ICtoken: An NFT for Hardware IP Protection

Shashank Balla
University of California, San Diego
San Diego, California, USA
sballa@ucsd.edu

Yiming Zhao
University of California, San Diego
San Diego, California, USA
yiz060@ucsd.edu

Farinaz Koushanfar
University of California, San Diego
San Diego, California, USA
farinaz@ucsd.edu

ABSTRACT

Protecting integrated circuits (ICs) from piracy and theft throughout their lifecycle is a persistent and complex challenge. In order to safeguard against illicit piracy attacks, this work proposes a novel framework utilizing Non-Fungible Tokens (NFTs) called ICtokens, uniquely linked to their corresponding physical ICs. Each ICtoken contains comprehensive information, including authentication data, supply chain stage and status, ownership details, and other IC metadata, while also making provision for the secure integration of a logic-locking key. Designed to be publicly logged, ICtokens securely obscure metering information without compromising functionality. In addition, the ICtracker, a distributed ledger technology powered by a swift and energy-efficient consortium blockchain, is used to register and manage ICtokens and their respective owners, tracking all associated interactions. This robust ledger guarantees the traceability and auditing of ICtokens while simultaneously developing a product-level NFT at every transaction point within the supply chain. Consequently, a scalable framework is established, creating unique, immutable digital twins for ICs and IC-embedded products in the form of ICtokens and their transactions. This provides a robust and reliable supply chain trail back to the original IP owner, while also offering unprecedented assurance to consumers of IC-embedded products. The rich information contained within ICtokens facilitates more detailed audits than previous proposals for IC supply chain monitoring. A proof-of-concept, implemented as an open-source solution, ensures the ease of adoption of the proposed framework.

CCS CONCEPTS

• **Hardware → Integrated circuits**; • **Security and privacy → Security in hardware**.

KEYWORDS

IC Piracy, Supply Chain, Blockchain, Non-Fungible Token, PUF, Logic Locking.

ACM Reference Format:
Shashank Balla, Yiming Zhao, and Farinaz Koushanfar. 2018. ICtoken: An NFT for Hardware IP Protection. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation emai (Conference acronym 'XX)*. ACM, New York, NY, USA, 7 pages. https://doi.org/XXXXXXX.XXXXXXX

Conference acronym 'XX, June 03–05, 2018, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/18/06...$15.00
https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Today, the microscopic size and negligible cost of ICs has enabled the addition of logical components to many devices, making them *smart*. A plethora of new businesses creating niche products have come up that require rapid prototyping of their products. To cater to this incredible demand the semiconductor industry has adopted a horizontal supply chain model. During the lifetime of an IC, multiple parties take up distinct roles; designers outsourcing fabrication, distributors stocking up/reselling, assemblers/integrators sourcing components, and original equipment manufacturers building the end products. Unfortunately, this highly distributed setting has also opened up many avenues for IP theft [9] [Figure:1]. Recently, the global shortage of chips has exacerbated the issue, where bootleggers pirate name-brand designs, claim high-quality assurances [25]. The infiltration of counterfeit products into the supply chain puts end-consumers and critical infrastructure at high risk [26], [27].

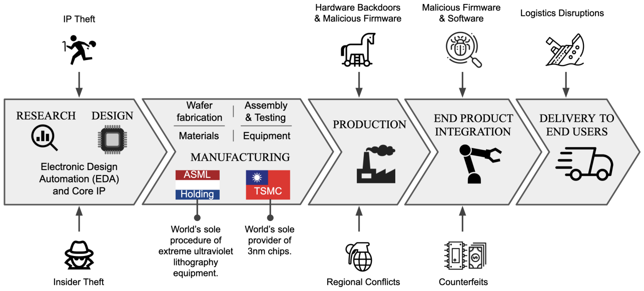

Figure 1: Supply chain threats. [9]

Many techniques have been proposed by the research community to tackle the various threats of hardware piracy [28]. Rostami et al [29] and Guin et al [15] survey distinct attacks at each stage of the supply chain for various threat models. For each of these attacks, they discuss methods for detecting it as well as techniques to prevent such an attack. The main limitation of these physical detection methods is that the application of just one is not sufficient to detect all attacks. Further, most of them destroy the Device-Under-Test in addition to being very laborious and requiring very costly specialized equipment. These techniques are impractical for businesses to deploy and impossible for individuals without the necessary tools. However, some of the prevention techniques, as discussed below, have seen commercial use with IP owners protecting their IPs to a substantial extent with the foundries.

Logic locking is a pre-silicon hardware IP protection technique that obfuscates the functionality of an IC by design until a correct key (known only to the IP owner) is provided along with the input [38] , [19]. Thus, even in fabrication, malicious actors who have physical access to the chip will not obtain the correct input-output behavior of the IC, as, at that point, only the IP owner has the key. Though this approach offers an effective IP protection mechanism against overbuilding and reverse engineering, it does not consider the post-fabrication testing of the die required for quality assurance purposes before it is packaged and released. Secure Split-Tests (SSTs) help overcome this challenge [8]. SSTs augment logic locking by making provisions for testing on the locked designs. This process requires rounds of communication between the IP owner and fabrication house to test the functionality of chips. A particular die is approved for packaging only after the IP owner is satisfied with the functionality.

For post-silicon fingerprinting and authentication, Physical Unclonable Functions (PUFs) is tried-and-tested technology [21], [17], [2]. PUFs leverage random process variations at the fabrication stage to generate unique in-borne identifiers for the IC. These identifiers can then be used for authentication and attestation of the IC by the device owners and guard against cloning [16]. The above two approaches, Logic-locking and PUFs, can be fused together to create device-unique keys that can obfuscate the functionality of an IC [30]. Techniques that create device-specific unique identifiers which can also be used to obfuscate functionality of the IC are referred to as Active Metering [20].

Such methodologies do provide protection against piracy attacks, but they do not encompass the integrity of ICs throughout the entire supply chain. Hence, there is a need for a supply-chain-wide foolproof framework that builds a guarantee in the product as it evolves through various transactions with the full knowledge of the IP owner. In this context, developing trustworthy transactions throughout the supply chain becomes the bedrock of a secure framework for both IP protection as well as a guarantee to the end-user. For a trustworthy transaction on any product, the buyer must be able to verify the authenticity of the product and the seller must be able to ensure licensed use of IP within the product among other things. A publicly accessible database incorporating relevant metadata of ICs as updated through transactions would help to authenticate registered products and detect any discrepancies. Blockchains are a popular approach to creating immutable public distributed ledgers and have been widely adopted by many businesses to offer supply chain monitoring services [24], [39], [31].

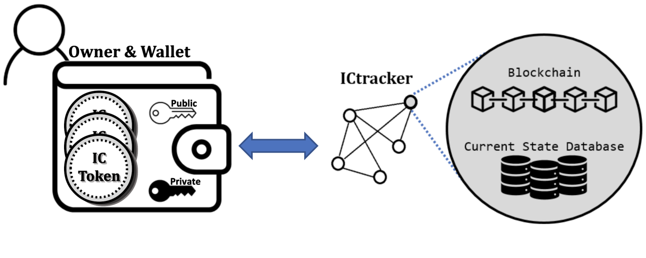

Figure 2: Components of the proposed framework.

This paper suggests ICtoken, a novel framework (Figure 2) to enable provable owner tracking of the ICs through the long supply chain. The contributions are as follows:

- Introduction of ICtoken, assigning an NFT per IC. The individual NFTs can be leveraged to create product-level digital assets.
- Provisions for secure transfer of IC specific information (e.g. active metering information) from one owner onto the next while ensuring privacy and tamper-resistance.
- Introduction of ICtracker, an essential block-chain component of the framework which can be utilized to trace ICs through its product life cycle.
- The robustness of ICtoken framework is demonstrated through analysis of the block-chain based method against various contemporary threats.

We review the related background underlying our motivations in Section 2. In Section 3, we describe in depth our framework with ICtokens and the ICtracker. Finally, we evaluate our framework by showing its resistance to various IC supply chain threats in Section 4.

2 BACKGROUND

Blockchain technology is a path breaking approach of recent times with revolutionary applications in several industries [24], [33], [32]. Blockchain is an immutable decentralized hash chain that stores information in a linked list of blocks which is mirrored on all nodes participating in the network. Every block is linked to the previous one by carrying a cryptographic hash of all the data stored in the previous one. Information is stored on the blockchain in its header field and a consensus protocol ensures that this information is mirrored across all nodes in the network. Bitcoin [23] was the first application to popularize the use of blockchains to log transactions on digital currency. Later applications also allowed more types of information to be stored on this database. Ethereum [5] introduced smart contracts which are Turing complete programs that can be stored on the blockchain, immutably binding the contracted parties to the terms of the contract. Four years ago, the introduction of NFTs added further impetus to the use of blockchains. They enable users to digitize their assets by pushing a digital signature of it onto a blockchain thereby establishing the signing entity's ownership [36]. Ethereum first proposed the design of NFTs in an Ethereum Improvement Proposal (EIP) [11]. This proposal introduced the ERC 721 standard [12] to define the interface and functionalities a smart contract must employ to be able to mint NFTs. This effectively created a marketplace for digital assets to be traded on a blockchain where original creators could also get paid royalties every time their NFT was traded [34].

One of the initial works that considers blockchains for IC supply chain monitoring is Islam et al [18]. They propose to use a smart contract deployed on a blockchain which has provisions for enrolling an IC, authenticating it and transferring its ownership. The information they log for every IC is a unique identifier, its current owner and PUF-derived challenge and response pairs (CRPs). The storage of CRPs on the blockchain, including it in every transaction, increases the size posing scalability issues. Chaudhary et al [6] aim to mitigate this issue by exporting the CRPs to an off-chain system called InterPlanetary File System (IPFS) [4] that exists as a storage oracle for the blockchain to access and retrieve files. However, [18] and [6] consider only one stage of production, that is, system integration, and do not cover the prior stages of production.

Cui et al [10] propose a two-step transaction process to eliminate errors that cause asset transfer to a wrong address and inadvertently risk losing proprietorship of the asset, namely the IC. But they do not track the production stages or the work that was done on the IC while it was in the hands of each of its owners. Xu et al [37], Vosatka et al [35] propose to start tracking right after fabrication and assume active metering techniques for protection against fabrication level counterfeiting attempts. However, they make no provision for saving this information at the blockchain requiring all future owners to interact with the IP owner to retrieve this.

Concurrently, there has been progress in the domain of hardware implementations for improved IP integrity for IC designers. Logic locking is a prominent technology, where the latest advances give cryptographic security guarantees. Beerel et al [3] lay down mathematical security definitions for logic locking. Chhotaray et al [7] and Ganji et al [14] propose cryptographically secure logic locking schemes. Further, [14] leverage blockchain technology to register gate-level representations of an IC as IP blocks and enable pay-per-use evaluations. Specifically, they allow limited number of accesses to the IC while all previous schemes lose the effectiveness of the logic locking key upon first access of the IC.

The proposed framework addresses the shortcomings of previous works while integrating the latest developments to strengthen IP integrity for IC designers across the product life cycle. It enables registration, authentication, active metering and end-to-end tracing in a privacy-preserving fashion, duly implemented on a transparent blockchain base.

3 FRAMEWORK DESIGN

We model the complete IC supply chain with three distinct constituent parts. ICtokens represent the merchandise; ICs, Printed Circuit Boards (PCBs) and Electronic Devices. Owners are the transactors of this merchandise and therein the ICtokens. Lastly, ICtracker is the encompassing protocol and moderation system for all transactions.

3.1 ICtoken

Here we introduce the design template for constructing an ICtoken i.e. an NFT uniquely bound to a physical IC capturing all movements throughout its life cycle. The metadata of the ICtoken has all the necessary information to achieve this. To ensure binding, we must tag ICtoken with the same identifier as the physical IC; and for this purpose we include *ICID*, a SHA256 hash of the unique identifier for the IC. Further, as the IC progresses through the supply chain, it might get built into a PCB, which can then be uniquely identified by the set of ICs embedded in it. A unique identifier for the PCB, a *PID*, can be constructed from a merkle hash of the *ICIDs*. To tag the ICtoken to its PCB, the *PID* is included in its metadata as well. Similarly, when this PCB gets built into an electronic device (system), it can further be uniquely identified by the set of PCBs used in it and an *EDID* can be constructed from a merkle hash of their *PIDs*. To tag the ICtoken with the electronic device, the *EDID* is also stored in the metadata. To capture the markings on the IC's package, a SHA256 hash of its markings, *markHash*, is included. The production *stage* (1: Fabrication; 2: PCB Assembly; 3: System Integration; 4: End-User) and *status* (0: In progress; 1: Completed) of the IC are included to track its progression in the supply chain. For provenance, *prevVer*, a linker to the previous version of the ICtoken, and *version*, the current version number of the ICtoken are also included. Finally, a flag, *isDefective*, is present to notify if the IC has been reported to be malfunctioning.

Table 1: ICtoken Size Analysis

Attributes	**Data**	**Size**
Metadata	*ICID, PID, EDID*	3 × 32B
	markHash	32B
	Stage, Status	3b, 1b
	prevVer, version	8B, 1B
	isDefective	1b
ICkey	*keyEncr*	256B
	keyHash	32B
Owner	*publicID*	32B
trnsaxnID		256B
Total		**714B**

By design, ICtoken also has provisions to securely carry the *key* for active metering. The ICkey attribute holds an encryption of the *key* under the current owner's public key, *keyEncr*, and an SHA256 hash of the *key*, *keyHash*, to verify correct decryption as well as to ensure a consistent trail with previous versions. More generally , the ICkey attribute is provisioned for in the template to securely store any information even if active metering is not employed. Proprietorship information is tracked through the owner attribute which carries the *publicID* of the current owner. And lastly, the *transaxnID* attribute is the owner's digital signature on all the above information, binding it as the source. Table 1 analyses the total size of an ICtoken object.

3.2 Owner

An Owner represents a participant in the IC supply chain, and can be any entity that possesses or wishes to possess an IC at any stage in its life cycle. These prospective owners could be hardware designers (IP holders), fabrication units, PCB assemblers, system integrators, end users, or electronics recyclers, coming in at different production/use stages of an IC, as well as any intermediate distributors/resellers. Thus, the variety of possible owners considered is comprehensive, which ensures every IC explicitly has an owner at any instance in its life-cycle. Every owner requires a unique *publicID* and a *public-private key pair* to participate in the framework's protocol. The *publicID* is used for identification and the *public-private key pair* is used towards establishing secure communication and transactions. The *public-private key pair* enables the following functionalities for the owner:

- `encrypt` and `decrypt`: Encrypts/decrypts the input with the *public*/*private key*.
- `changeEncKey`: Changes the encryption key of the input to a *public key* of another owner; decrypt input and encrypt result with new owner's *public key*.
- `signMessage`: Creates a digital signature of a message with the *private key*.
- `verifySign`: Verifies a digital signature with the *public key*.

Every Owner has a *wallet* that authorizes its interface with the blockchain and stores all owned ICtokens. The *wallet* stores the *publicID*, the *public-private key pair* of the owner, and a public profile of the owner consisting of the *publicID*, *public key* attributes and the `verifySign` functionality. The *wallet* uses the public profile to enroll with the blockchain and authorize transactions to/from other wallets on the blockchain.

3.3 ICtracker

ICtracker is a blockchain-based public distributed ledger that keeps track of enrolled ICs throughout their product life cycle by logging all transactions of ICtokens. It maintains a database of the current state of the blockchain, public profiles of all enrolled owners and the ICtokens they currently own. ICtokens are logged as transactions in our blockchain. Since every block is of limited size it can store only a fixed number of ICtokens. ICtracker receives transactions from a wallet in the form of new ICtokens while requesting for a particular service. ICtracker will perform the necessary checks to ensure that the new ICtokens have only those changes that are permitted by the service requested. If these requirements are met, ICtracker adds the new ICtokens to the latest block and updates its current state database when the addition is committed. In the distributed setting, this protocol will run on every node in the network, and any transactions of ICtokens will require a consensus across all nodes in the network. Every node stores a local copy of the latest committed state of the blockchain along with four mappings in its database that keeps up with this state. These mappings are paramount for authentication and verification of transactions and overall efficiency of the protocol. The mappings are as follows:

ICdb : *ICID* ⟶ latest index in the blockchain.
PCBdb: *PID* ⟶ list of *ICIDs*
DEVdb: *EDID* ⟶ list of *PIDs*
OWNdb: *publicID* ⟶ {*isEnrolled, pubProf, assets*}

ICdb quickly identifies the index of a particular *ICID* on the blockchain. Specifically, it retrieves the latest version of the ICtoken for an *ICID* on the blockchain. It is useful for auditing, i.e., when someone wishes to verify the most up-to-date information for a particular IC. Moreover, since the ICtoken also stores the index of its previous version, it is simple to trace back the entire history of the IC. PCBdb and DEVdb keep track of the PCB-level and System-level tokens, i.e., collections of the ICtokens used in the device. OWNdb stores the information of enrolled owners, their public profile and all assets (ICs, PCBs and Devices) currently associated with their account. It makes for quick verification and authentication of owners on the node's end and faster updates for owners' wallets to find all the latest ICtokens linked to their accounts.

ICtracker provides the following services for owners to transact on ICtokens.

- `enrollOwner`: Lets owners enroll themselves as authorized users by storing their data in OWNdb.
- `verifyTransaxn`: Verifies the authenticity of the transaction by validating the owner's signature in the ICtoken's *trnsaxnID* with the owner's *public key*. It aborts the service requested if verification fails.
- `enrollIC`: Lets enrolled owners register new ICs into ICtracker for the first time. [Algorithm 1]
- `reportDefective`: Lets users report malfunctioning ICs. ICtracker verifies the transaction and sets the *isDefective* bit of the ICtoken if successful.
- `updateStage`: Lets owners update the *stage* or *status* of a single ICtoken. [Algorithm 2]
- `updatePIDorEDID`: Lets owners update the *PID* or *EDID* of multiple ICtokens. [Algorithm 3]
- `transferIC`: Lets owners transfer their ICtoken to the next owner. [Algorithm 3]

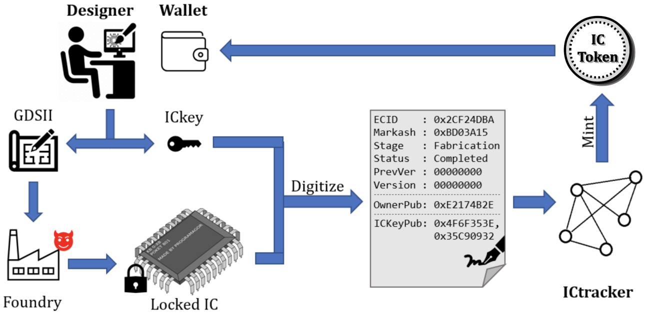

Figure 3: IC enrollment.

Algorithm 1 Enrolling an IC in ICtracker

```
procedure enrollIC(ICtoken):
    set ICID = ICtoken.metaData.ICID
    assert ICdb[ICID] == None                          ▷ can't re-enroll
    set ownID = ICtoken.owner.publicID
    verifyTransaxn(OWNdb[ownID], ICtoken)
    assert ICtoken.metaData.stage == Fabrication
    assert ICtoken.metaData.status == Completed
    assert ICtoken.metaData.PID == None
    assert ICtoken.metaData.EDID == None
    assert ICtoken.metaData.prevIdx == None
    assert ICtoken.metaData.version == 0
    add ICtoken to blockchain                    ▷ Data matches a new IC
    update ICdb, OWNdb with latest ICtokens and assets.
```

4 PROTOCOL ANALYSIS

4.1 Feasibility analysis

We have made two open-source implementations for the framework proposed in the paper [1]. The first one is an interactive python notebook where all components of the framework i.e. ICtokens, Owners, Wallets, ICtracker, and all their functionalities as proposed are codified. The python notebook simulates the complete protocol for the interface between owners' wallets and a node in the ICtracker network. Thus, it serves as an accurate blueprint for establishing the necessary blockchain infrastructure required for the framework from the ground up. The second implementation is targeted towards deploying the framework on pre-existing infrastructure. Here we developed a smart contract for ICtracker in solidity programming language [13] which is supported by various

Algorithm 2 Updating an IC's stage/status in ICtracker

```
procedure updateStage(ICtoken):
    assert ICID of ICtoken is enrolled and not defective.
    set metaData = ICtoken.metaData
    fetch prevToken from ICdb
    verifyTransaxn(OWNdb[prevToken.owner], ICtoken)
    set prevData = prevToken.metaData  ▷ previous metadata
    assert metaData == prevData other than stage, status
    assert metaData.stage >= prevData.stage
    if metaData.status < prevData.status then
        assert metaData.stage > prevData.stage
    ICtoken.version += 1                  ▷ increment version
    set ICtoken.prevVer = ICdb[ICID]    ▷ add link to previous
    add ICtoken to ICtracker.blockchain
    update ICdb, OWNdb with latest ICtokens and assets.
```

Algorithm 3 Updating PID/EDID of multiple ICs in ICtracker

```
procedure updatePIDorEDID(ICtokens):
    assert all ICtokens' ICIDs are enrolled and not defective.
    fetch prevTokens of all ICtokens
    for every ICtoken in ICtokens do ▷ check all owners same
        verifyTransaxn(previous owner, ICtoken)
    assert ICtokens == prevTokens other than PID, EDID.
    assert prevTokens' status is Completed.
    if all ICtokens have same PID then            ▷ update PID
        assert all ICtokens have stage == PCB Assm
        assert all ICtokens have EDID == None
        assert all prevTokens have PID and EDID == None
        PCBdb[PID] = list(ICIDs)                ▷ update PCBdb
    else if all ICtokens have same EDID then ▷ update EDID
        assert PID of all ICtokens, prevTokens is same
        assert all ICtokens have stage == Sys Int
        assert all prevTokens have EDID == None
        DEVdb[EDID] = list(PIDs)                ▷ update DEVdb
    else EXIT                                      ▷ invalid data
    for every ICtoken in ICtokens do
        ICtoken.version += 1                ▷ increment version
        add ICtoken to ICtracker.blockchain
        update ICdb, OWNdb with latest ICtokens and assets.
```

Algorithm 4 Transferring IC ownership in ICtracker

```
procedure transferIC(ICtoken):
    assert ICID of ICtoken is enrolled and not defective.
    assert ICtoken.owner is enrolled                 ▷ new owner
    fetch prevToken for the ICID
    assert prevToken's status is Completed.
    verifyTransaxn(prevToken.owner, ICtoken)
    assert ICtoken == prevToken other than key, owner
    ICtoken.version += 1                  ▷ increment version
    add ICtoken to ICtracker.blockchain
    update ICdb, OWNdb with latest ICtokens and assets.
```

prominent blockchain platforms [5], [22]. We compile and deploy this as an application on the Ethereum blockchain. Once deployed, this smart contract can mint ICtokens, as presented in the paper, that are compliant with the ERC721 standard [12]. It also enables similar transactions on ICtokens by providing all the services that were proposed to be offered by ICtracker in the paper.

Our implementations serve as a concrete proof-of-concept for the proposed framework by showing all the necessary functionalities and infrastructure required to solve the problem of hardware IP protection in a globalized supply chain and its feasibility of providing these functionalities through implementation on a pre-existing general-purpose public blockchain.

4.2 Reliability analysis

Here we analyse how our protocol ensures reliability by incorporating the following design principles in all of its functionalities.

4.2.1 *Authenticity.*

Any service performed by ICtracker first begins with verifying the authenticity of the transaction. This is done by verifying the ICtoken's transaction ID, *trnsaxnID*, which is a digital signature of the owner on all the data stored in the ICtoken. It is verified with the Owner's *public key* that is stored in ICtracker's database. This ensures that only authenticated and registered users/ owners of the ICtoken are permissioned to change any data stored on the ICtoken. Furthermore, an ICtoken can only be transferred to an enrolled owner.

4.2.2 *Access control.*

The active metering information for the IC is very confidential information. This must be accessible only to the current owner. To ensure this, ICtokens by design only store an encryption of this information under the current owner's *public key*. Furthermore, the presence of a cryptographic hash of the metering information ensures that it has not been tampered with.

4.2.3 *Component privacy.*

For the physical IC, all its metadata is stored as a cryptographic hash in its ICtoken. Pre-image resistance of the hash function employed guarantees that the original information cannot be duplicated or spoofed simply by viewing the contents on the ICtoken. Furthermore, it also ensures that any entity even with full access to all information stored in ICtracker cannot learn anything about ICs that are not in its possession.

4.2.4 *Device Integrity.*

The framework also generates device level tokens as a composition of the basic ICtokens built into it. The identifiers for these devices are generated from a merkle hash of all the ICtokens integrated into it. This ensures swapping one component with another will be impossible without causing a change to the device identifier. Moreover, once a device identifier is loaded into the ICtoken for the first time, it can never be modified. This ensures that reuse of ICs is not possible.

4.2.5 *Production Integrity.*

For any of the electronic device production stages, the *stage* or *status* attributes of an ICtoken can only be incremented [Algorithm 2]. This offers rollback protection, ensuring that ICs in the production phase do not re-enter the supply chain at earlier stages. Moreover, ICtracker allows adding device identifiers to ICtokens only at the relevant stages of production. This ensures a one-way progression of an IC through the production

Version	Function	Owner	Stage/ Status	PID	EDID	Description
GDSII → IC						
1	`enollIC()`	Owner1	1/ 1	Null	Null	Owner1 enrolls the IC after fabrication.
2	`transferIC()`	Owner2	1/ 1	Null	Null	Owner1 transfers the IC to Owner2
3	`updateStage()`	Owner2	2/ 0	Null	Null	Owner2 takes IC to PCB assembly
IC → PCB						
4	`updateStage()`	Owner2	2/ 1	Null	Null	PCB assembly finished
5	`updatePIDorEDID()`	Owner2	2/ 1	"......"	Null	Owner2 updates PID for all ICs in PCB
6	`transferIC()`	Owner3	2/ 1	"......"	Null	Owner2 sells the PCB to Owner3 (distributor)
7	`transferIC()`	Owner4	2/ 1	"......"	Null	Owner3 sells the PCB to Owner4
8	`updateStage()`	Owner4	3/ 0	"......"	Null	Owner4 takes PCB to System Integration
PCB → Device						
9	`updateStage()`	Owner4	3/ 1	"......"	Null	System Integration finished
10	`updatePIDorEDID()`	Owner4	3/ 1	"......"	"......"	Owner4 updates EDID for all ICs in SoC
11	`transferIC()`	Owner5	3/ 1	"......"	"......"	Owner4 sells the SoC to Owner5 (retailer)
12	`transferIC()`	Owner6	3/ 1	"......"	"......"	Owner5 sells the SoC to Owner6 (end user)
13	`updateStage()`	Owner6	4/ 0	"......"	"......"	Owner6 updates the stage during device setup
14	`updateStage()`	Owner6	4/ 1	"......"	"......"	Owner6 updates the stage before recycling
15	`transferIC()`	Owner7	4/ 1	"......"	"......"	Owner6 sells the device to Owner7 (recycler)
End of Life						

Table 2: Life cycle of an ICtoken from design to recycling

process. In table 2 we analyse our complete protocol, by following the ICtoken of an IC as it progresses from fabrication to end of life.

4.3 Security

We now evaluate our proposed framework on the various threats in an untrusted supply chain.

4.3.1 Overbuilding and Cloning. Overbuilding or Cloning of chips is a threat made possible by malicious actors with access to design information from the IP owner. Such chips are not accounted for by the IP owner and will not be enrolled under the IP owner's account and ergo cannot be sold under the original brand. The support for active hardware metering within the framework ensures that such chips are not functional even if enrolled under a different brand.

4.3.2 Remarking and illegitimate-recycling. Often bootleggers try to tamper with the markings on the package of an IC to sell it under the guise of a higher-grade product. Our framework stores all information present on the package of the IC and can immediately catch such discrepancies and invalidate a transaction. A similar situation is presented when bootleggers try to sell a recycled product under the guise of being brand new, which is easily caught with the production information.

4.3.3 Defective, Out-of-spec, Forged certification and Tampering. The framework allows for owners to report malfunctioning ICs and ensures that such devices do not circulate in the supply chain.

4.4 Comparative Analysis

In table 3 we compare the ICtoken framework with prior works leveraging blockchain technology to monitor the IC supply chain. [18], [6] and [10] do not track the production stage or status of an IC and can not provide product-level provenance. [10], [37] and [35] propose consortium managed frameworks that are not transparent. They only permit entities within the consortium to utilize the system. Except for [10] all of the other works store details of genuine ICs in the plain and do not account for data privacy. [18], [37] and [35] store CRPs for authenticating an IC leading to huge transaction sizes. Unlike all previous works, ICtoken can securely store active metering information for an IC in the form of an in-borne *public key* which can then be used to validate its authenticity.

Framework	OT	PT	Tr	DP	AM	TS
[18]	✓	✗	✓	✗	✗	at least 2KB
[6]	✓	✗	✓	✗	✗	-
[10]	✓	✗	✗	✓	✗	-
[37]	✓	✓	✗	✗	✗	at least 2KB
[35]	✓	✓	✗	✗	✗	at least 2KB
ICtoken	✓	✓	✓	✓	✓	**0.7KB**

Table 3: Comparison with prior works. [OT: Ownership Tracking; PT: Production Tracking; Tr: Transparency; DP: Data Privacy; AM: Active Metering; TS:Transaction Size]

5 CONCLUSION

In this paper, we address the supply chain threats to the IP owner that arise due to the globalization of the electronics supply chain. The main problem with prior work in this area is that they are not designed with the end-product in mind and do not serve much purpose to the end-user. The proposed framework mitigates their shortcomings to offer enhanced IP integrity and stronger assurance of a genuine product to the end-users. The main contribution of this work is the idea of ICtoken as a building block NFT unit for a product-level NFT. The transparent design with certified metadata of all components used in the product fortifies credibility across the supply chain and authenticity of end-product.

REFERENCES

[1] Anonymous. 2022. ICtoken Implementations. (2022). This citation has been redacted to protect the anonymity of the source.

[2] Quinn Barry. 2021. Inside Radical Semiconductor: The Stanford Startup Disrupting the Chip Industry. Identity Review. (2021). https://identityreview.com/inside-radical-semiconductor-the-stanford-startup-disrupting-the-chip-industry/

[3] Peter Beerel, Marios Georgiou, Ben Hamlin, Alex J. Malozemoff, and Pierluigi Nuzzo. 2022. Towards a Formal Treatment of Logic Locking. Cryptology ePrint Archive, Report 2022/503. https://ia.cr/2022/503.

[4] Juan Benet. 2014. IPFS - Content Addressed, Versioned, P2P File System. IPFS Whitepaper. (2014). https://ipfs.io/ipfs/QmR7GSQM93Cx5eAg6a6yRzNde1FQv7uL6X1o4k7zrJa3LX/ipfs.draft3.pdf

[5] Vitalik Buterin. 2014. Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform. Ethereum Whitepaper. (2014). https://ethereum.org/en/whitepaper/

[6] Chandan Kumar Chaudhary, Urbi Chatterjee, and Debdeep Mukhopadhayay. 2021. Auto-PUFChain: An Automated Interaction Tool for PUFs and Blockchain in Electronic Supply Chain. In *2021 Asian Hardware Oriented Security and Trust Symposium (AsianHOST)*. 1–4. https://doi.org/10.1109/AsianHOST53231.2021.9699720

[7] Animesh Chhotaray and Thomas Shrimpton. 2021. Hardening Circuit-Design IP Against Reverse-Engineering Attacks. Cryptology ePrint Archive, Report 2021/456. https://ia.cr/2021/456.

[8] Gustavo K. Contreras, Md. Tauhidur Rahman, and Mohammad Tehranipoor. 2013. Secure Split-Test for preventing IC piracy by untrusted foundry and assembly. In *2013 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFTS)*. 196–203. https://doi.org/10.1109/DFT.2013.6653606

[9] National Counterintelligence and Security Center. 2022. Supply Chain Risks to Semiconductors. (2022). https://www.dni.gov/files/NCSC/documents/supplychain/semiconductor-supply-chain-2022-39E2C6B0-.pdf

[10] Pinchen Cui, Julie Dixon, Ujjwal Guin, and Daniel Dimase. 2019. A Blockchain-Based Framework for Supply Chain Provenance. *IEEE Access* 7 (2019), 157113–157125. https://doi.org/10.1109/ACCESS.2019.2949951

[11] William Entriken, Dieter Shirley, Jacob Evans, and Nastassia Sachs. 2018. EIP-721: Non-Fungible Token Standard. Ethereum Improvement Proposals. (2018). https://eips.ethereum.org/EIPS/eip-721

[12] Ethereum. 2018. ERC-721 NON-FUNGIBLE TOKEN STANDARD. (2018). https://ethereum.org/en/developers/docs/standards/tokens/erc-721/

[13] Ethereum. 2022. Solidity Documentation Release 0.8.15. (2022). https://buildmedia.readthedocs.org/media/pdf/solidity/develop/solidity.pdf

[14] Fatemeh Ganji, Shahin Tajik, Jean-Pierre Seifert, and Domenic Forte. 2019. Blockchain-enabled Cryptographically-secure Hardware Obfuscation. Cryptology ePrint Archive, Report 2019/928. https://ia.cr/2019/928.

[15] Ujjwal Guin, Daniel DiMase, and Mohammad Tehranipoor. 2014. Counterfeit integrated circuits: Detection, avoidance, and the challenges ahead. *Journal of Electronic Testing* 30, 1 (2014), 9–23. https://doi.org/10.1007/s10836-013-5430-8

[16] Charles Herder, Meng-Day Yu, Farinaz Koushanfar, and Srinivas Devadas. 2014. Physical Unclonable Functions and Applications: A Tutorial. *Proc. IEEE* 102, 8 (2014), 1126–1141. https://doi.org/10.1109/JPROC.2014.2320516

[17] INTRINSIC ID. 2017. SRAM PUF : The Secure Silicon Fingerprint. https://www.intrinsic-id.com/resources/white-papers/white-paper-sram-puf-secure-silicon-fingerprint/.

[18] Md Nazmul Islam and Sandip Kundu. 2019. Enabling IC Traceability via Blockchain Pegged to Embedded PUF. *ACM Trans. Des. Autom. Electron. Syst.* 24, 3, Article 36 (apr 2019), 23 pages. https://doi.org/10.1145/3315669

[19] Hadi Mardani Kamali, Kimia Zamiri Azar, Farimah Farahmandi, and Mark Tehranipoor. 2022. Advances in Logic Locking: Past, Present, and Prospects. Cryptology ePrint Archive, Report 2022/260. https://ia.cr/2022/260.

[20] Farinaz Koushanfar. 2010. Hardware Metering: A Survey. (10 2010). https://doi.org/10.1007/978-1-4419-8080-9_5

[21] Serge Leef. 2018. Supply Chain Hardware Integrity for Electronics Defense. *NIST - CSRC* (2018). https://csrc.nist.gov/CSRC/media/Projects/cyber-supply-chain-risk-management/documents/SSCA/Winter_2018/TuePM2.1-SHIELD.pdf

[22] Monax. 2019. (2019). https://monax.io/help/advanced/nft-integration/

[23] Satoshi Nakamoto. 2008. Bitcoin: A Peer-to-Peer Electronic Cash System. Bitcoin Whitepaper. (2008). https://bitcoin.org/en/bitcoin-paper

[24] Mahipal Nehra. 2021. Blockchain in Supply Chain: A Transparent Prospect for Products. ReadWrite. (2021). https://readwrite.com/blockchain-in-supply-chain-a-transparent-prospect-for-products/

[25] Emily Newton. 2022. Is the chip shortage leading to more counterfeit components? Embedded.com. (2022). https://www.embedded.com/is-the-chip-shortage-leading-to-more-counterfeit-components/

[26] U.S. Department of Commerce: Bureau of Industry and Security Office of Technology Evaluation. 2010. Defense Industrial Base Assessment: Counterfeit Electronics. (2010). https://agmaglobal.org/uploads/BIS%20Survey%20(January%202010%20final).pdf

[27] United States Government Accountability Office. 2016. Counterfeit Parts: DOD Needs to Improve Reporting and Oversight to Reduce Supply Chain Risk. *Report to Congressional Committees* (2016). https://www.gao.gov/assets/gao-16-236.pdf

[28] Jeyavijayan J V Rajendran. 2017. An overview of hardware intellectual property protection. (2017), 1–4. https://doi.org/10.1109/ISCAS.2017.8050883

[29] Masoud Rostami, Farinaz Koushanfar, and Ramesh Karri. 2014. A Primer on Hardware Security: Models, Methods, and Metrics. *Proc. IEEE* 102, 8 (2014), 1283–1295. https://doi.org/10.1109/JPROC.2014.2335155

[30] Jarrod A. Roy, Farinaz Koushanfar, and Igor L. Markov. 2008. EPIC: Ending Piracy of Integrated Circuits. In *2008 Design, Automation and Test in Europe*. 1069–1074. https://doi.org/10.1109/DATE.2008.4484823

[31] skuchainweb. 2016. Skuchain: Here's how blockchain will save global trade a trillion dollars. (2016). https://www.skuchain.com/skuchain-heres-how-blockchain-will-save-global-trade-a-trillion-dollars/

[32] Precision Software. 2019. Could Blockchain Revolutionize Parcel Shipping? *FedEx* (2019). https://www.fedex.com/content/dam/fedex/us-united-states/Compatible-Solutions/images/2019/Q2/Could_Blockchain_Revolutionize_Parcel_Shipping_V2_50457811.pdf

[33] Archana Sristy. 2021. National Counterintelligence and Security Center. *Walmart* (2021). https://one.walmart.com/content/globaltechindia/en_in/Tech-insights/blog/Blockchain-in-the-food-supply-chain.html

[34] OpenSea the largest NFT marketplace. 2017. (2017). https://opensea.io/

[35] Jason Vosatka, Andrew Stern, M.M. Hossain, Fahim Rahman, Jeffery Allen, Monica Allen, Farimah Farahmandi, and Mark Tehranipoor. 2020. Tracking Cloned Electronic Components using a Consortium-based Blockchain Infrastructure. In *2020 IEEE Physical Assurance and Inspection of Electronics (PAINE)*. 1–6. https://doi.org/10.1109/PAINE49178.2020.9337735

[36] Qin Wang, Rujia Li, Qi Wang, and Shiping Chen. 2021. Non-Fungible Token (NFT): Overview, Evaluation, Opportunities and Challenges. (05 2021).

[37] Xiaolin Xu, Fahim Rahman, Bicky Shakya, Apostol Vassilev, Domenic Forte, and Mark Tehranipoor. 2019. Electronics Supply Chain Integrity Enabled by Blockchain. *ACM Trans. Des. Autom. Electron. Syst.* 24, 3 (2019), 25 pages. https://doi.org/10.1145/3315571

[38] Muhammad Yasin and Ozgur Sinanoglu. 2017. Evolution of Logic Locking. (10 2017). https://doi.org/10.1109/VLSI-SoC.2017.8203496

[39] Ernst & Young. 2022. EY OpsChain Contract Manager. ey.com. (2022). https://www.ey.com/en_gl/blockchain-platforms/contract-manager