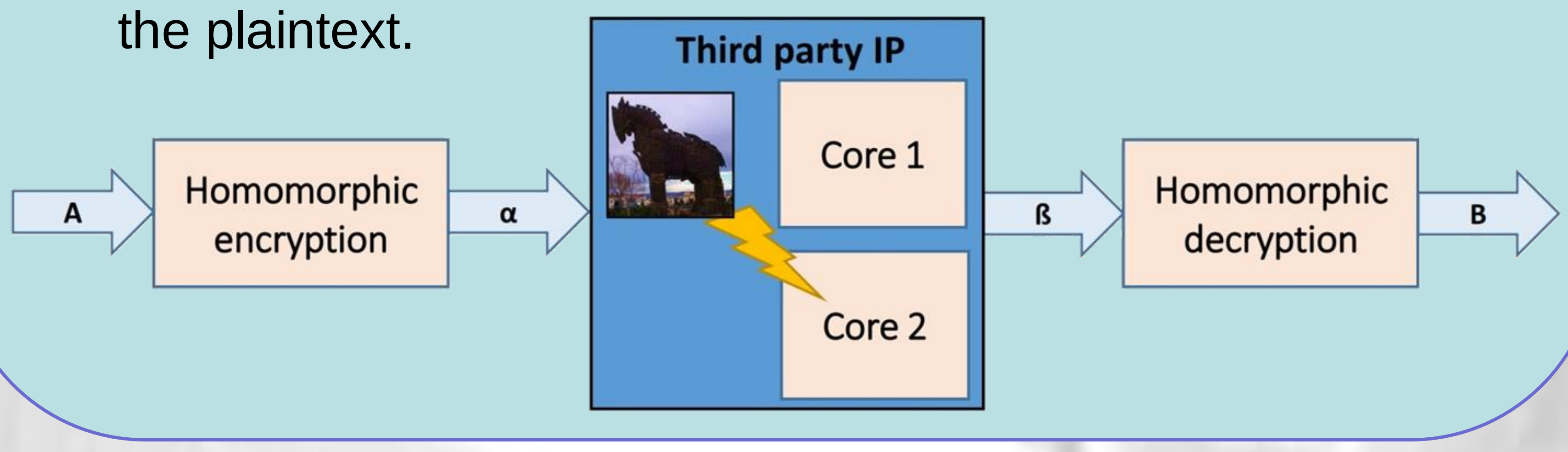
Homomorphic Data Isolation for Hardware Trojan Protection

M. Tarek Ibn Ziad*, Amr Alanwar**, Yousra Alkabani*, M. Watheq El-Kharashi*, and Hassan Bedour*

*Department of Computer and Systems Engineering, Ain Shams University, Cairo, Egypt
**Department of Electrical Engineering, UCLA, Los Angeles, CA, USA
Email: mohamed.tarek@eng.asu.edu.eg, alanwar@ucla.edu,
{yousra.alkabani, watheq.elkharashi, hassan.bedour}@eng.asu.edu.eg

Introduction

- Maintaining technology secrets of the fabrication facilities and design royalties of third party IP owners raises the difficulty of Hardware Trojan detection and protection.
- Homomorphic encryption may be used to solve this issue and defeat Hardware Trojans.
- Homomorphic encryption is a type of encryption, which allows specific types of operations to be carried out on ciphertext and generates an encrypted result which, when decrypted, matches the result of operations performed on the plaintext.



Background

- **Partial Homomorphism (PH)**
  - It offers the ability to perform a certain type of operations, addition or multiplication, on ciphertexts without revealing data.
  - Multiplicative homomorphic scheme:
    $$E(m_1) \times E(m_2) = E(m_1 \times m_2)$$
  - Additive homomorphic scheme:
    $$E(m_1) \times E(m_2) = E(m_1 + m_2)$$

- **ElGamal Scheme**
  - It is a multiplicative homomorphic scheme. In order to illustrate its functionality, let us consider the secret key (k) and the public key (g, h), $where\ h = g^k\ mod\ n$
  - **Encryption:** $C_1 = g^l\ (mod\ n)$ and $C_2 = h^l \times m\ (mod\ n)$
  - **Decryption:** $m = C_1^{-k} \times C_2\ (mod\ n)$
  - If (x_1, y_1) and (x_2, y_2) are valid encryptions for m_1 and m_1, with the same key, then $(x_1 x_2,\ y_1 y_2)$ is a valid encryption of $m_1\ m_2$.

- **CRT-based ElGamal (CEG) Scheme**
  - It is an additive homomorphic scheme that uses the Chinese Remainder Theorem (CRT).
  - **Encryption:** $C_1 = g^{l_i}\ (mod\ n)$ and $C_2 = h^{l_i} \times g^{m_i}\ (mod\ n)$

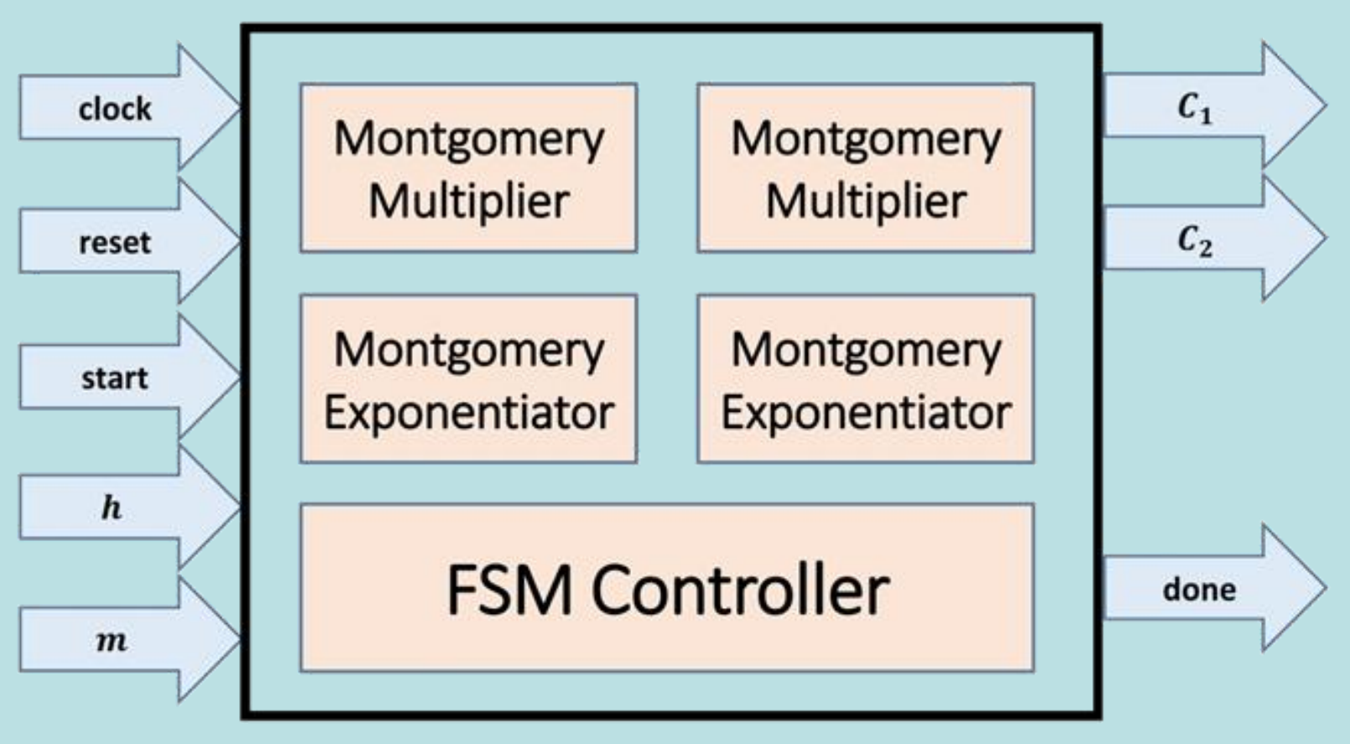
  where $m_i = m\ (mod\ d_i),\ d_i\ is\ a\ random\ number,$
  $i = 1, \ldots, t$ and $\gcd(d_i, d_j) = 1\ for\ i \neq j$
  - **Decryption:**
    $$m = \mathrm{CRT}^{-1}[(\log_{\mathrm{g}}\left(\mathrm{C}_{2_\mathrm{i}} \times \mathrm{C}_{1_\mathrm{i}}^{-\mathrm{k}}\ (\mathrm{mod\ n})\right), i = 1, \ldots, t)]$$
    $$\mathrm{CRT}^{-1}[C_i] = \sum_{i=1}^{t} C_i \frac{d}{d_i}\left(\frac{d}{d_i}^{-1}\ mod\ d_i\right) mod\ d$$

Major Contributions

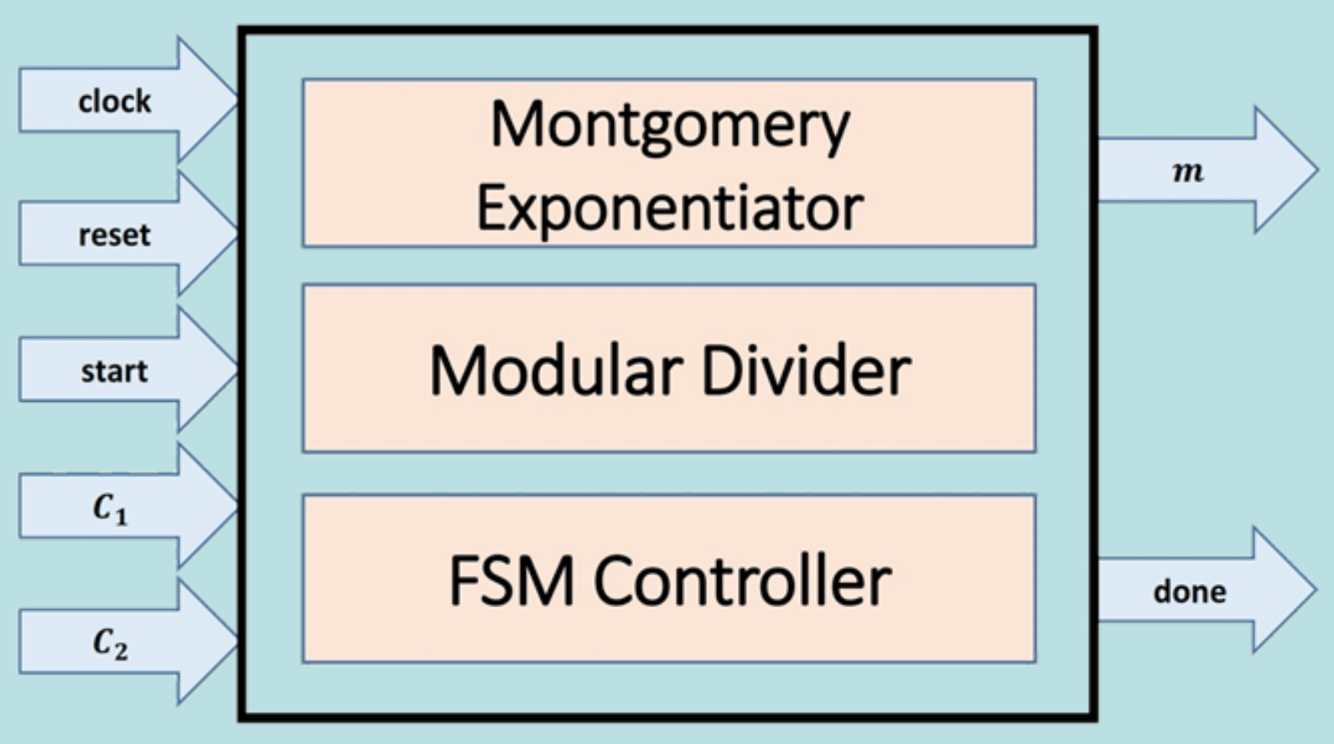
- ✓ Discussing new ideas to have a blind data processing by the third party IP with a minimum cost.
- ✓ Implementing ElGamal scheme, which is multiplicative homomorphic and the CEG scheme, which is additive homomorphic, on a low-cost FPGA and showing the resource utilization, performance, and power analysis.
- ✓ Introducing a dual-circuit design (Dual ElGamal) that supports both, multiplicative and additive homomorphic properties and providing the obtained savings on area and power over a regular design that has no resource sharing.

Hardware Trojan Protection Using PH

- **PH Support**
  - ElGamal Scheme Implementation

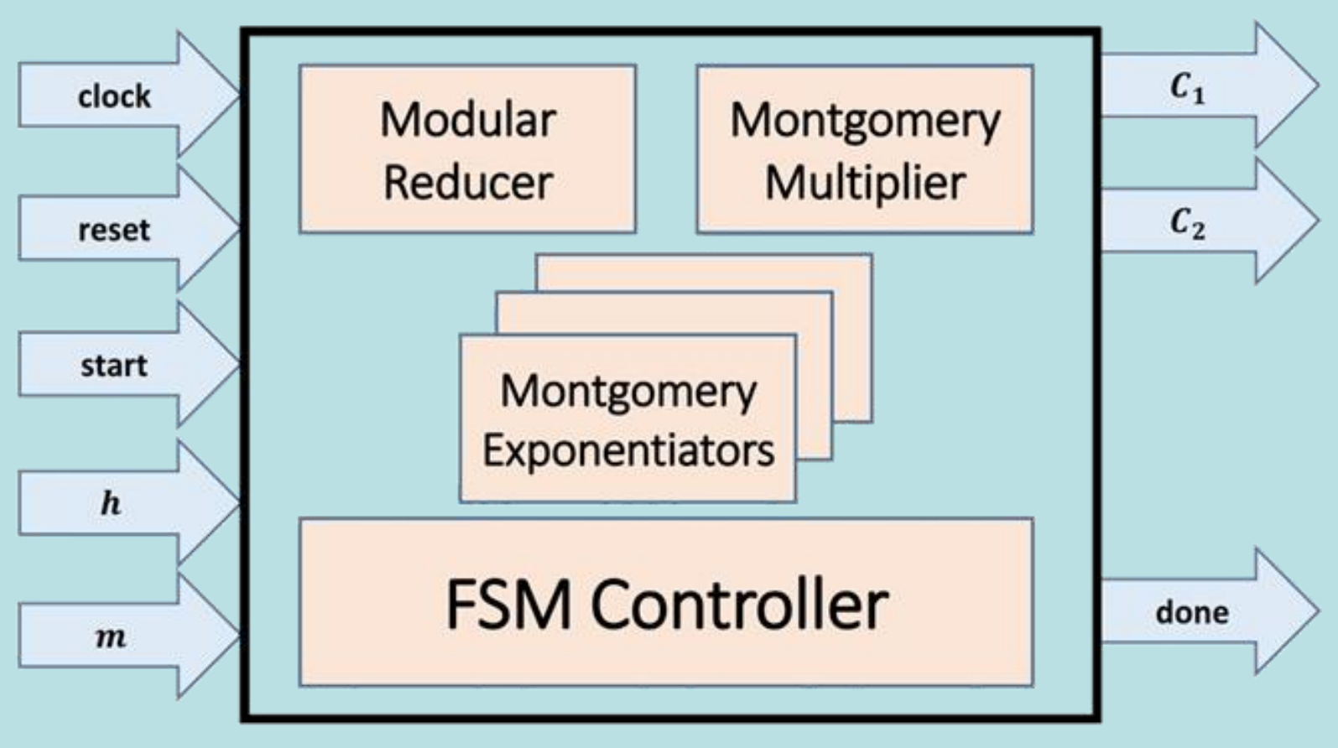


(a) Encryption.

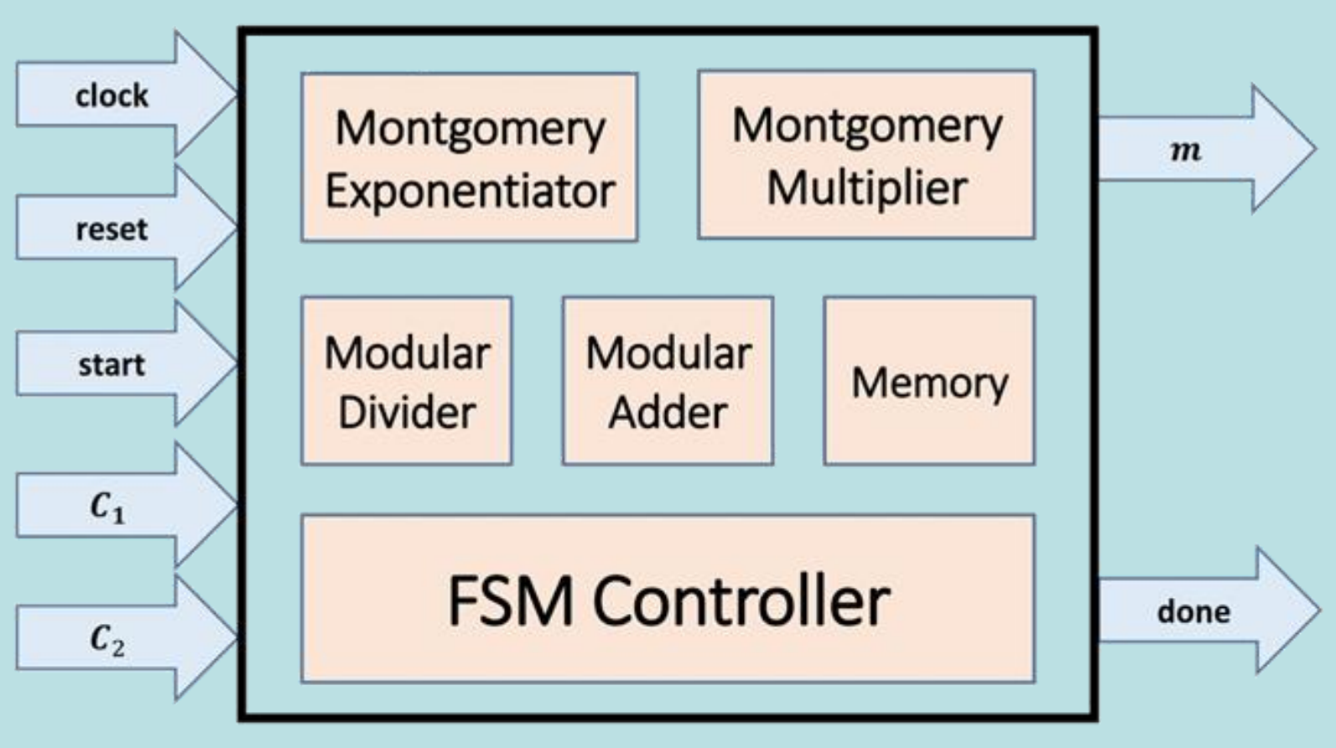


(b) Decryption.

  - CEG Scheme Implementation



(a) Encryption.



(b) Decryption.

- **Dual ElGamal Design**
  - Some third party IPs require the usage of more than one single type of operation. Ex: an ALU that uses a selection line to switch its mode between two different operations.
  - Instead of implementing two different schemes, we suggest a solution by combining the two previously schemes, ElGamal and the CEG, in a single dual-circuit design. Thus, the proposed design supports both additive and multiplicative homomorphism.

Experimental Results

Comparing our Dual ElGamal design to the Regular ElGamal design for k = 8 bits.

- **Area reduction**

	Encryption			Decryption		
	Regular ElGamal	Dual ElGamal	Area reduction (%)	Regular ElGamal	Dual ElGamal	Area reduction (%)
Registers	909	635	30.14	536	364	32.09
LUTs	1137	735	35.36	626	457	26.09
BRAMs	0	0	00.00	1	1	00.00

- **Timing comparisons**

	Encryption		Decryption	
	Regular ElGamal	Dual ElGamal	Regular ElGamal	Dual ElGamal
Frequency (MHz)	161.277	158.51	117.099	121.344
Used cycles	651	662	665	665

- **Power consumption (mW)**

	Encryption		Decryption	
	Regular ElGamal	Dual ElGamal	Regular ElGamal	Dual ElGamal
Dynamic	54.44	30.03	26.77	16.52
Leakage	65.00	65.00	65.00	64.00
Total power	119.44	95.03	91.77	80.52

Conclusions

- ✓ As PH is sufficient enough with some third party IPs, we implemented two designs that supports PH (multiplicative only and additive only) based on ElGamal encryption /decryption scheme.
- ✓ We integrated the two designs together and implemented a dual-circuit design on a Xilinx Spartan-6 FPGA. The design saved **35%** of the logic resources and **20%** in power compared to a regular design that combines two IPs, one for ElGamal and another for CEG, without any resource sharing between them.