DFT-2023
IEEE International Symposium on
Defect and Fault Tolerance in VLSI and Nanotechnology Systems

Fault Secured JPEG-Codec Hardware Accelerator with Piracy Detective Control using Secure Fingerprint Template

Authors: Rahul Chaurasia, Abhinav Reddy Asireddy, Anirban Sengupta

Speaker Name: Rahul Chaurasia

Post-Doc Researcher, TRF

Computer Science and Engineering

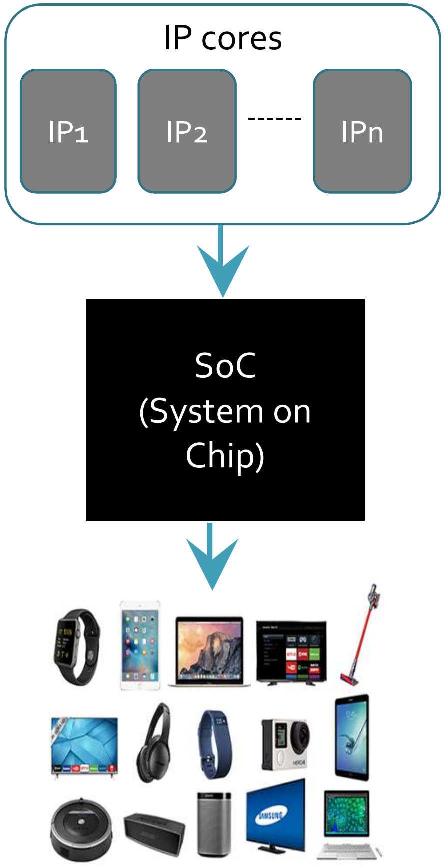
Indian Institute of Technology, Indore, India

Outline

- ❑Introduction
- ❑Contemporary Approaches
- ❑Overview of Proposed Methodology
- ❑Discussion on Proposed Methodology
- ❑Results and Analysis

Introduction

- Hardware accelerators form key components of SoCs used in computing/CE systems.
- Highly **computationally intensive** nature of application.
- JPEG-codec is widely used in medical applications and digital imaging devices for tasks related to image processing (*compression and decompression)*.
- The correct functionality of hardware design may be affected due to **occurrence of faults** (vulnerabilities emanating from SEU).
- **Globalized supply chain** involves untrustworthy third-party IP (**3PIP**) vendor houses.
- Designing its hardware accelerator is not sufficient as it necessitates protection from hardware security threats also.
- **Robust and seamless detective security control**.



IP cores

IP1 IP2 ------ IPn

SoC (System on Chip)

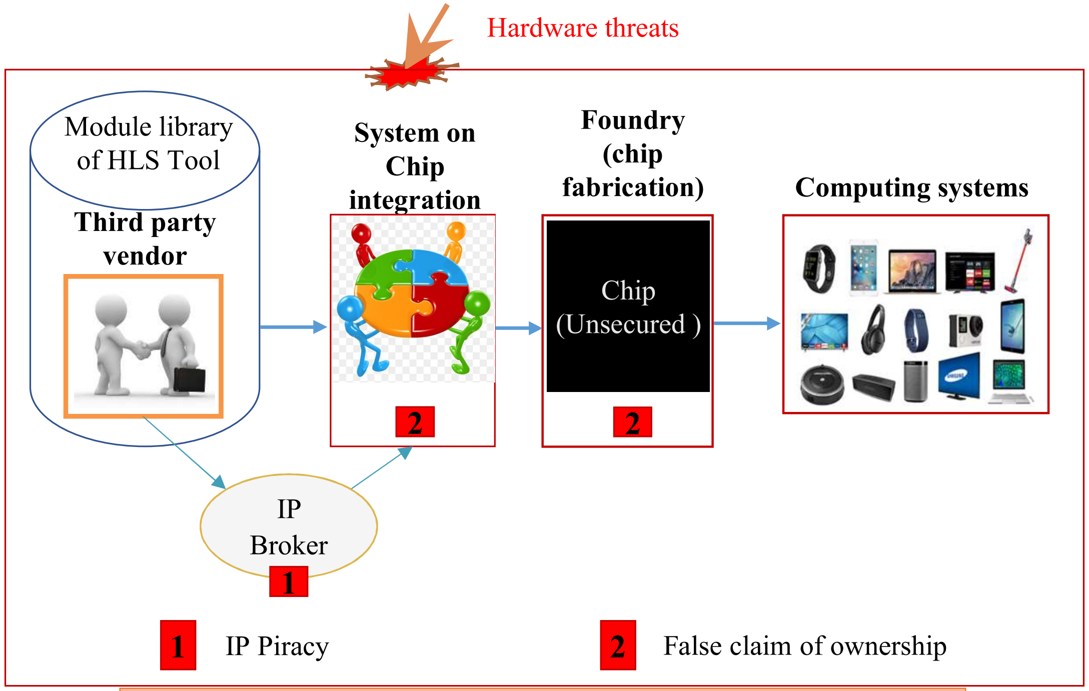
IC: Integrated circuit, **SoCs**: system-on-chips, **CE**: Consumer Electronics, **SEU**: single event upsets

Threat Model: Cont.

- ❑ In untrustworthy IP design houses, an attacker or an IP broker may attempt to produce pirated IP versions without the knowledge of original IP owner.

- The pirated/tampered designs could be hazardous as they may contain malicious logic, thereby causing integrity and reliability hazards.

- ❑ An adversary may also attempt to achieve piracy evasion by exactly regenerating and copying the original security signature into fake/pirated design versions.



Usual practice of IC design being used in computing systems

Novel Contributions

a) A novel methodology for generating transient fault secured JPEG-codec hardware accelerator with seamless piracy detective control has been proposed.

b) The proposed approach ensures the detection of pirated design versions through the embedded key controlled secure fingerprint template of the IP vendor as countermeasure.

c) The proposed generated secure fingerprint template is capable of offering higher tamper tolerance and lesser probability of coincidence than related works [5]-[10].

Advantages of Generating Secure Fingerprint Template

- Multiple non-linkable larger secure fingerprint templates can be generated.
- For an adversary regeneration of secure fingerprint template is not possible.
- Generates larger template size (by choosing larger key).

Contemporary Approaches

- **Hardware watermarking approach:** [5], [6] watermark is generated using auxiliary signature variable combinations. It provides the piracy detection by embedding the watermark of IP vendor.
  - Vulnerable if the relevant information such as digit encoding into security constraints, signature size and combinations of digits are leaked to an adversary, s/he can replicate and re-use it. This renders the watermark a weaker secret mark.
- **Digital Signature:** [7] digital signature is generated through encoding, SHA-512 and RSA encryption.
  - Involves complex computation during signature generation and also results into higher design cost.
- **Hardware steganography** [8] address the IP counterfeiting threat by embedding the secret stego-constraints into the design. These stego constraints are generated based on stego-keys, encoding rules and entropy threshold parameter.
  - Vulnerable if the encoding rules and the secret value of chosen entropy threshold are leaked to an adversary.
- **Facial and Palmprint biometric** [9], [10] based hardware security approach embeds IP vendor's authentic facial/palmprint biometric constraints into the design. The approaches [9], [10] offers more robust security than [5]-[8].
  - Generates lesser number of hardware security constraints (resulting into higher probability of coincidence and lower tamper tolerance ability of the design) than proposed approach.

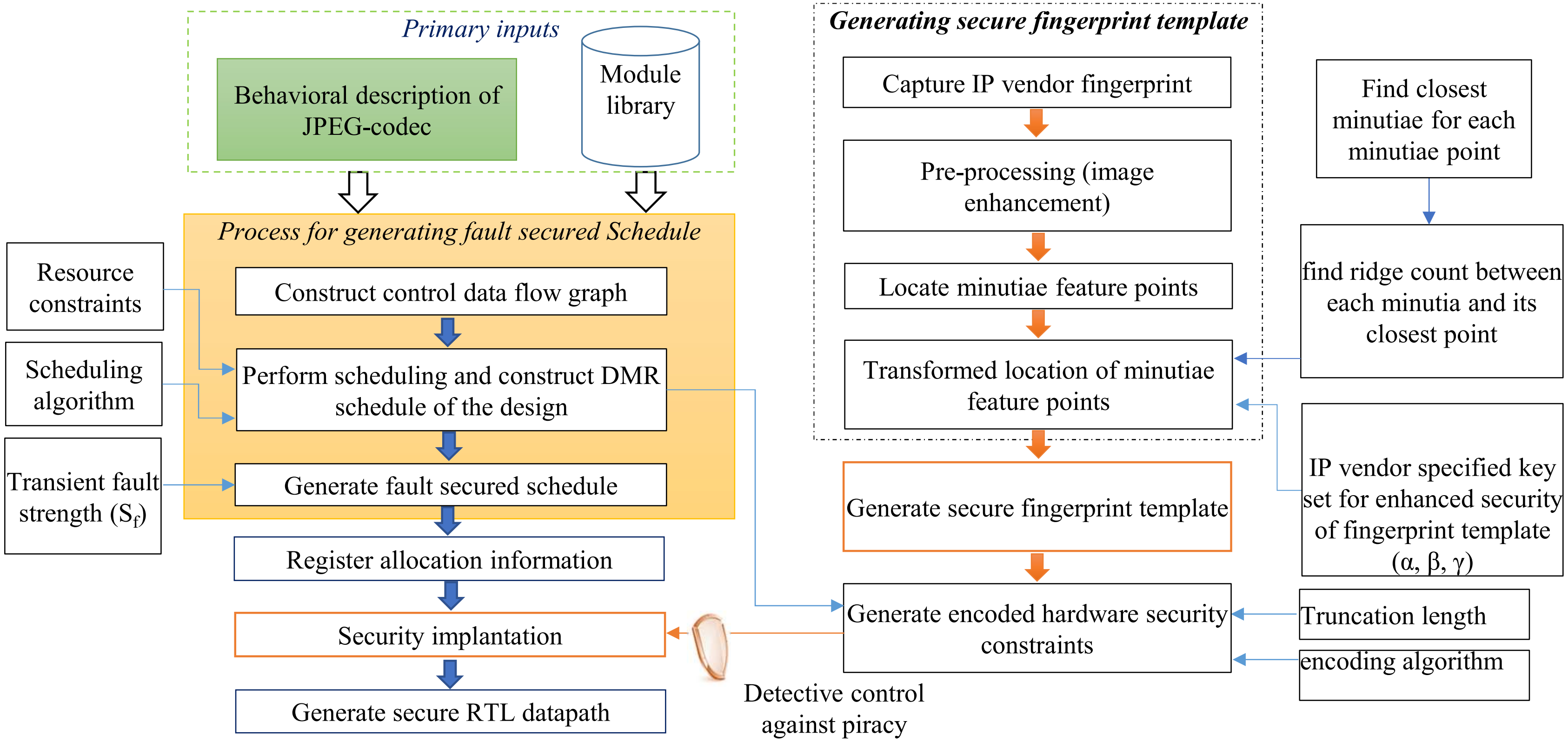
➢ On the contrary, the proposed methodology using secure fingerprint template is capable of generating a greater number of hardware security constraints than [5]-[10], thereby enabling more robust security.

➢ Approaches [5]-[8] do not associate naturally unique identity of IP vendor.

➢ The proposed approach is able to achieve more robust security at almost negligible design cost overhead.

SHA: Secure Hashing Algorithm

Proposed Methodology: Design Flow



Primary inputs
Behavioral description of JPEG-codec
Module library
Process for generating fault secured Schedule
Resource constraints
Scheduling algorithm
Transient fault strength (S_f)
Construct control data flow graph
Perform scheduling and construct DMR schedule of the design
Generate fault secured schedule
Register allocation information
Security implantation
Generate secure RTL datapath
Detective control against piracy
Generating secure fingerprint template
Capture IP vendor fingerprint
Pre-processing (image enhancement)
Locate minutiae feature points
Transformed location of minutiae feature points
Generate secure fingerprint template
Generate encoded hardware security constraints
Find closest minutiae for each minutiae point
find ridge count between each minutia and its closest point
IP vendor specified key set for enhanced security of fingerprint template (α, β, γ)
Truncation length
encoding algorithm

Overview: Bhavioral synthesis-based design flow of the proposed methodology

RTL: Register transfer level

Module 1: Generating Fault Secured Schedule for JPEG-codec

- ➢ In order to do so, firstly its behavioural description/transfer function is transformed into control data flow graph (CDFG):

Step1: Transform the input image (to be compressed) into matrix form.

Step2: Perform matrix slicing and generate non-overlapping matrix or block, each of size 8×8.

Step3: Transform each 8x8 block of pixels using 2-D DCT transformation using following function:

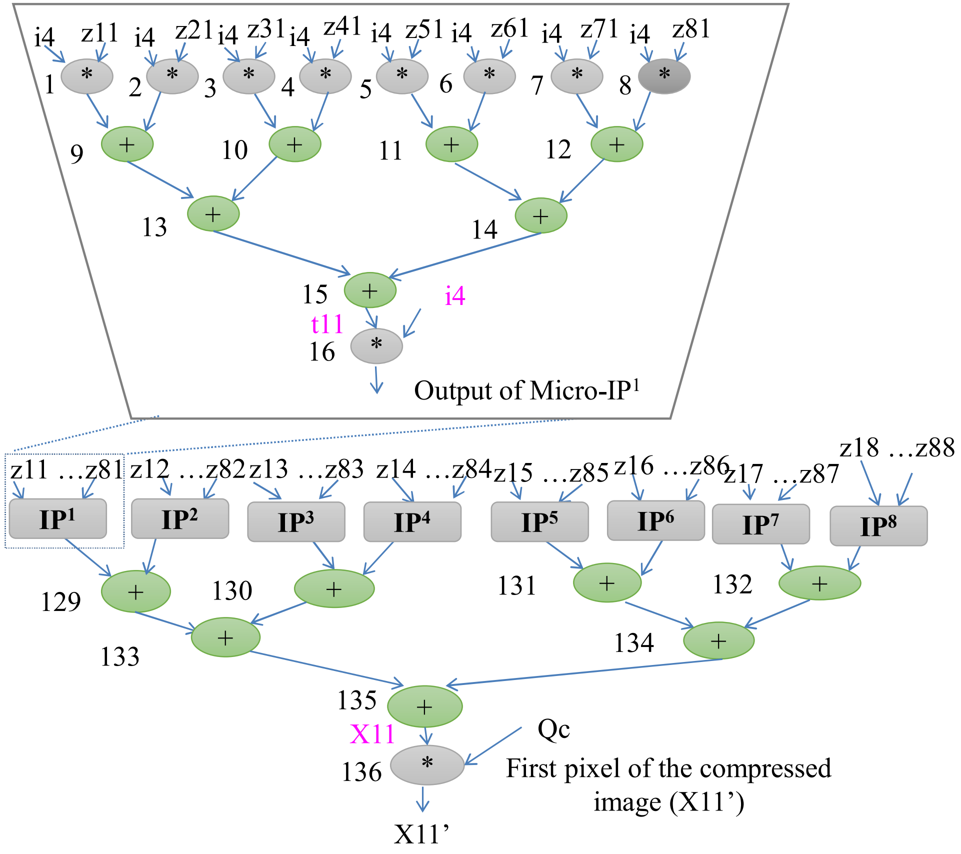
$$X = (I*Z) *I'$$

Step4: Compute the first pixel value of the transformed matrix, ‘X11’ .

$$X11 = (d11*i4) + (d12 *i4) + (d13 *i4) + (d14 *i4) + (d15 *i4) + (d16 *i4) + (d17 *i4) + (d18 *i4)$$

$$d11 = (i4*z11) + (i4*z21) + (i4*z31) + (i4*z41) + (i4*z51) + (i4*z61) + (i4*z71) + (i4*z81)$$

Step5: Now compression using a quantization coefficient Qc.


i4 z11 i4 z21 i4 z31 i4 z41 i4 z51 i4 z61 i4 z71 i4 z81
1 2 3 4 5 6 7 8
9 10 11 12
13 14
15
i4
t11
16
z11 …z81 z12 …z82 z13 …z83 z14 …z84 z15 …z85 z16 …z86 z17 …z87 z18 …z88
IP1 IP2 IP3 IP4 IP5 IP6 IP7 IP8
129 130 131 132
133 134
135
X11
Qc
136
X11’

Output of Micro-IP1

First pixel of the compressed image (X11’)

Control data flow graph of JPEG-codec

where X denotes the DCT transformed matrix, I denotes the 2-D-DCT coefficient matrix, Z denotes the 8 × 8 block of pixels of input image and I′ denotes the transpose of I.

Cont.

Schedule of Macro IP of JPEG-codec using [3+, 3*]

CS	Opns assign to M1	Opns assign to M2	Opns assign to M3	Opns assign to A1	Opns assign to A2	Opns assign to A3
1	1	2	3			
2	4	5	6	9		
3	7	8	17	10	11	
4	18	19	20	12	13	
5	21	22	23	25	26	14
6	24	33	34	27	29	15
7	35	36	37	28	41	
8	38	39	40	42	30	
9	49	50	51	43	44	45
10	52	53	54	31	57	46
11	55	56	65	58	59	47
12	66	67	68	60	61	
13	69	70	71	73	74	62
14	72	81	82	75	77	63
15	83	84	85	76	89	
16	86	87	88	90	78	
17	97	98	99	91	92	93
18	100	101	102	79	105	94
19	103	104	113	106	107	95
20	114	115	116	108	109	
21	117	118	119	121	122	110
22	120	16	32	123	125	111
23	48	64	80	124	129	
24	96	112		130	126	
25				131	133	127
26	128					
27				132		
28				134		
29				135		
30	136					

- ➢ Subsequently, dual modular redundant (DMR) design is formed.
- ➢ Next the DMR design is scheduled using LIST scheduling algorithm.
- ➢ Next, transient fault secure schedule is generated from DMR scheduled design by employing the fault security rules and by considering transient fault strength (S_f):

Cont.

***Fault security rules*:**

- ***Rule1***- allocate opn (X)ϵU_{OG} and opn (X′)ϵU_{DP} to different operators based on availability.
- ***Rule2***- in case if allocation of distinct operators is not possible, then have the same allocation for X′ (by way of X) in U_{DP} such that:

$$t(X') - t(X) > S_f \qquad (1)$$

- ***Rule3***- if the condition in (1) does not comply, then push X′ (with its successors) ϵ$U_{DP,}$ one control step down at a time and continue the process till rule1 or rule2 holds true.

❑ We assume that transient fault due to single event upset (SEU) affects multiple cycles.

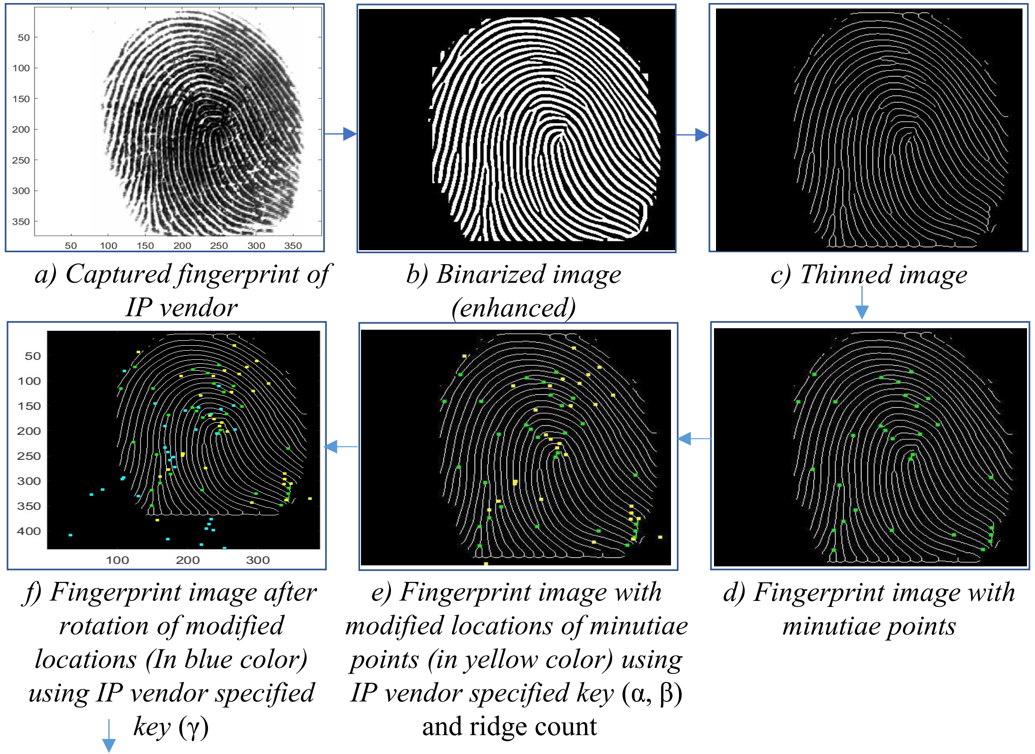
'$\mathbf{U_{OG}}$' : Original unit and '$\mathbf{U_{DP}}$' : Duplicate unit

TABLE I

Fault Secured Schedule for JPEG-codec using [3+, 3*]

Control steps	Opns. assign to M^1	Opns. assign to M^2	Opns. assign to M^3	Opns. assign to A^1	Opns. assign to A^2	Opns. assign to A^3
C1	1	2	3			
C2	4	5	6	129		
C3	7	8	9	130	131	
C4	10	11	12	193	132	
C5	13	14	15	194	133	134
C6	16	17	18	225	195	135
C7	19	20	21	136	137	
C8	22	23	24	196	138	
C9	25	26	27	197	139	140
C10	28	29	30	226	198	141
C11	31	32	33	227	142	143
C12	34	35	36	199	144	
C13	37	38	39	200	145	146
C14	40	41	42	228	201	147
C15	43	44	45	148	149	
C16	46	47	48	202	150	
--	--	--	--	--	--	--
--	--	--	--	--	--	--
C49				267	263	240
C50	256					
C51				264		
C52				268		
C53				270		
C54	272					

Module 2: Secure Fingerprint Template Generation



a) Captured fingerprint of IP vendor

b) Binarized image (enhanced)

c) Thinned image

d) Fingerprint image with minutiae points

e) Fingerprint image with modified locations of minutiae points (in yellow color) using IP vendor specified key (α, β) and ridge count

f) Fingerprint image after rotation of modified locations (In blue color) using IP vendor specified key (γ)

Binarized secure template corresponding to fingerprint image with transformed locations of minutiae points post applying subsequent translation using keys (δ and γ)

Process for generating secure fingerprint template

```
Input: Minutiae points (n) with their dimensions and key set {α, ß, γ}

Output: Secured fingerprint template of IP vendor

Start:
 δ = ⌊α⌋ + ⌊β⌋*2^8   /*keys α, β, γ are 8 bit and δ is of 16-bit size respectively.
 for i = 1 -> n do
   $dist_i$ ← infinity
/* below for loop is to find closest point from each minutia ($x_i$,$y_i$)
   for z = 1 -> n do
     if i ≠ z then
       dist = Euclidian distance between ($x_i$,$y_i$) and ($x_Z$,$y_Z$)
       if $dist_i$ > dist then
         $dist_i$ ← dist
         j = z
       Endif
     Endif
   Endfor
/* translation using keys α, β and ridge line count ($Rc_{ij}$)
   $x_i' \leftarrow x_i + \alpha * Rc_{ij} * \text{Cos}(\beta + \cot^{-1}(x_i - x_j / y_i - y_j))$
   $y_i' \leftarrow y_i + \alpha * Rc_{ij} * \text{Sin}(\beta + \cot^{-1}(x_i - x_j / y_i - y_j))$
/* security enhancement through rotation (w.r.to origin) using key γ
   $x_i'' \leftarrow x_i' * \text{Cos}(\gamma) - y_i' * \text{Sin}(\gamma)$
   $y_i'' \leftarrow x_i' * \text{Sin}(\gamma) + y_i' * \text{Cos}(\gamma)$
 /* final translation using key γ and δ
     $x_i$_new ← $|x_i'' + \delta * \text{Cos}(\gamma)|$
     $y_i$_new ← $|y_i'' + (\gamma)|$
   Endfor
End
```

Pseudo code for generating secure fingerprint template

- The proposed approach exploits the following features of transformed fingerprint to compute secure template:

  i) co-ordinates of minutiae points (x_i,y_i)   ii) orientation (ɵ) and iii) type of minutiae i.e., ridge ending or bifurcation

Module 3: Hardware Security Constraints Generation

❑ The process of generating constraints for hardware security accepts the following inputs:

➢ IP vendor chosen strength of secure fingerprint template,

➢ Encoding algorithm

➢ Storage variable information and their ordering corresponding to fault secured JPEG design.

❖ The generated secure template using IP vendor specified concatenation order of minutiae points, transformation functions and key set is:

“110101110001001001110111011111……..111100011010” (893 bits).

TABLE II

IP VENDOR SPECIFIED ENCODING ALGORITHM TO ENCODE SECURE FINGERPRINT TEMPLATE INTO HARDWARE SECURITY CONSTRAINTS

Bit	Fingerprint template encoding algorithm (For example)
0	Embedding the security constraints during register allocation between ‘even-even’ pairs of storage variable.
1	Embedding the security constraints during register allocation between ‘odd-odd’ pairs of storage variable.

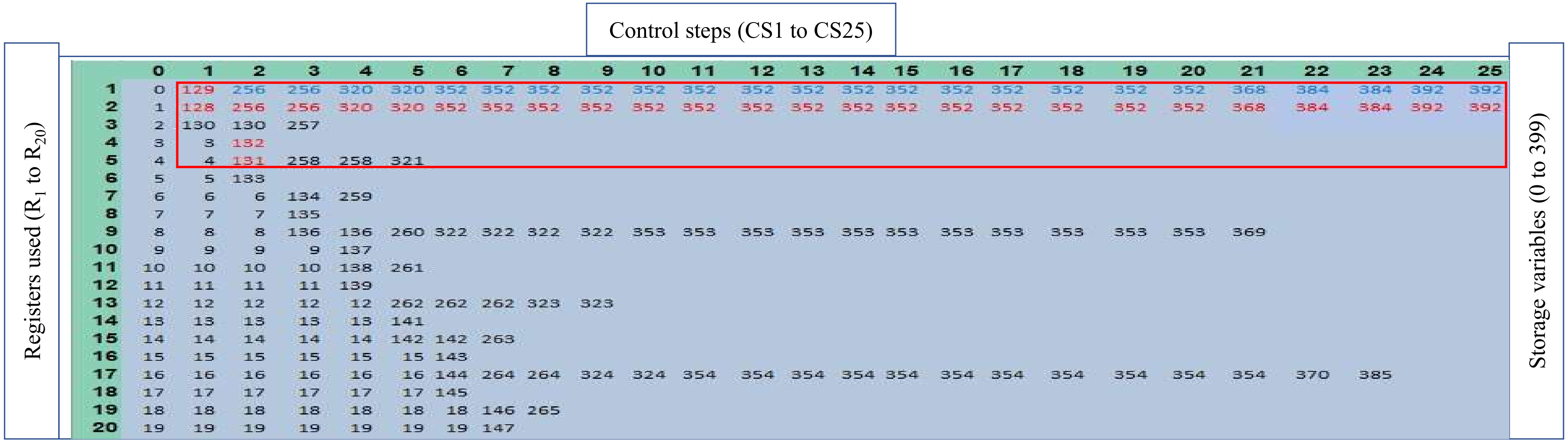
TABLE III

GENERATED HARDWARE SECURITY CONSTRAINTS CORRESPONDING TO SECURE FINGERPRINT TEMPLATE BASED ON ENCODING

For ‘0’			For ‘1’		
<0, 2>	<0, 398>	<2, 18>	<1, 3>	<1, 399>	--
<0, 4>	<2, 4>	--	<1, 5>	<3, 5>	<3, 215>
--	--	--	--	--	--

Module 4: Generating Fault Secured JPEG-codec Hardware Accelerator with Piracy Detective Control

- To enable the detective control against piracy for fault secured JPEG, encoded hardware security constraints are covertly implanted during register allocation module of behavioral synthesis.

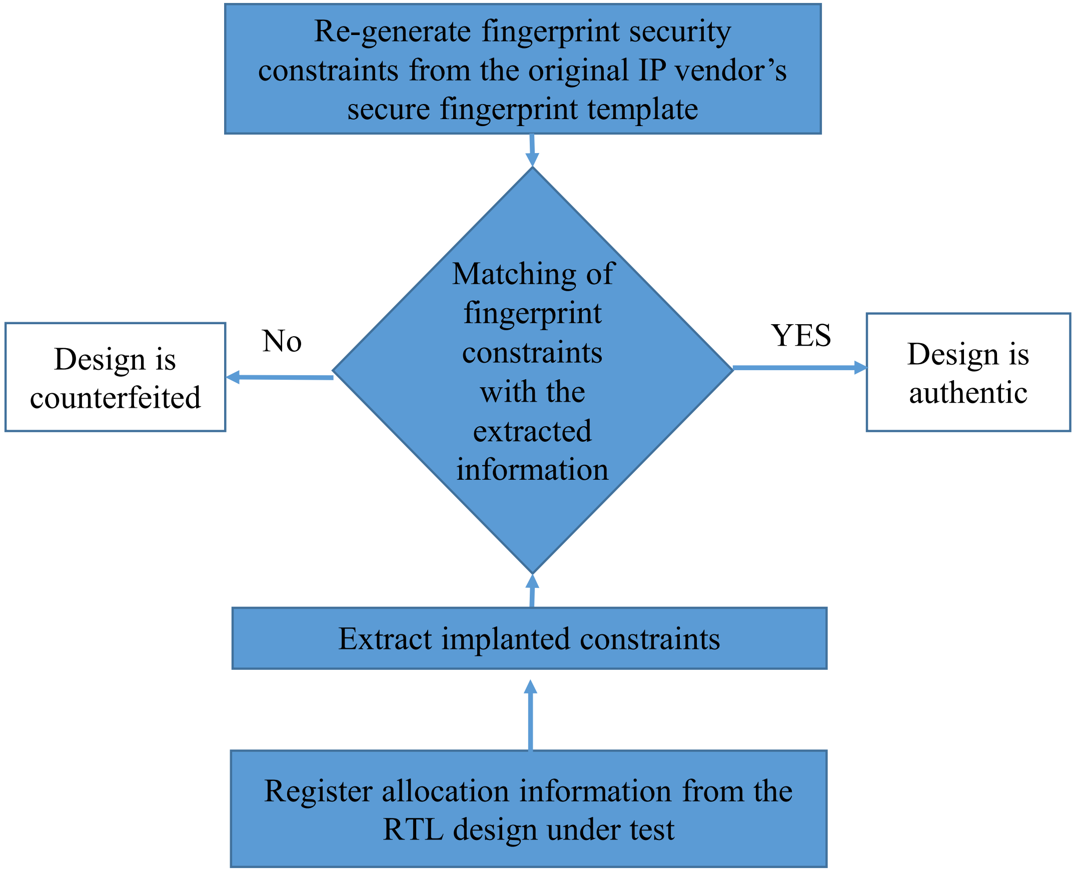

Control steps (CS1 to CS25)
Registers used (R_1 to R_{20})
Storage variables (0 to 399)

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
1	0	129	256	256	320	320	352	352	352	352	352	352	352	352	352	352	352	352	352	352	352	368	384	384	392	392
2	1	128	256	256	320	320	352	352	352	352	352	352	352	352	352	352	352	352	352	352	352	368	384	384	392	392
3	2	130	130	257																						
4	3	3	132																							
5	4	4	131	258	258	321																				
6	5	5	133																							
7	6	6	6	134	259																					
8	7	7	7	135																						
9	8	8	8	136	136	260	322	322	322	322	353	353	353	353	353	353	353	353	353	353	353	369				
10	9	9	9	9	137																					
11	10	10	10	10	138	261																				
12	11	11	11	11	139																					
13	12	12	12	12	12	262	262	262	323	323																
14	13	13	13	13	13	141																				
15	14	14	14	14	14	142	142	263																		
16	15	15	15	15	15	15	143																			
17	16	16	16	16	16	16	144	264	264	324	324	354	354	354	354	354	354	354	354	354	354	354	370	385		
18	17	17	17	17	17	17	145																			
19	18	18	18	18	18	18	18	146	265																	
20	19	19	19	19	19	19	19	147																		

Register allocation information of fault secured JPEG-codec post implanting encoded fingerprint security constraints. Note: the details of only 20 registers (out of 146) and 25 control steps (out of 54) have been presented (for the sake of brevity).

- Finally, the design with covertly implanted fingerprint security constraints is synthesized to generate secure register transfer level (RTL) datapath design.

Piracy Detection and Security Properties of Proposed Methodology



Re-generate fingerprint security constraints from the original IP vendor’s secure fingerprint template

Matching of fingerprint constraints with the extracted information

No

Design is counterfeited

YES

Design is authentic

Extract implanted constraints

Register allocation information from the RTL design under test

***Security Properties*:**

❑ The proposed methodology comprises of several security layers, such as:

- key set (α, β, γ),
- Number of minutiae points and
- Concatenation order of their features, concatenation order of IP vendor selected minutiae points for generating final secure fingerprint template,
- IP vendor chosen template size,
- Encoding algorithm and
- Ordering of storage variables for generating security constraints.

❑ All these security parameters are impossible to decode for an attacker to perform piracy evasion successfully

Results and Analysis

A. Security Strength Analysis:

➢ Robustness of the proposed methodology is analysed in terms of

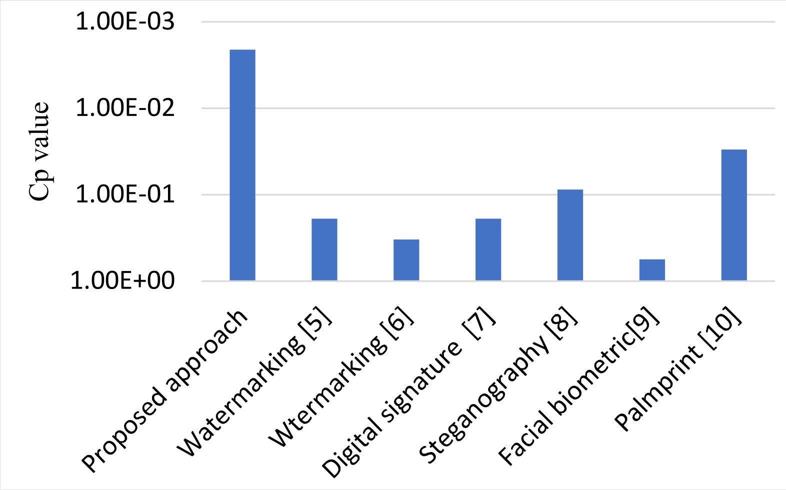
- Probability of coincidence ‘Cp’ and
- Tolerance against tampering attack using brute-force (A_T).

❑ The probability of coincidence (Cp), which is a measure of false positive, is computed as follows [5], [7], [8]:

$$\text{Cp}= \left(1 - \frac{1}{R_m}\right)^K \quad (2)$$

TABLE IV
Variation in Cp and TT for Proposed Approach

# Constraints (K)	Cp	TT
256	1.7E-1	1.15E+77
512	2.9E-2	1.34E+154
768	5.1E-3	1.55E+231
893	2.1E-3	6.60E+268



Cp comparison of the proposed approach with other techniques

‘**K**’: the number of implanted hardware security constraint and ‘$\mathbf{R_m}$’ : the numeral value of registers used in baseline

Proposed Methodology: Design Flow

- The effort required for an attacker in guessing the exact signature by performing brute force-attack is evaluated using the following metric [5], [7]:

$$A_T = F^K \qquad (3)$$

Where ‘$\mathbf{F^K}$’: the signature space and ‘**F**’ : the number of variables in fingerprint template

TABLE V
Comparison of Tamper Tolerance w.r.t related Works [5]-[10]

Proposed		Hardware watermarking [5]		Automatic signature insertion [6]		Digital signature [7]		Hardware steganography [8]		Facial biometric [9]		Palmprint security [10]	
#Security constraints (K)	**A_T**	**K**	**A_T**	**K**	**A_T**	**K**	**A_T**	**K**	**A_T**	**K**	**A_T**	**K**	**A_T**
256	1.1E77	120	1.3E36	40	1.0E12	30	1.0E9	125	NA	60	1.1E18	155	8.9E73
512	1.3E154	160	1.4E48	80	1.2E24	60	1.1E18	203	NA	81	2.4E24	182	6.8E86
768	1.5E231	200	1.6E60	120	1.3E36	120	1.3E36	317	NA	83	9.6E24	227	2.0E108
893	6.6E268	240	1.7E72	160	1.4E48	240	1.7E72	355	NA	84	1.9E25	231	1.6E110

B. Design Cost Analysis

Design cost is computed as follows [9], [5], [7], [8]:

$$C_d(R^c) = Q_1 \frac{\Psi_A}{\Psi_m} + Q_2 \frac{\omega_L}{\omega_m} \qquad (4)$$

TABLE VI
Fault Secured JPEG-CODEC Hardware Accelerator Design Cost Pre and Post Embedding Secure Fingerprint Template

Fingerprint template size	**# of registers in baseline**	**# of registers in fingerprint implanted design**	**Design cost of baseline**	**Design cost of fingerprint implanted design**	**% Cost overhead**
256 bits	146	146	0.2228	0.2228	0.00%
512 bits	146	146	0.2228	0.2228	0.00%
610 bits	146	146	0.2228	0.2228	0.00%
893 bits	146	147	0.2228	0.2229	0.04%

‘$\boldsymbol{R^c}$’ :the resource constraints, Ψ_A and ω_L:the area and latency of the design respectively, $\Psi_{\mathbf{m}}$ and $\omega_{\mathbf{m}}$:maximum area and design latency, $\mathbf{Q_1}$ and $\mathbf{Q_2}$: weighing factors for normalized area and design latency

Conclusion

- The high-level synthesis based design methodology to generate fault secured JPEG-codec hardware accelerator design has been presented.

- The proposed methodology exploits IP vendor’s fingerprint biometric information to generate secure fingerprint security constraints for enabling seamless piracy detection through the covertly embedded fingerprint constraints into the design during register allocation phase of HLS.

- Proposed methodology ensured more robust and seamless detective control against piracy at negligible design cost overhead.

References

[1] C. Pilato, S. Garg, K. Wu, R. Karri and F. Regazzoni, “Securing hardware accelerators: a new challenge for high-level synthesis,” *IEEE Embedded Syst. Lett.*, vol. 10, no. 3, pp. 77-80, Sept. 2018.

[2] W. Hu, C. -H. Chang, A. Sengupta, S. Bhunia, R. Kastner and H. Li, “An Overview of Hardware Security and Trust: Threats, Countermeasures, and Design Tools,” *IEEE Trans. Comput.-Aided Design of Integr. Circuits and Syst.*, vol. 40, no. 6, pp. 1010-1038, 2021.

[3] “Single event upsets,” Intel [online], Available: https://www.intel.com/content/www/us/en/support/programmable/support-resources/quality/seu.html, Nov. 2022.

[4] A. Sengupta, S. P. Mohanty, F. Pescador and P. Corcoran, “Multi-Phase Obfuscation of Fault Secured DSP Designs With Enhanced Security Feature,” *IEEE Trans. Consum. Electron.*, vol. 64, no. 3, pp. 356-364, Aug. 2018.

[5] F. Koushanfar, I. Hong, and M. Potkonjak, “Behavioral synthesis techniques for intellectual property protection,” *ACM Trans. Design Autom. Electron. Syst.*, vol. 10, no. 3, pp. 523–545, Jul. 2005.

[6] E. Castillo, L. Parrilla, A. Garcia, U. Meyer-Baese, G. Botella and A. Lloris, “Automated Signature Insertion in Combinational Logic Patterns for HDL IP Core Protection,” *2008 4th Southern Conference on Programmable Logic*, Bariloche, Argentina, 2008, pp. 183-186.

[7] A. Sengupta, E. R. Kumar and N. P. Chandra, “Embedding digital signature using encrypted-hashing for protection of DSP cores in CE,” *IEEE Trans. Consum. Electron.*, vol. 65, no. 3, pp. 398-407, Aug. 2019.

[8] A. Sengupta and M. Rathor, “IP core steganography for protecting DSP kernels used in CE systems,” *IEEE Trans. Consum. Electron.*, vol. 65, no. 4, pp. 506-515, 2019.

[9] R. Chaurasia and A. Sengupta, “Symmetrical Protection of Ownership Right's for IP Buyer and IP Vendor using Facial Biometric Pairing,” *2022 IEEE International Symposium on Smart Electronic Systems (iSES)*, Warangal, India, 2022, pp. 272-277.

[10] A. Sengupta, R. Chaurasia and T. Reddy, “Contact-Less Palmprint Biometric for Securing DSP Coprocessors Used in CE Systems,” *IEEE Trans. Consum. Electron.*, vol. 67, no. 3, pp. 202-213, Aug. 2021.

[11] V. K. Alilou, “FingerPrint Matching: A simple approach MATLAB Central File Exchange,” [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/44369- fingerprint-matching-a-simple-approach, last accessed on dec. 2022.

[12] S. S. Ali, V. S. Baghel, I. I. Ganapathi, S. Prakash, “Robust biometric authentication system with a secure user template,” *Image Vis. Comput.*, vol. 104, pp.104004, 2020.

[13] 15 nm open cell library. [Online], Available: https://si2.org/open-cell-library/, last accessed on Aug. 2022.

[14] AMD Xilinx Versal Core. [Online], Available: https://www.xilinx.com/products/silicon-devices/acap/versal-ai-core.html, last accessed on July, 2023.

Thank You