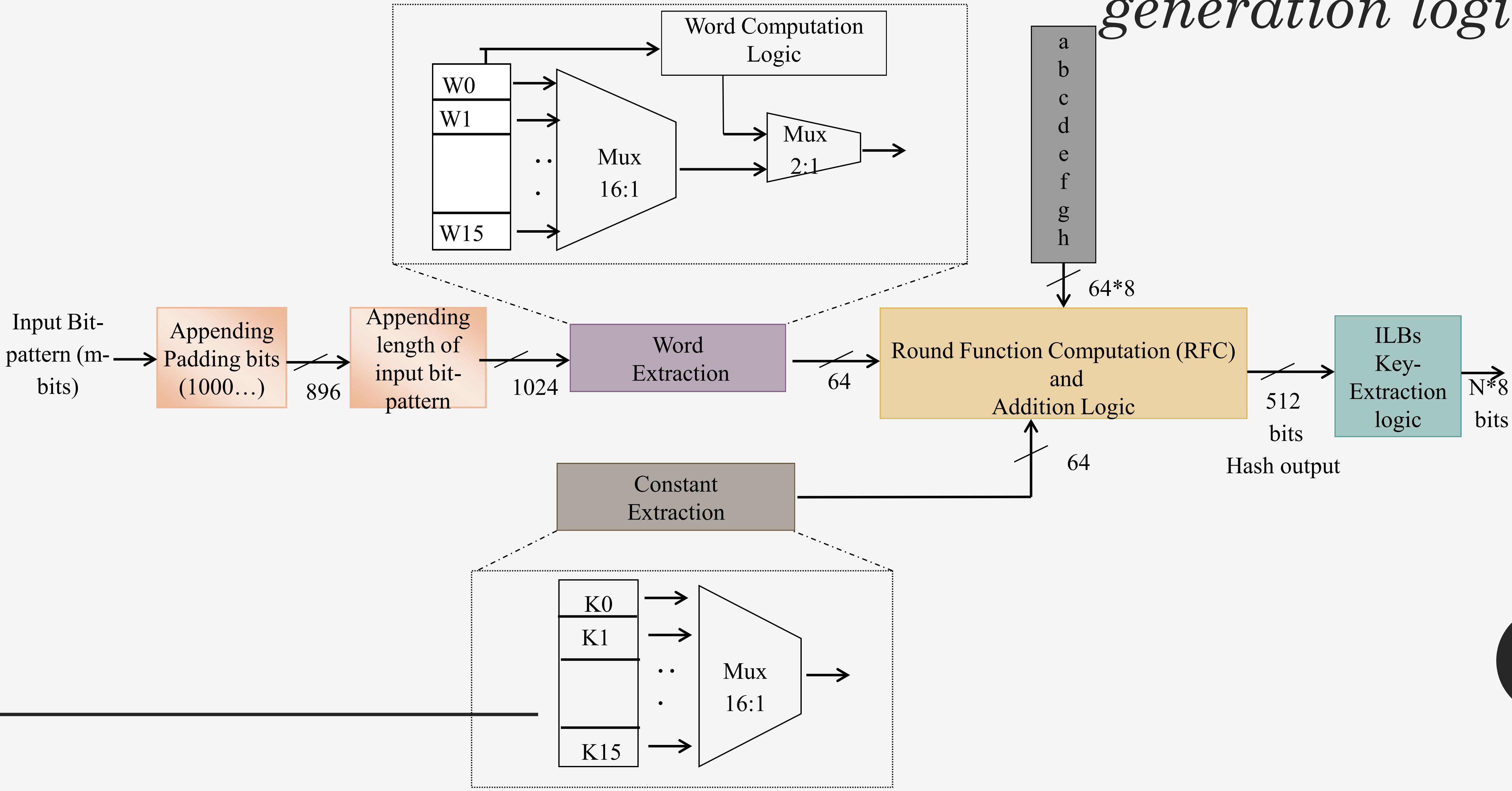
- The input to the custom hardware of SHA-512 based ILBs-key generation logic is of arbitrary length (m bits) and output is N*8 bits, where N is the number of ILBs to be reconfigured and ‘8’ is the number of key bits per ILB.
- The maximum value of N can be 64 because the total length of the hash digest is of 512 bits and one ILB needs an 8-bit key to be activated.
- Further, the keys of remaining ILBs of the functionally obfuscated design are kept non-encrypted.
- The secure encryption of ILB keys using SHA-512 based key generation logic leads to robust structural reconfiguration of a number of ILBs simultaneously.
- Post synthesis, the reconfigured ILBs structures are camouflaged with the DSP circuit, thus resulting into enhanced security against the removal attack.

Block diagram of custom Hardware for SHA-512 based ILBs-key generation logic



- The custom SHA-512 based key generation hardware consists of following two logics: **(i) SHA-512 custom logic (ii) ILB keys-extraction logic**.
- An arbitrary length (m-bit) input and initial hash buffer values are fed to the SHA-512 custom logic and 512-bit hash digest is generated at the output.
- The custom SHA-512 logic first converts the m-bit input into a block of size 1024-bit.
- This conversion process is shown in the next sub-section.
- Further, the custom SHA-512 logic processes the 1024-bit block and hash buffer values (a, b, c, d, e, f, g and h) and updates the hash buffer in each round.
- The number of rounds of execution of SHA-512 logic is customized (based on designer).
- After the final round, the eight hash buffers carry the 512-bit digest.
- Further, the ILB keys-extraction logic extracts the chunk of 8-bits (from 512-bit hash digest) that act as encrypted keys for ILBs.
- How many chunks of 8 bits are extracted depends on the number of ILBs to be reconfigured. To reconfigure 'N' number of ILBs of a functionally obfuscated DSP core, the output from the keys-extraction logic is extracted in the form of **N*8 bits**.

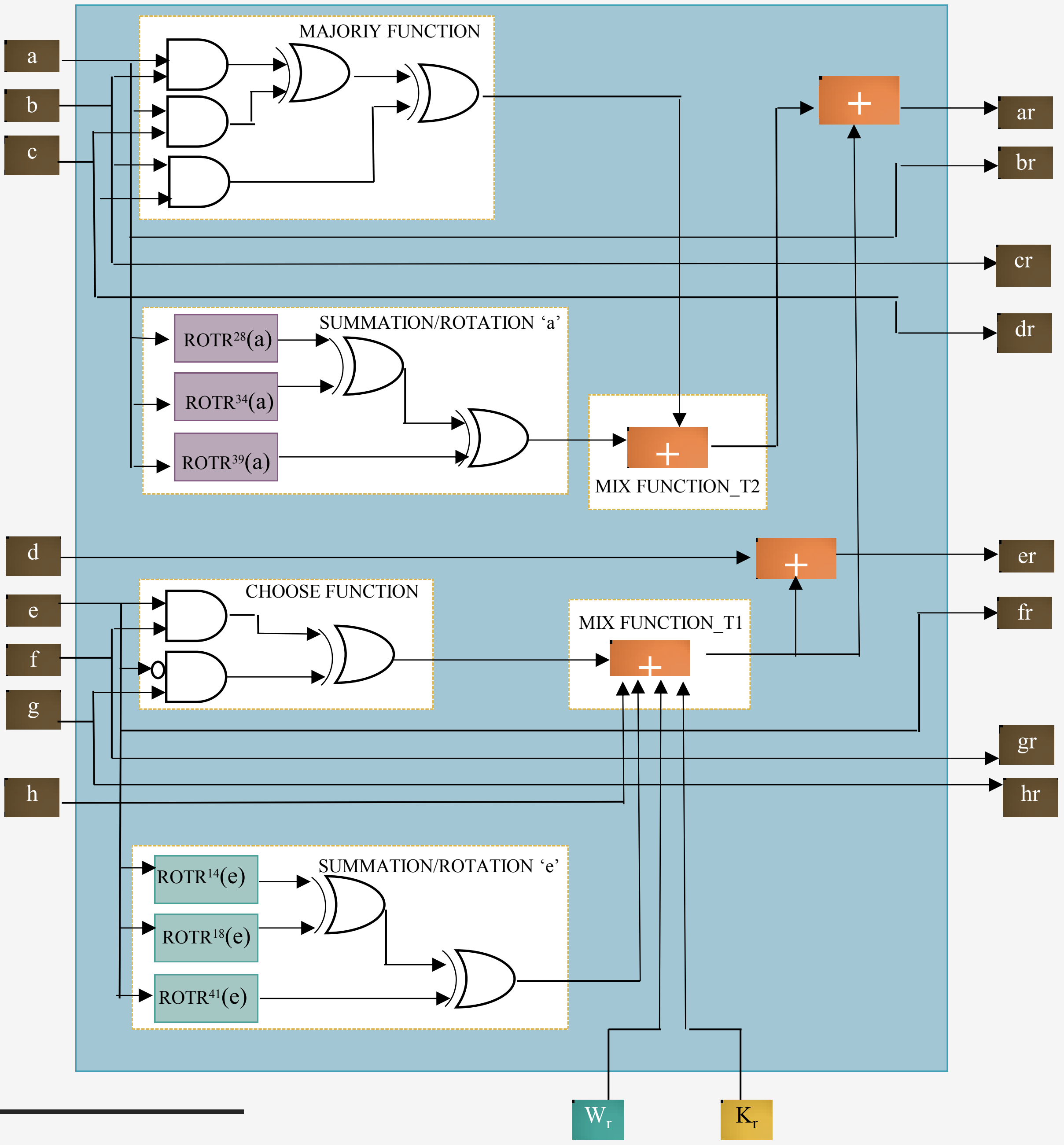
Hardware design of custom SHA-512 based key generation logic



Word Computation Logic
W0
W1
W15
Mux 16:1
Mux 2:1
a b c d e f g h
64*8
Input Bit-pattern (m-bits)
Appending Padding bits (1000…)
896
Appending length of input bit-pattern
1024
Word Extraction
64
Round Function Computation (RFC) and Addition Logic
512 bits
Hash output
ILBs Key-Extraction logic
N*8 bits
64
Constant Extraction
K0
K1
K15
Mux 16:1

- The input to the design is a bit-string of arbitrary length (m-bit) and output is the encrypted keys for the ILBs of a functionally obfuscated design.
- The steps are:
- ***Appending padding bits:*** In this step, the input bit-string of arbitrary length is transformed in to a block of 896-bit by padding the following bit-sequence: "10000………" (assuming the length of input bit-string is less than 896 bits)
- ***Appending input bit-string length*****:** The block of 896-bit is further transformed in to a block of 1024-bit by padding the 128-bit representation of the length of input bit-string.
- ***Word extraction*****:** In this step, a 64-bit word is extracted to be used in the round function computation (RFC) step. The word is either extracted from the 1024-bit block (obtained in previous step) or computed using 'word computation logic' hardware.
- ***Constant extraction:*** The 16 constants (K0, k1,…, K15) used in the custom SHA-512 logic are all standard values (same as standard SHA). The size of each constant is 64-bit. The extraction of one constant (Kr) from the set of 16 constants is performed for each round of RFC.
- **Hash buffer processing**: In the first round, this step provides the initial hash buffer values to the RFC step. Further for subsequent rounds of RFC, the updated hash buffer values are obtained from this step.

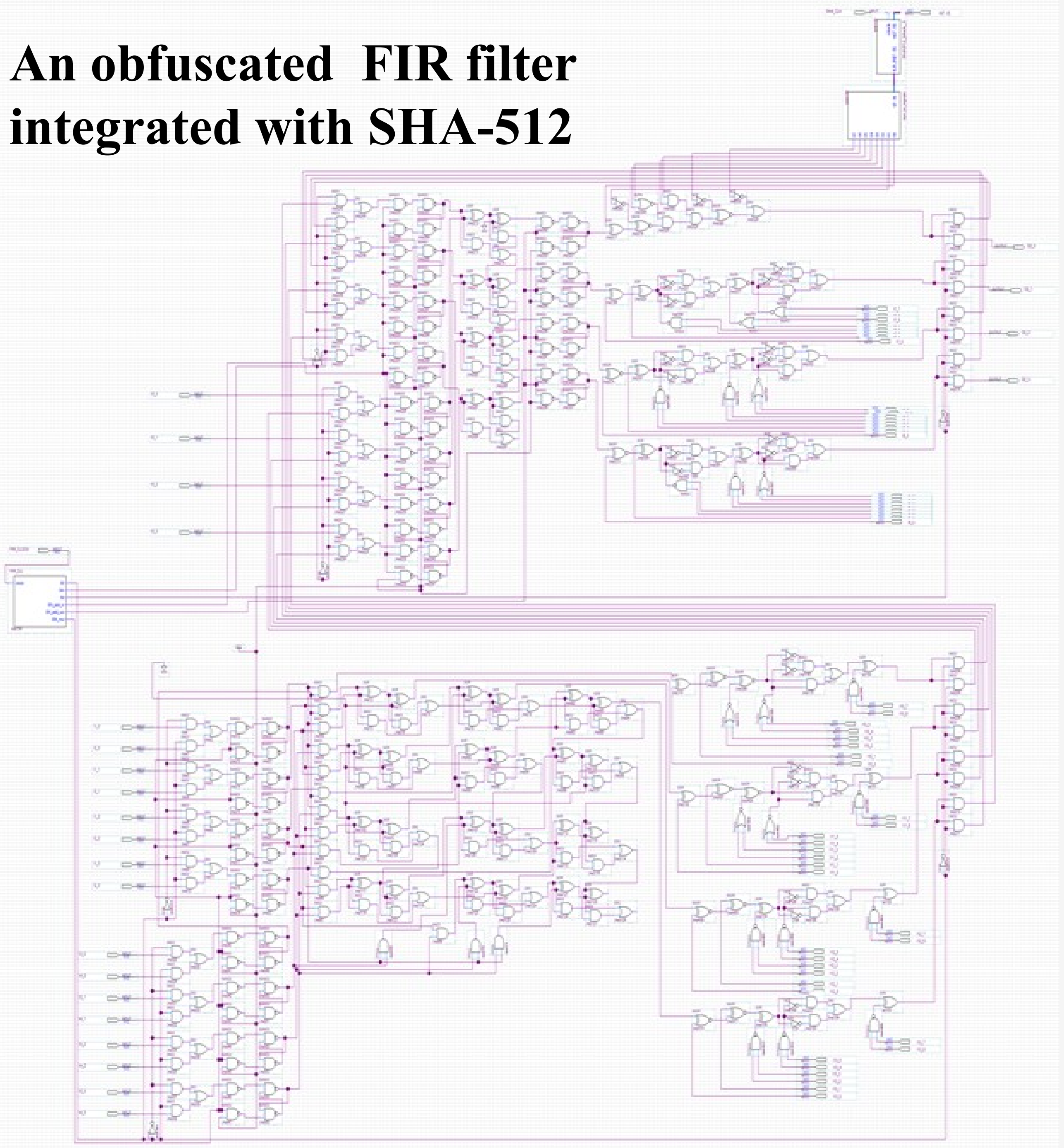
- **Round Function Computation (RFC):**
    - the round function depends on the previous three steps viz. word extraction, constant extraction and hash buffer processing.
    - A 64-bit word (Wr), a constant (Kr) and eight hash buffer values (a, b, c, d, e, f, g and h) obtained from previous steps are processed in RFC hardware to compute the updated hash values in each round.
    - The RFC hardware performs the following six basic functions: (i) MAJORITY function (ii) SUMMATION/ ROTATION ‘a’ (iii) CHOOSE function (iv) SUMMATION/ ROTATION ‘e’ (v) MIX function T1 (vi) MIX function T2.



Round Function Computation hardware of custom SHA-512

- **ILBs keys extraction:**
  - The 512-bit hash-digest (output of final round of RFC) is further fed to the ILBs key-extraction logic.
  - This logic performs the extraction of N*8 bits (from 512-bit hash output) that act as keys of ‘N’ number of ILBs of a functionally obfuscated design. Where, the maximum possible value of ‘N’ can be 64.
  - Thus, upto 64 ILBs of a functionally obfuscated design can be structurally reconfigured using SHA-512 based key generation hardware.
  - Suppose ‘N’ number of ILBs of a functionally obfuscated design is to be structurally reconfigured. Therefore, N*8 bits are extracted (from the 512-bit hash output) by the key-extraction logic.
  - According to these N*8 bits, ‘N’ ILBs structure are reconfigured by reorganizing their internal gates such that the functionality of the ILBs does not change (output of ILBs is valid only on applying correct keys).

An obfuscated FIR filter integrated with SHA-512



Integration of SHA-512 based ILBs-key generation hardware with a functionally obfuscated DSP core

- ➢ The custom SHA-512 based ILBs-key generation hardware is integrated with a functionally obfuscated DSP core by connecting its output with the keys of ILBs to be reconfigured .
- ➢ The integration of SHA-512 based key generation hardware eliminates the vulnerability of the functionally obfuscated DSP design against the removal attack.

Security of Functionally Obfuscated DSP core against Removal Attack using custom Lightweight Anti-removal Unit

➢ ***Features of the Approach***

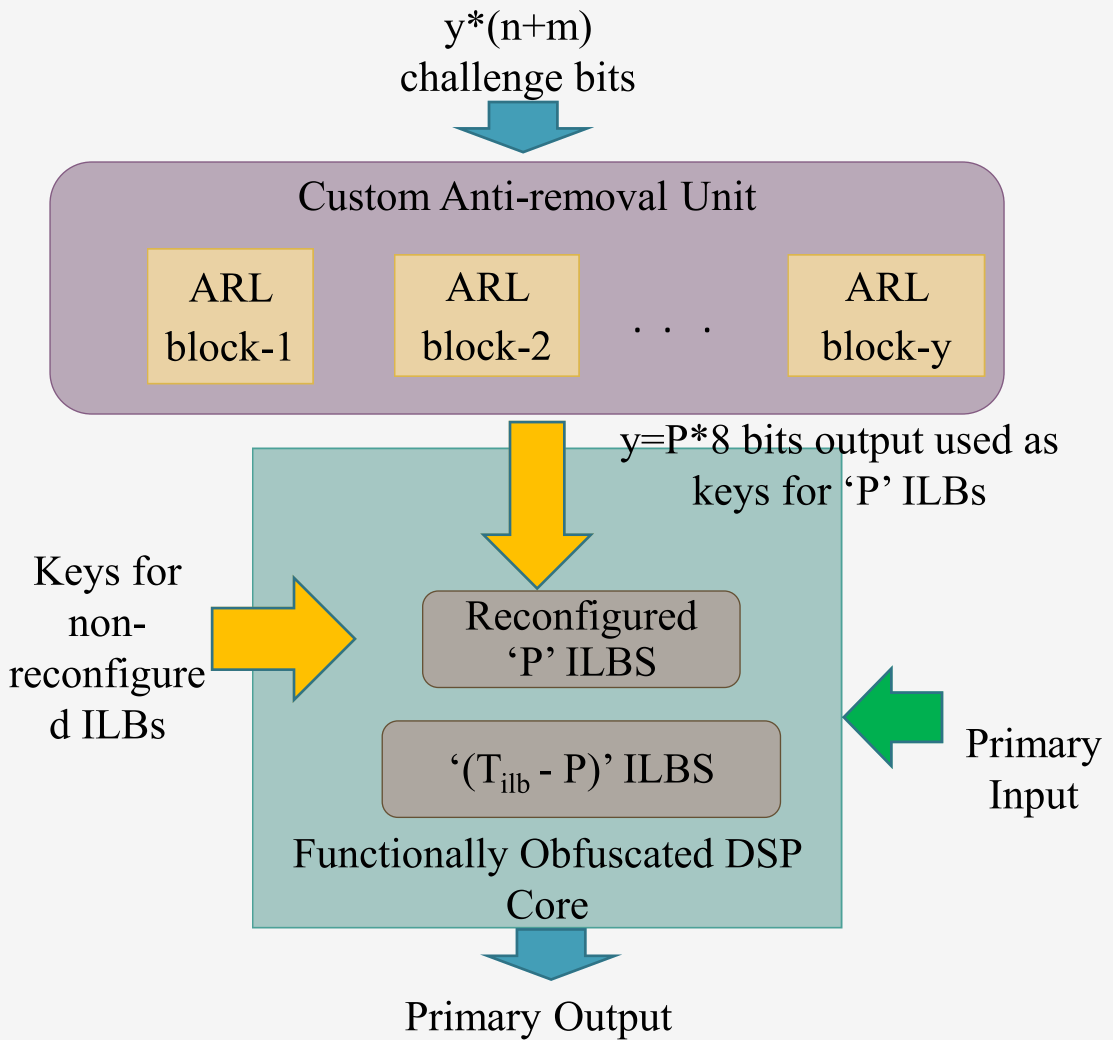
- In this approach, ILB (locking logic) is structurally reconfigured according to the output of a lightweight anti-removal (ARL) unit.
- The detection of ILBs in the complex DSP netlist is difficult because the ILB structure is not known to the attacker in advance. This is because, the ILB structure depends on the architecture and complex challenge inputs of the ARL unit which is only known to the designer.
- Post synthesis, the ILB gate structure gets camouflaged in the entire design netlist and becomes undetectable. Thus, the removal attack on ILBs becomes extremely difficult.
- Since design of lightweight 'ARL unit' can be architecturally customized (i.e. complexity of AND-OR network can be tuned as per designers choice), thus post-synthesis detection of the custom 'ARL unit' in the design netlist (comprising of DSP core and ILBs) is highly challenging.

➢ ***Features of the Approach***

- The architecture of the ARL unit is hidden (and not fixed). The number of ARL blocks used in an ARL unit is not fixed rather it depends on the number of ILBs to be configured.
- The length of the two challenge-bits sets ‘S’ and ‘C’ (n and m) is also decided by the designer. The length of n and m depend on the size and the number of AND-OR networks used in each ARL block.
- To decode one bit of ARL unit output, a total n + m of the challenge input bits should be known in advance.
- To know the 512-bit ARL output, an attacker needs to find 512 * (n + m) challenge bits.
- For n=3 and m=3, the attacker needs to find 3072 bits of the challenge input.
- For an attacker to know the architecture of 64 reconfigured ILBs, 512 bits of ILB keys (for n=3 and m=3) are required to be decoded which in turn requires upto 3027 bits of challenge input to be decoded.
- In order to reconfigure 64 ILBs, only one ARL unit is integrated with a functionally obfuscated DSP design. In case more ILBs are needed to be reconfigured, then only the number of ARL blocks are required to be increased in the same ARL unit.

Overview of ARL based Approach

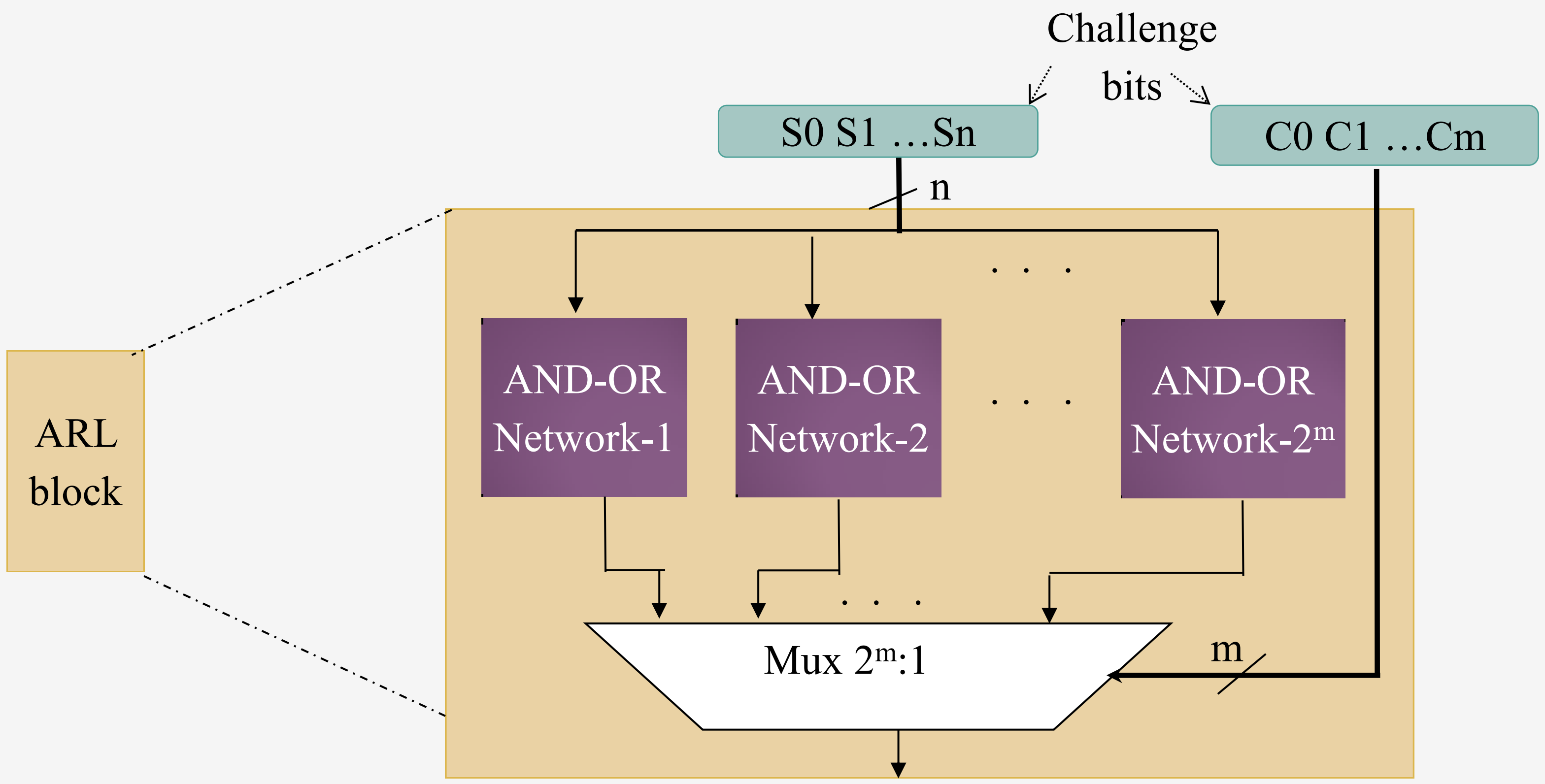
- A custom lightweight and more secure solution against the removal attack has been proposed.
- The ARL unit generates the key-bits for ILBs of a functionally obfuscated DSP core.
- Based on these key-bits, ILBs get structurally reconfigured ensuring security against the removal attack.



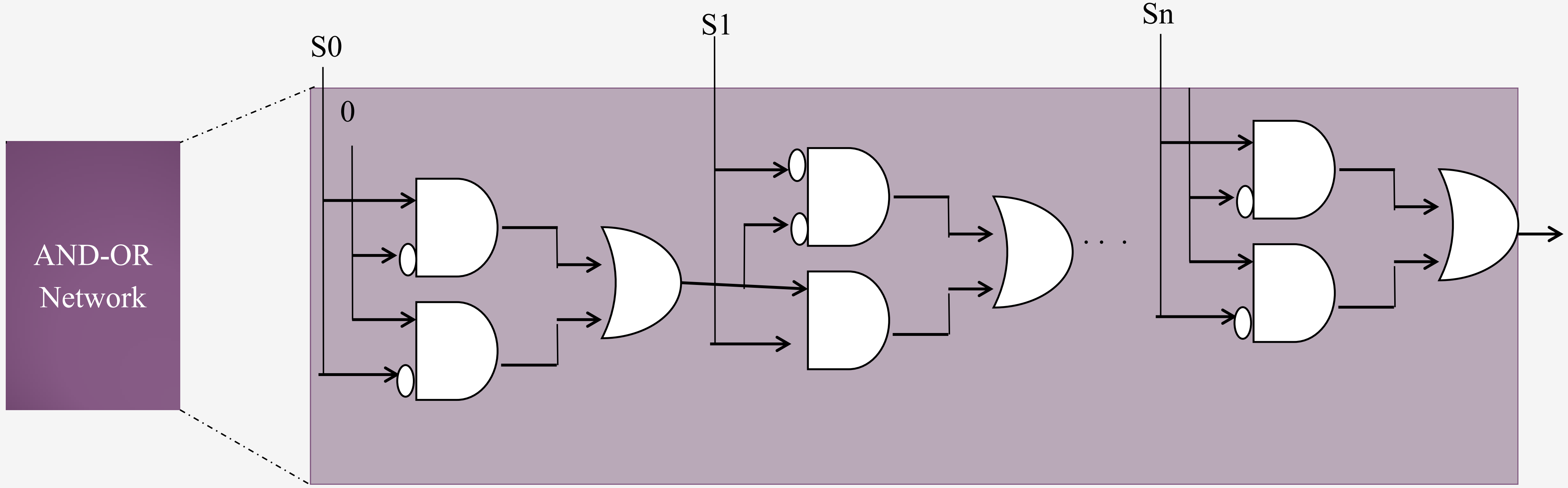
A protection scenario of a functionally obfuscated DSP core against the removal attack using ARL

- The number of ARL blocks used in an ARL unit is decided based on the number of ILBs a designer intends to reconfigure.
- For example, say a designer intends to reconfigure ‘P’ number of ILB and each ILB activates on applying an 8-bit key.
- Since each ARL block generates only one bit key, therefore total P*8 ARL blocks need to be used in the ARL unit.
- Since an attacker is unaware of the custom design of the ARL unit, therefore he/she fails to find the keys of reconfigured ILBs.
- Thus the attacker becomes unable to derive the ILB structures based on 8-bit keys. This renders the ILBs undetectable for the attacker, hence thwarting the removal attack on ILBs.
- Further, as the ARL unit architecture is customizable and also not known to the attacker, thus ARL unit itself remains secured against the removal attack.
- Additionally, post synthesis, the internal gates of ILBs and ARL unit are further camouflaged in the entire design.
- This leads to the detection of ILBs by the attacker more challenging, thus enhancing the security against the removal attack

Hardware Design: Overview of ARL block



Hardware Design: Overview of an AND-OR network used in ARL block



Understanding the Hardware Design

- Several ARL blocks jointly constitute an ARL unit, where each ARL block is responsible for producing only a single key-bit for ILBs.
- For producing 'y' number of key-bits, 'y' ARL blocks are used in an ARL unit, where y=P*8. Here P represents the number of ILBs to be reconfigured and '8' is the number of key-bits required per ILB.
- An ARL block comprises of several AND-OR networks responsible for producing a single bit of output.
- The number of AND-OR networks and size of each depend on the two sets of the challenge inputs given as follows: S = {S0, S1, …, Sn} and C = {C0, C1, …, Cm}.
- Here, the size of challenge inputs 'S' and 'C' has been given as 'n' and 'm' respectively.
- The size of AND-OR network is decided by the 'n' and number of AND-OR networks in an ARL block is determined by the 'm'.
- An ARL block uses total 2^m AND-OR networks. The size of multiplexer used in the ARL block also depends on 'm'.

Understanding the Hardware Design

- The bit-pattern of challenge input set 'S' decides the value of a key-bit generated from the selected AND-OR network.
- The selection of a AND-OR network depends on the challenge input set 'C'.
- More explicitly, based on the bit-pattern of challenge input 'C', the respective AND-OR network architecture is selected to generate the one bit key.
- Each AND-OR network in an ARL block uses the same set of challenge input 'S'
- The architecture of an ARL unit depends on various factors such as the number of ARL blocks used, the size of challenge inputs 'S' and 'C' which in turn decides the structure of each ARL block and the configuration of AND-OR network.
- These tuning factors of ARL architecture are decided by the designer.
- Thus only designer has the knowledge of the custom ARL unit.
- For an attacker, it is extremely hard to know the exact ARL unit architecture used and its functionality.

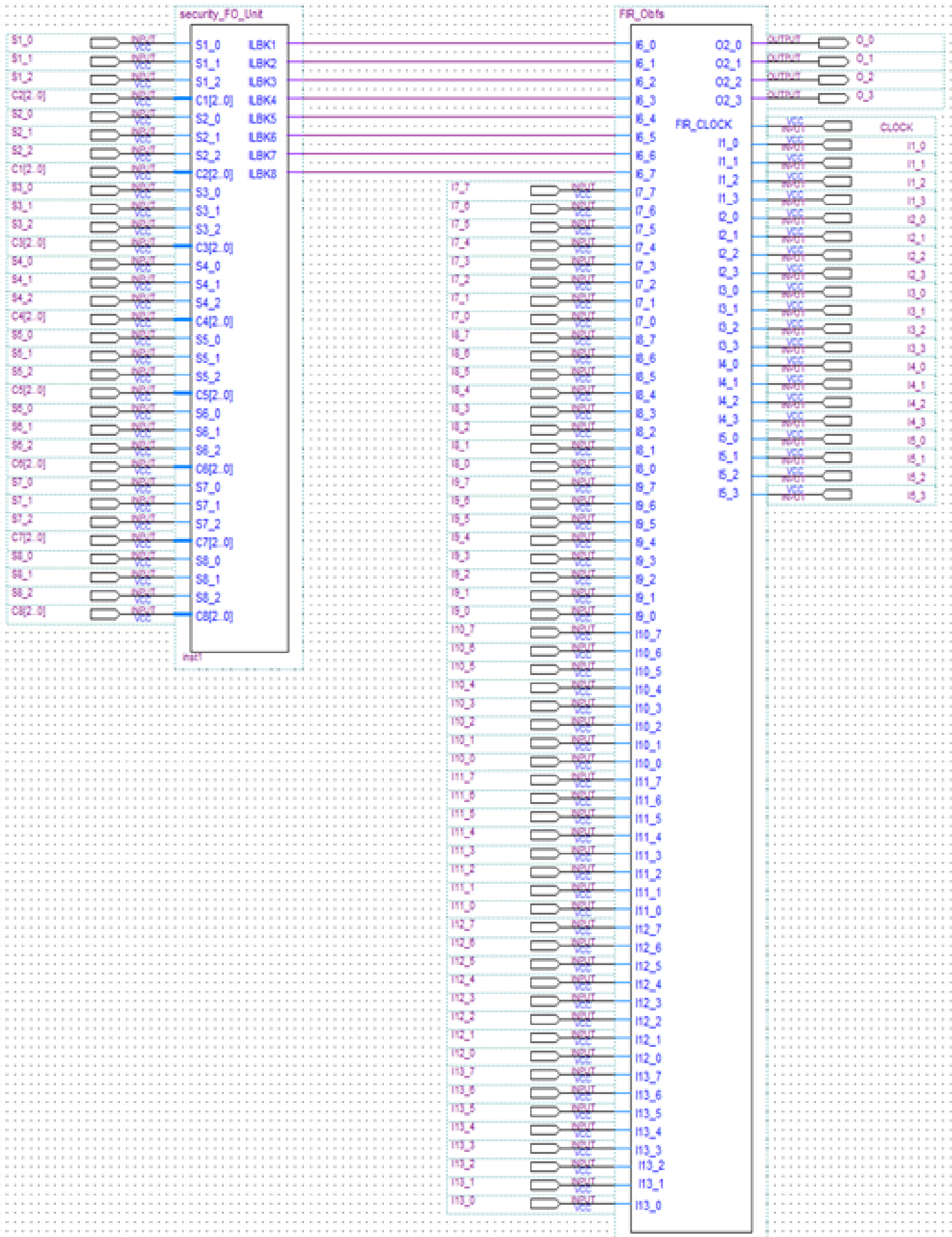
Understanding Security Feature of ARL Unit

- The security in terms of number of challenge bits required to decode the keys of reconfigured ILBs is also achieved high using ARL unit (from an attacker's perspective).
- This is because n + m bits of the challenge inputs are responsible to generate the one bit of an ILB key.
- To reconfigure 64 ILBs, 512 key-bits are generated. This requires 512* (n + m) bits of challenge input.
- If we consider n=3 and m=3, then total 3072 bits of challenge input will be required to decode the 512 key-bits.
- On the contrary, to reconfigure 64 ILBs, 512 key-bits are generated using 4 instances of AES.
- To decode the 512 bits of keys, total 1024 bits (4*128 bits of input + 4*128 bits of fixed secret key) of challenge inputs are required.
- In case of SHA-512 based approach, 1024-bit of input should be known to the attacker to decode the 512-bit output (key-bits for ILBs).
- Hence, ARL unit provides higher security owing to more number of challenge-bits needed for decoding same number of ILB keys (or ILB structure).

Understanding Lightweight Feature of ARL Unit

- The generation of key-bits for ‘X’ more ILBs using ARL unit can be accomplished simply by adding X*8 more ARL blocks.
- However, the generation of key-bits for more ILBs using AES-128/SHA-512 hardware requires one extra instance to be integrated with the functionally obfuscated design.
- This incurs substantial area and power overhead.
- Hence, the ARL unit comes evolves to be a lightweight solution against removal attack on a functionally obfuscated DSP core.

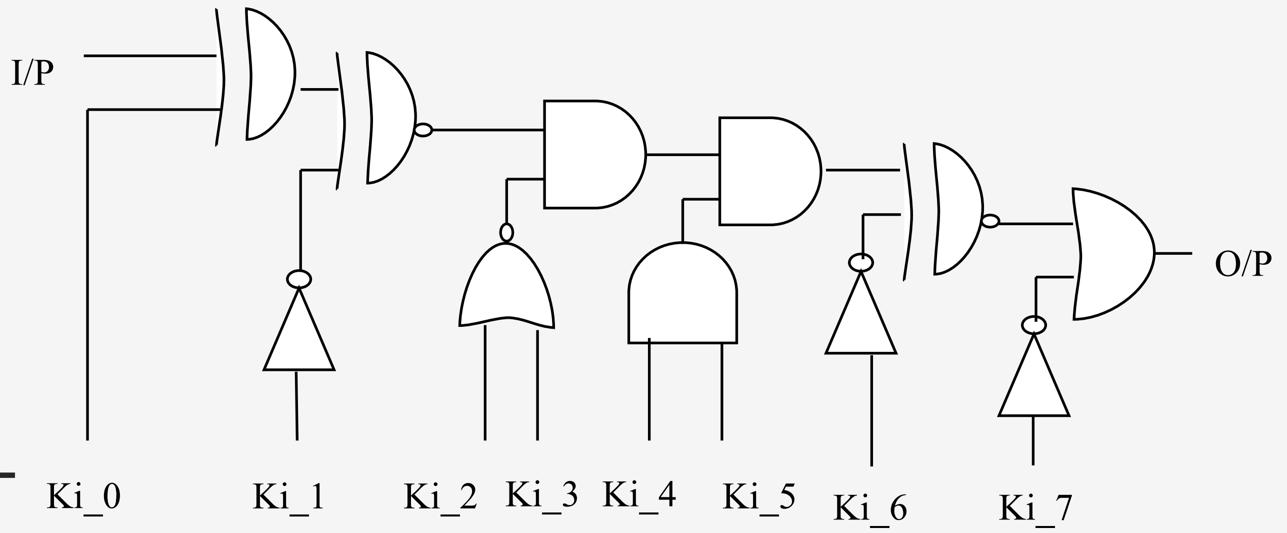
Integration of ARL Unit based ILBs-key generation hardware with a functionally obfuscated DSP core



- ➢The ARL unit is integrated with the functionally obfuscated DSP core to generate ILB keys.
- ➢A top level diagram of custom lightweight ARL unit integrated with the FIR filter design is shown.

Demonstration of ILB Structure Reconfiguration

- The reconfiguration of an ILB structure is performed based on its key value which should be unresolvable by an attacker.
- The randomness of the keys (for the attacker) is ensured by connecting the keys of a sub-set of ILBs with the output of AES-128 or SHA-512 based key generation hardware or lightweight ARL unit.
- Therefore, the ILBs of a functionally obfuscated design use the random values of keys generated.
- According to these keys, the reconfiguration of ILBs structure is performed by rearranging the internal gates in such a fashion that the functionality of the ILBs do not change (i.e. it produces correct output only upon applying a valid key).
- For example, a reconfigured ILB based on an 8-bit key (10110000) is shown.
- Due to structural reconfiguration, no extra gates in the ILBs are added; rather internal rearrangement of the gates occurs.



Structurally reconfigured i^{th} ILB based on key input (10110000)

Analysis on Case Studies

Security Analysis:

Comparative study between **"obfuscated DSP core with custom SHA-512 based key encryption hardware" vs "obfuscated DSP core with custom AES-128 based key encryption hardware"**

DSP applications (32-bit)	# of key-bits generated for ILBs of a functionally obfuscated DSP design		% increase in security using (Sengupta and Rathor, 2019) in terms of more % key bits generated
	SHA-512 based	AES-128 based	
FIR	512	256	50 %
ARF	512	256	50 %
DWT	512	256	50 %
IIR	512	256	50 %
MESA INTERPOLATE	512	384	25 %
JPEG IDCT	512	384	25 %

- The security analysis of the custom SHA-512 based key encryption hardware is performed on the basis of the number of encrypted key bits generated for ILBs.
- If more number of ILB key-bits is generated, then more number of ILBs of a functionally obfuscated DSP design can be structurally reconfigured (resulting into enhanced security).
- For the comparative security analysis, the SHA-512 based key encryption logic has been compared with the AES-128 logic in terms of the number of encrypted key bits generated for ILBs.
- The custom SHA-512 based key encryption hardware is capable to generate key-bits for more number of ILBs in contrast to custom AES-128 based key encryption hardware.
- Therefore, more number of ILBs in an obfuscated DSP design can be structurally reconfigured using SHA-512 based approach, thus providing higher security against the removal attack.
- This improvement in security compared to AES-128 based approach is achieved because single SHA-512 based key encryption hardware is capable to produce 512-bit output (that is fed as 512 key-bits for ILBs), hence upto 64 ILBs can be structurally reconfigured (assuming each ILB requires an 8-bit key) simultaneously.
- On the contrary, one AES-128 is capable to produce 128-bit output resulting into structural reconfiguration of 16 ILBs only (which is much lesser than SHA-512) simultaneously.
- **Larger number of ILBs structurally reconfigured results into stronger obfuscation, thus leading to greater indistinguishability in the gate level design netlist**.

Analysis on Case Studies

Security Analysis:

Comparative study between **"obfuscated DSP core with ARL unit"** vs **"obfuscated DSP core with AES-128 based key encryption hardware:**

DSP applications (32-bit)	No. of challenge -bits		% increase in security using ARL
	ARL based	AES Based	
FIR	1536	512	66.67 %
ARF	1536	512	66.67 %
DWT	1536	512	66.67 %
IIR	1536	512	66.67 %
MESA INTERPOLATE	2304	768	66.67 %
JPEG IDCT	2304	768	66.67 %

- The security analysis of the ARL unit based approach is performed on the basis of the number of challenge bits required to decode/find the key bits of the reconfigured ILBs.
- If more number of challenge bits is required, then it is harder to guess/find the key-bits of reconfigured ILBs of a functionally obfuscated DSP design (resulting into enhanced security).
- For the comparative security analysis, the ARL unit based approach has been compared with the AES-128 based approach in terms of the number of challenge bits required to decode/find the key bits of ILBs (keeping the number of key bits of ILBs same for both approaches).
- The ARL unit requires more challenge bits in contrast to AES-128 to generate same number of key bits for the ILBs of a functionally obfuscated DSP core.
- Therefore with the ARL unit, more challenge bits are needed to be decoded for the same number of key-bits.
- Thus, the ARL unit provides enhanced security against the removal attack owing to increase in the complexity of finding the challenge bits in contrast to AES-128 (from an attacker's perspective).

Conclusion

- ❑ This module highlights the various hardware threats for functionally obfuscated DSP cores. Among all the potential attacks, the removal attack has been emphasized more for discussion in this chapter.
- ❑ As a solution to the removal attack on the functionally obfuscated DSP cores, three recent security techniques (custom AES-128 based approach, custom SHA-512, custom ARL unit based approach) have been discussed.
- ❑ This module provides a comprehensive perspective on the following concepts :

- ➢ Attack scenario for functionally obfuscated DSP cores.
- ➢ Removal attack on functionally obfuscated DSP cores.
- ➢ Solution to removal attack using custom AES-128 based approach.
- ➢ Solution to removal attack using custom SHA-512 based key encryption hardware.
- ➢ Solution to removal attack using custom lightweight ARL unit.
- ➢ Comparative analysis of different approaches used for security of functionally obfuscated DSP cores.

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(1), pp. 83-88.
- A. Sengupta, D. Kachave and D. Roy (2019), "Low Cost Functional Obfuscation of Reusable IP Cores Used in CE Hardware Through Robust Locking," in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 38(4), pp. 604-616.
- A. Sengupta and S. P. Mohanty (2018), 'Functional Obfuscation of DSP Cores Using Robust Logic Locking and Encryption,' *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, Hong Kong, pp. 709-713.
- A. Sengupta and M. Rathor (2019), 'Security of Functionally Obfuscated DSP Core against Removal Attack Using SHA-512 Based Key Encryption Hardware,' *in IEEE Access*, vol. 7, pp. 4598-4610.
- M. Rathor and A. Sengupta (2019), 'Low-cost robust anti-removal logic for protecting functionally obfuscated DSP core against removal attack,' in *Electronics Letters*, vol. 55(7), pp. 374-376.
- R. Torrance and D. James (2009), 'The State-of-the-Art in IC Reverse Engineering,' *in CHES*, vol. 5747, pp. 363-381.
- Anirban Sengupta, Saraju Mohanty, "Advanced Encryption Standard (AES) and its Hardware Watermarking for Ownership Protection", *IET Book: IP Core Protection and Hardware-Assisted Security for Consumer Electronics*, e-ISBN: 9781785618000, 2019, pp. 123-170.
- M. Yasin, J. J. Rajendran, O. Sinanoglu, and R. Karri (2016), 'On Improving the Security of Logic Locking,' *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 35(9), pp. 1411-1424.
- P. Subramanyan, S. Ray, and S. Malik (2015), 'Evaluating the security of logic encryption algorithms,' *In IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pp. 137-143.
- Y. Xie and A. Srivastava (2019), 'Anti-SAT: Mitigating SAT Attack on Logic Locking,' *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 38(2), pp. 199-207.
- M. Yasin, B. Mazumdar, O. Sinanoglu, and J. Rajendran (2017), 'Removal Attacks on Logic Locking and Camouflaging Tech-niques,' *IEEE Transactions on Emerging Topics in Computing*, doi: 10.1109/TETC.2017.2740364.

Thank You