EPIC: Ending Piracy of Integrated Circuits

Firas Darwish

- **LSI Logic** quit semiconductor manufacturing in 2005
- **Texas Instruments** chose not to develop sub-45nm fabrication in-house
- **Qualcomm** became the first fabless semiconductor company to rank among top 10 Integrated Circuits (IC) producers worldwide (2007)
- **AMD** has been outsourcing some of its production to foundries throughout the world.

The trend is clear!

"However, with the growth of manufacturing potential in Asia, piracy has become rampant, thanks to loose IP protection policies and weak enforcement"

:(

As we move away from in-house fabrication, there are critical privacy and security concerns that arise.

The paper proposes a novel technique to **end piracy of integrated circuits** (EPIC).

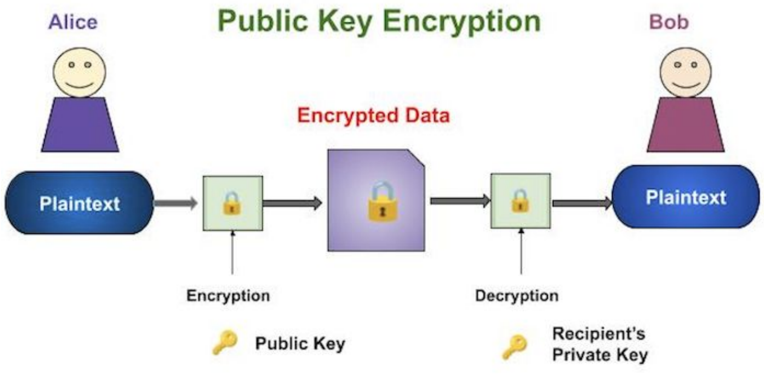
(non-technical big idea)

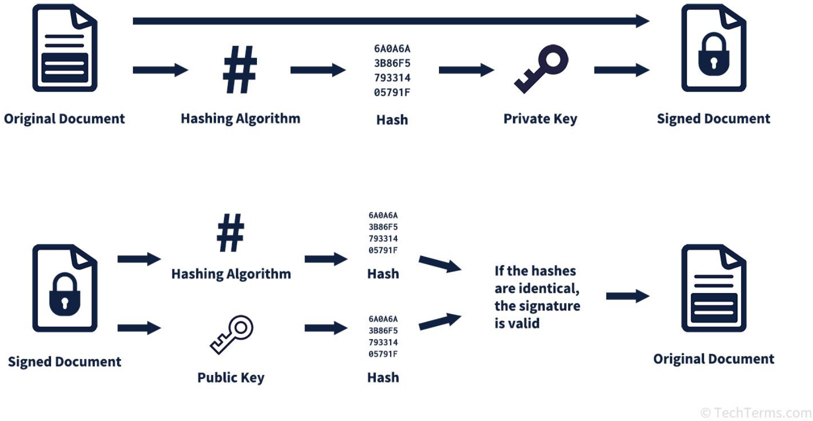
1. Each chip creates a unique ID using standard methods before it's tested.
2. The manufacturer sends this ID to the IP-holder.
3. The IP-holder provides a special activation code that works only for that specific chip

⇒ The IP-holder controls exactly how many chips are made and prevents others from making functional copies.

:)

(review on public-key cryptography)





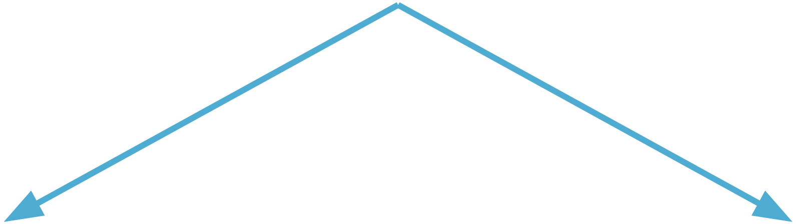
Public-key Encryption: *Guarantees that only recipient will be able to decrypt and read transmitted message.*

Digital Signature: *Verifies the authenticity of a digital message, but by itself does not guarantee that only the recipient will be able to decrypt and read transmitted message.*

Encryption and decryption rely on hard-to-reverse (one-way) mathematical functions, such as high-precision integer multiplication and modular exponentiation.

RSA-style crypto-systems are among the most studied in the literature, but remain resilient against a variety of attacks 30 years after their inception.

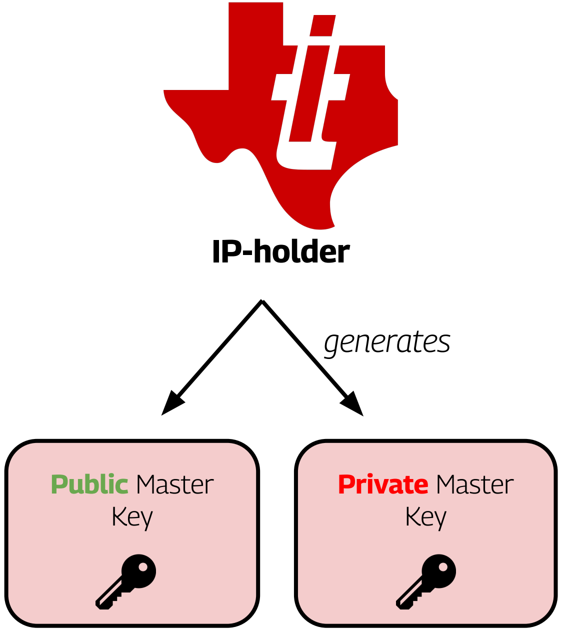
The proposed technique consists of modifications to existing IC design flows (great!)



(1) Embed keys
into the circuit

(2) New protocol
for chip activation

IP-holder



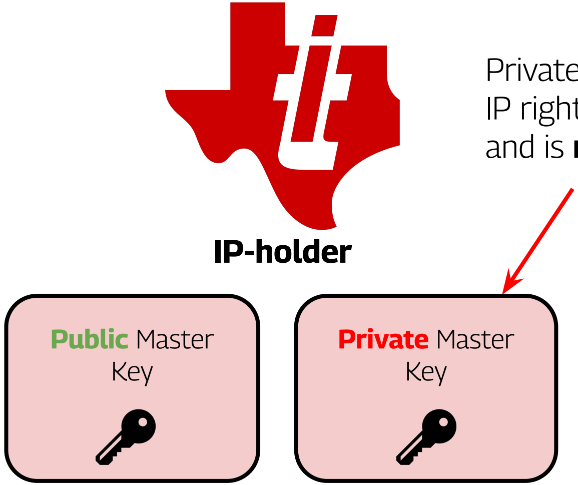
IP-holder

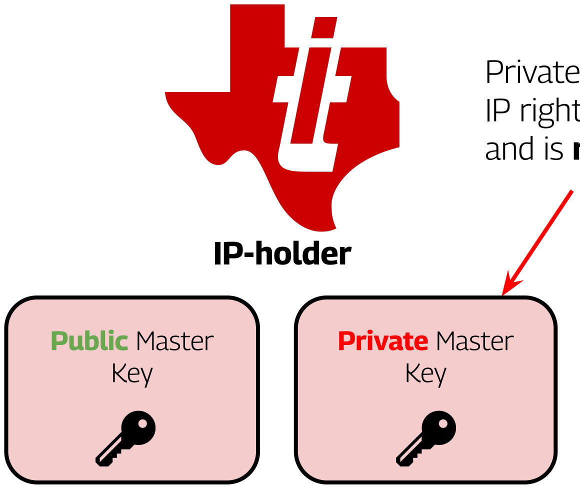
generates

Public Master Key

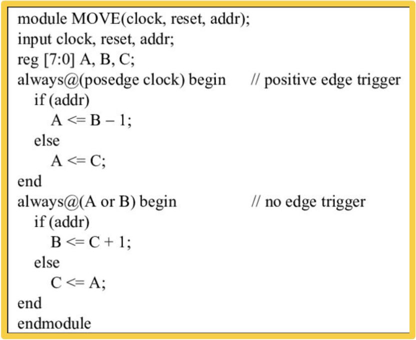
Private Master Key

Private Master Key embodies
IP rights for a given design
and is **never transmitted**.





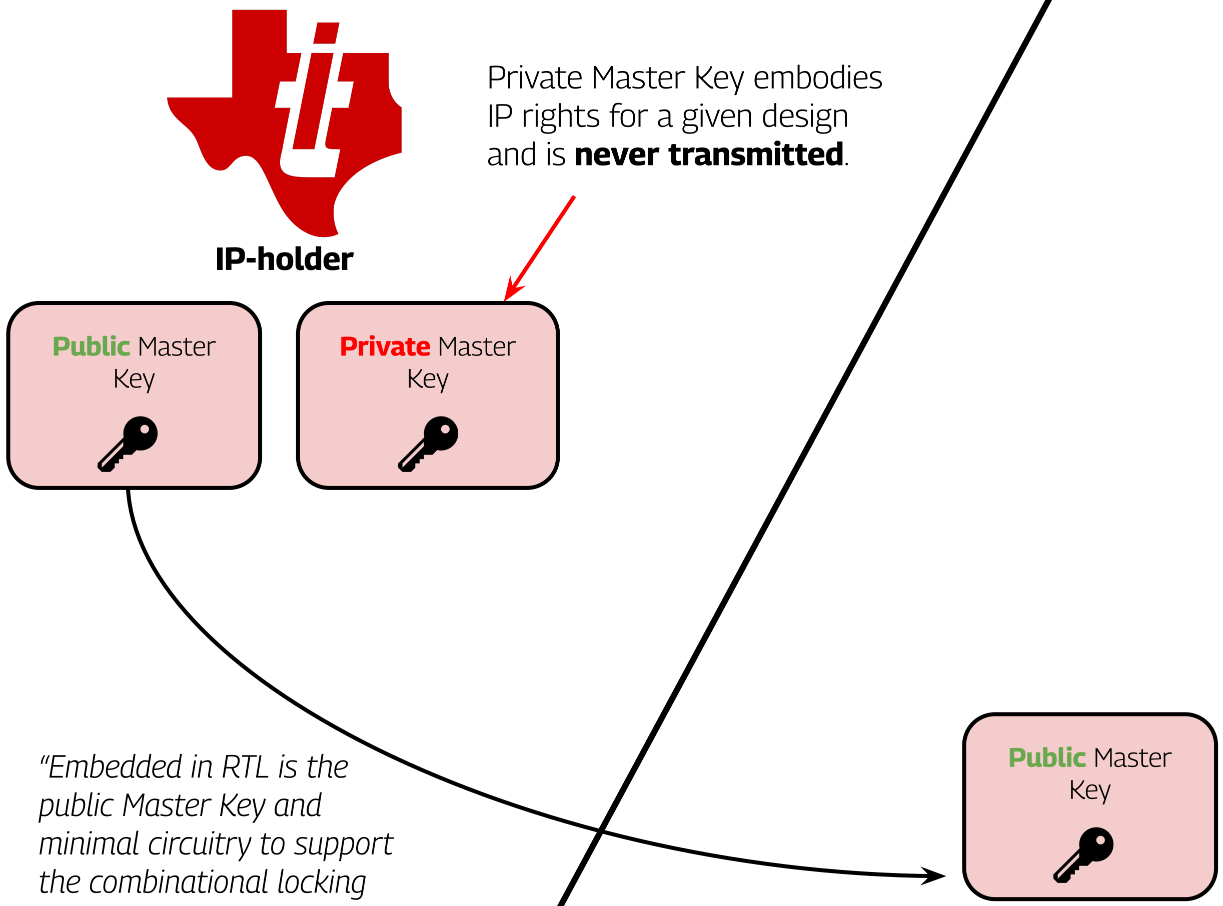
Private Master Key embodies IP rights for a given design and is **never transmitted**.



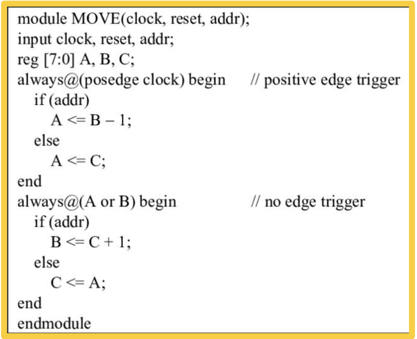
```
module MOVE(clock, reset, addr);
input clock, reset, addr;
reg [7:0] A, B, C;
always@(posedge clock) begin      // positive edge trigger
   if (addr)
      A <= B – 1;
   else
      A <= C;
end
always@(A or B) begin             // no edge trigger
   if (addr)
      B <= C + 1;
   else
      C <= A;
end
endmodule
```

RTL

Private Master Key embodies IP rights for a given design and is **never transmitted**.

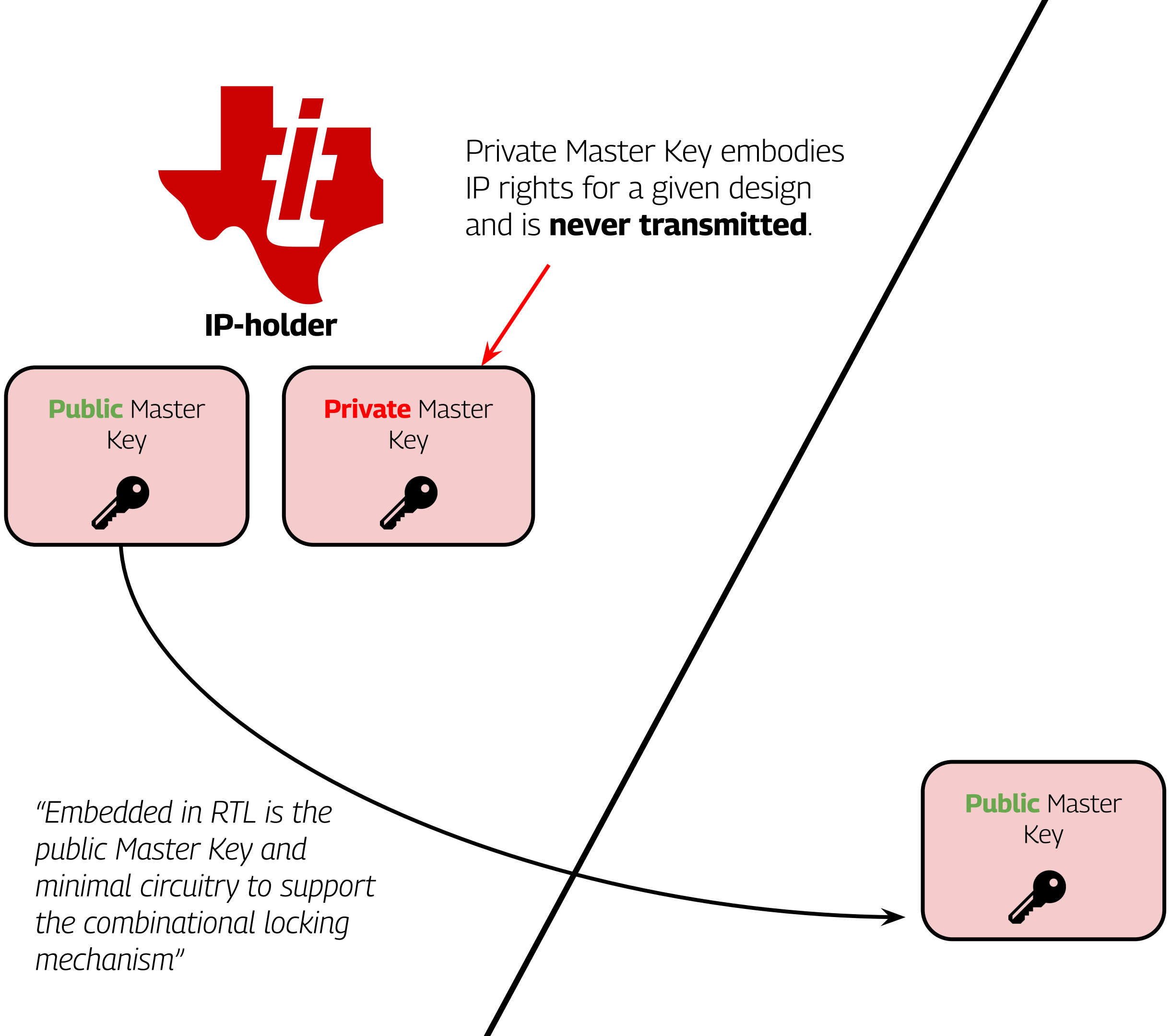


"Embedded in RTL is the public Master Key and minimal circuitry to support the combinational locking mechanism"



```
module MOVE(clock, reset, addr);
input clock, reset, addr;
reg [7:0] A, B, C;
always@(posedge clock) begin      // positive edge trigger
   if (addr)
      A <= B – 1;
   else
      A <= C;
end
always@(A or B) begin              // no edge trigger
   if (addr)
      B <= C + 1;
   else
      C <= A;
end
endmodule
```

RTL

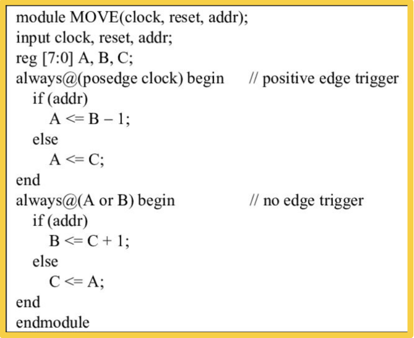


Private Master Key embodies IP rights for a given design and is **never transmitted**.

"RTL descriptions are enriched with support for on-chip TRNG and public-key cryptography"

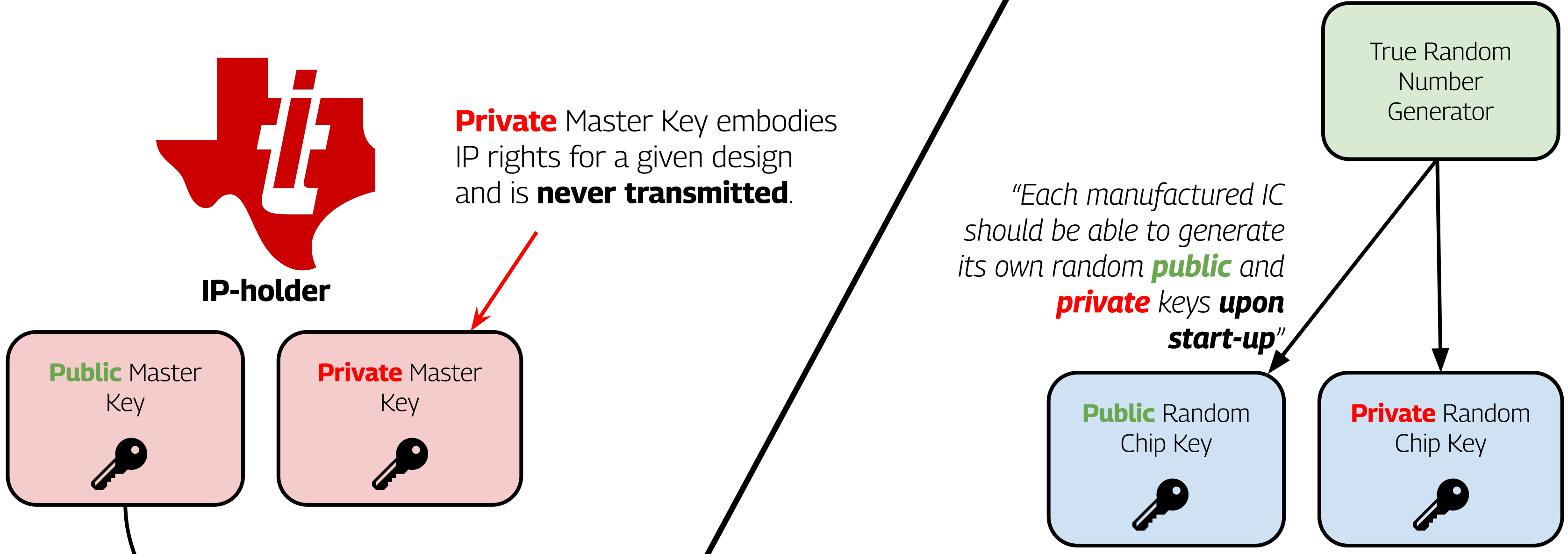
True Random Number Generator

"Embedded in RTL is the public Master Key and minimal circuitry to support the combinational locking mechanism"



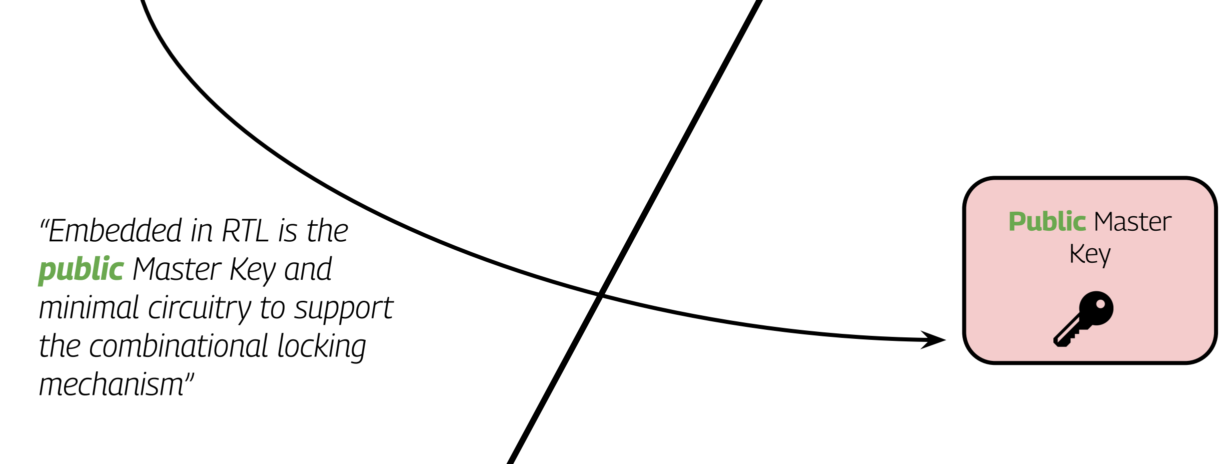
```
module MOVE(clock, reset, addr);
input clock, reset, addr;
reg [7:0] A, B, C;
always@(posedge clock) begin      // positive edge trigger
   if (addr)
      A <= B – 1;
   else
      A <= C;
end
always@(A or B) begin              // no edge trigger
   if (addr)
      B <= C + 1;
   else
      C <= A;
end
endmodule
```

RTL

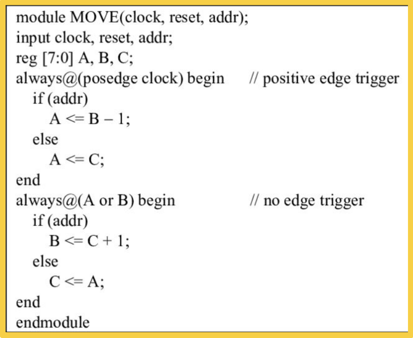


Private Master Key embodies IP rights for a given design and is **never transmitted**.

"Each manufactured IC should be able to generate its own random ***public*** *and* ***private*** *keys* ***upon start-up****"*



"Embedded in RTL is the ***public*** *Master Key and minimal circuitry to support the combinational locking mechanism"*

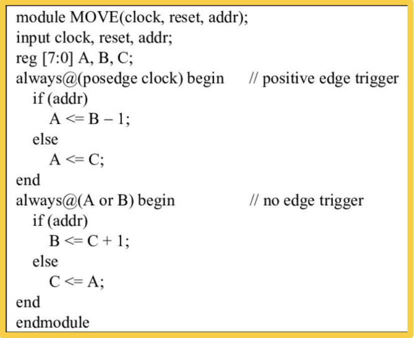


```
module MOVE(clock, reset, addr);
input clock, reset, addr;
reg [7:0] A, B, C;
always@(posedge clock) begin      // positive edge trigger
   if (addr)
      A <= B – 1;
   else
      A <= C;
end
always@(A or B) begin              // no edge trigger
   if (addr)
      B <= C + 1;
   else
      C <= A;
end
endmodule
```

RTL

True Random Number Generator

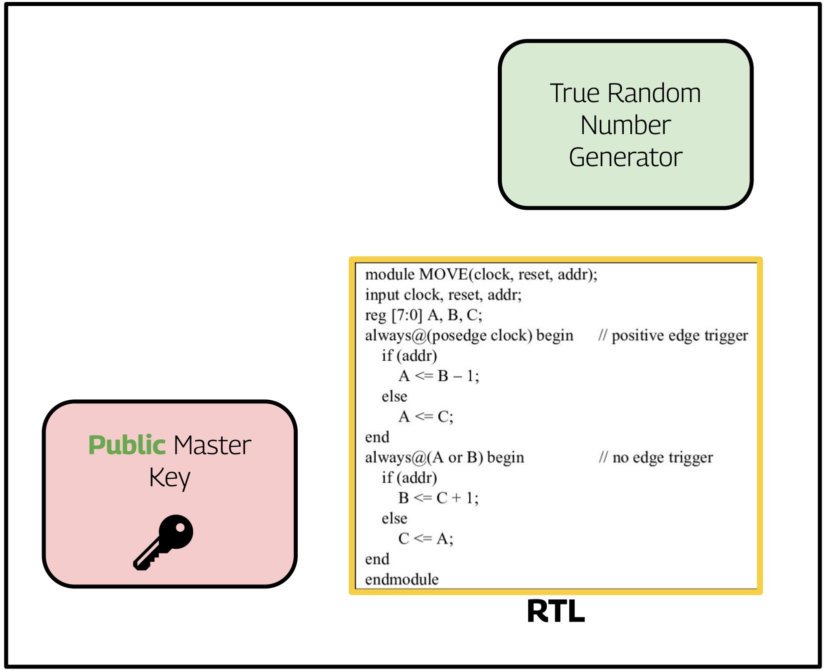
Public Master Key



```
module MOVE(clock, reset, addr);
input clock, reset, addr;
reg [7:0] A, B, C;
always@(posedge clock) begin      // positive edge trigger
   if (addr)
      A <= B – 1;
   else
      A <= C;
end
always@(A or B) begin             // no edge trigger
   if (addr)
      B <= C + 1;
   else
      C <= A;
end
endmodule
```

RTL

"A gate-level netlist is produced from the enriched RTL using traditional logic synthesis and technology mapping, followed by circuit placement"



True Random Number Generator

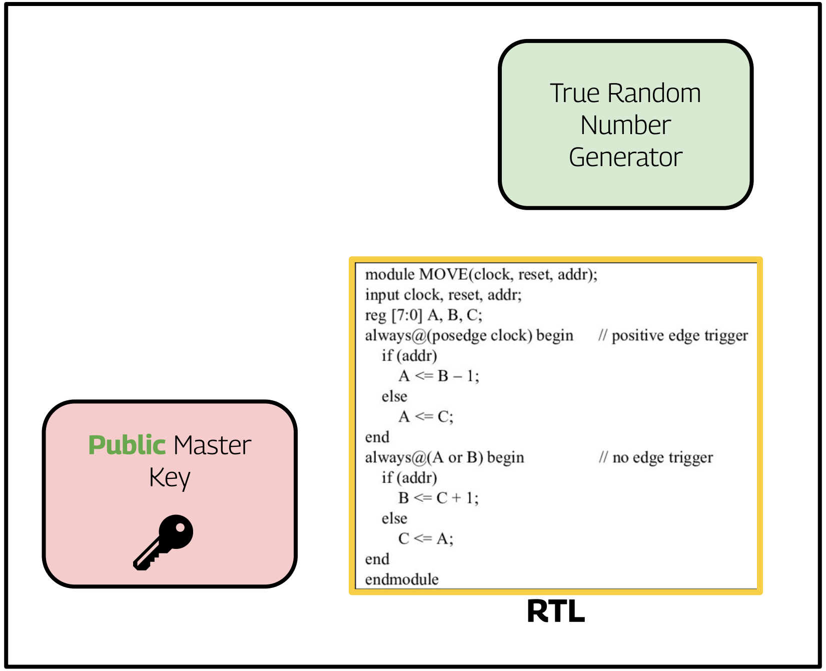
Public Master Key

```
module MOVE(clock, reset, addr);
input clock, reset, addr;
reg [7:0] A, B, C;
always@(posedge clock) begin      // positive edge trigger
   if (addr)
      A <= B – 1;
   else
      A <= C;
end
always@(A or B) begin             // no edge trigger
   if (addr)
      B <= C + 1;
   else
      C <= A;
end
endmodule
```

RTL

*“Now critical paths in the circuit are known, and one may connect the **anti-piracy logic** without disturbing them”*

“A gate-level netlist is produced from the enriched RTL using traditional logic synthesis and technology mapping, followed by circuit placement”

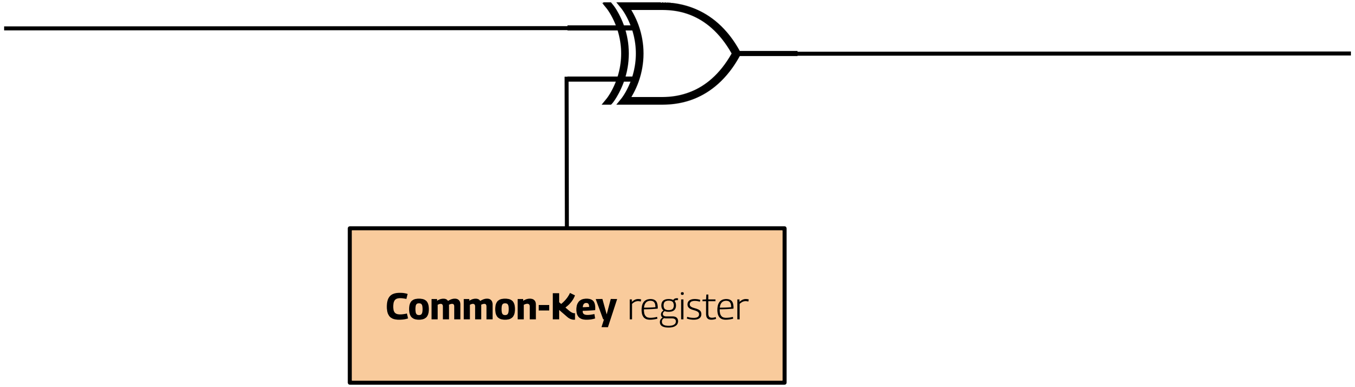


```
module MOVE(clock, reset, addr);
input clock, reset, addr;
reg [7:0] A, B, C;
always@(posedge clock) begin      // positive edge trigger
   if (addr)
      A <= B – 1;
   else
      A <= C;
end
always@(A or B) begin             // no edge trigger
   if (addr)
      B <= C + 1;
   else
      C <= A;
end
endmodule
```

(***anti-piracy logic***)

(**anti-piracy logic**)

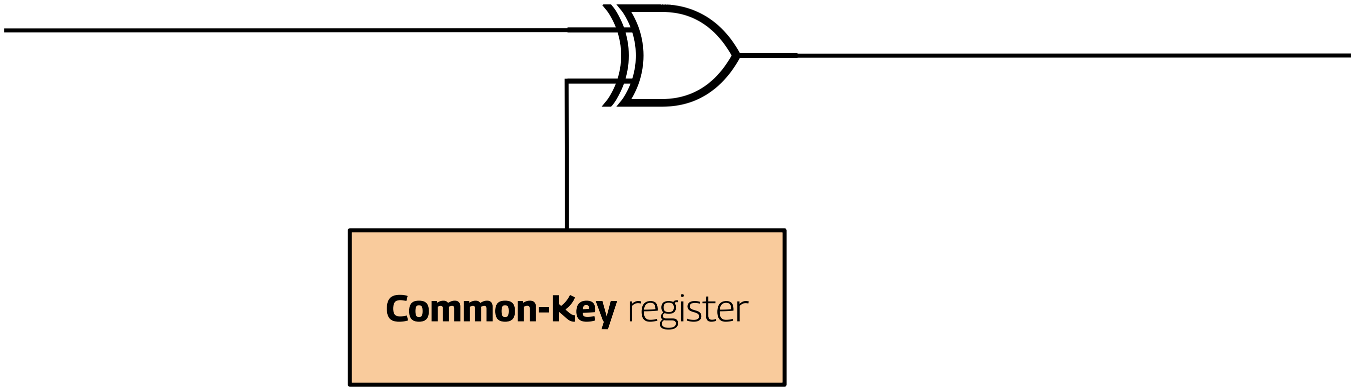
"Combinational locking is performed in most important modules of the IC by adding XOR gates on selected non-critical [have slack in their timing budget] wires"



Common-Key register

(**anti-piracy logic**)

"Combinational locking is performed in most important modules of the IC by adding XOR gates on selected non-critical [have slack in their timing budget] wires"



Common-Key register

*"When the correct **Common Key** (CK) appears, the circuit is equivalent to the original. Otherwise, the circuit's behavior is altered, as if stray inverters were placed on selected wires.*

(*combinational logic locking mathematics*)

To protect a combinational circuit C(***x***) with a *k*-bit key, we develop a simple procedure that uses *k* new gates:

- First, *k* wires {*w_i*} are selected and matched with the bits {*y_i*} of the key.
- For each selected wire *w_i*, its driver is disconnected from the sinks and either an XOR gate *w′_i* = *w_i* ⊕ *y_i* or XNOR gate *w′_i* = ‘(*w_i* ⊕ *y_i*) is inserted, where *y_i* is the matched key bit and *w′_i* is a new wire that drives all sinks previously driven by *w_i*.
    - The choice between XOR and XNOR depends on whether y_i is 0 (XOR) or 1 (XNOR)

$$\exists! y \; \forall x \; C'(x, y) = C(x)$$

Solving such a boolean equation is harder than NP-complete, due to alternating quantifiers.

(***combinational logic locking mathematics***)

- A key should be long enough to withstand *brute-force attacks*, which are defined as algorithms searching for a key that evaluate combinations and spend **Ω**(1) time per combination.

- Most incorrect key combinations can be weeded out by scanning in-test patterns and comparing circuit's responses to expected values.
  - With a single scan-chain, this will take $\Omega(2^k)$ time for a *k*-bit key.

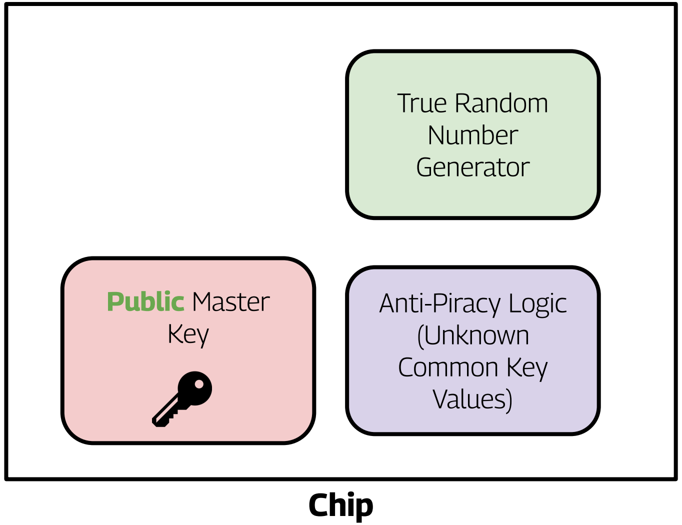
Definition 1 *Given a circuit C'(x,y) locked with key y, the effective length L(y) of the key is log_2 of the expected number of combinations checked by best brute-force attack.*

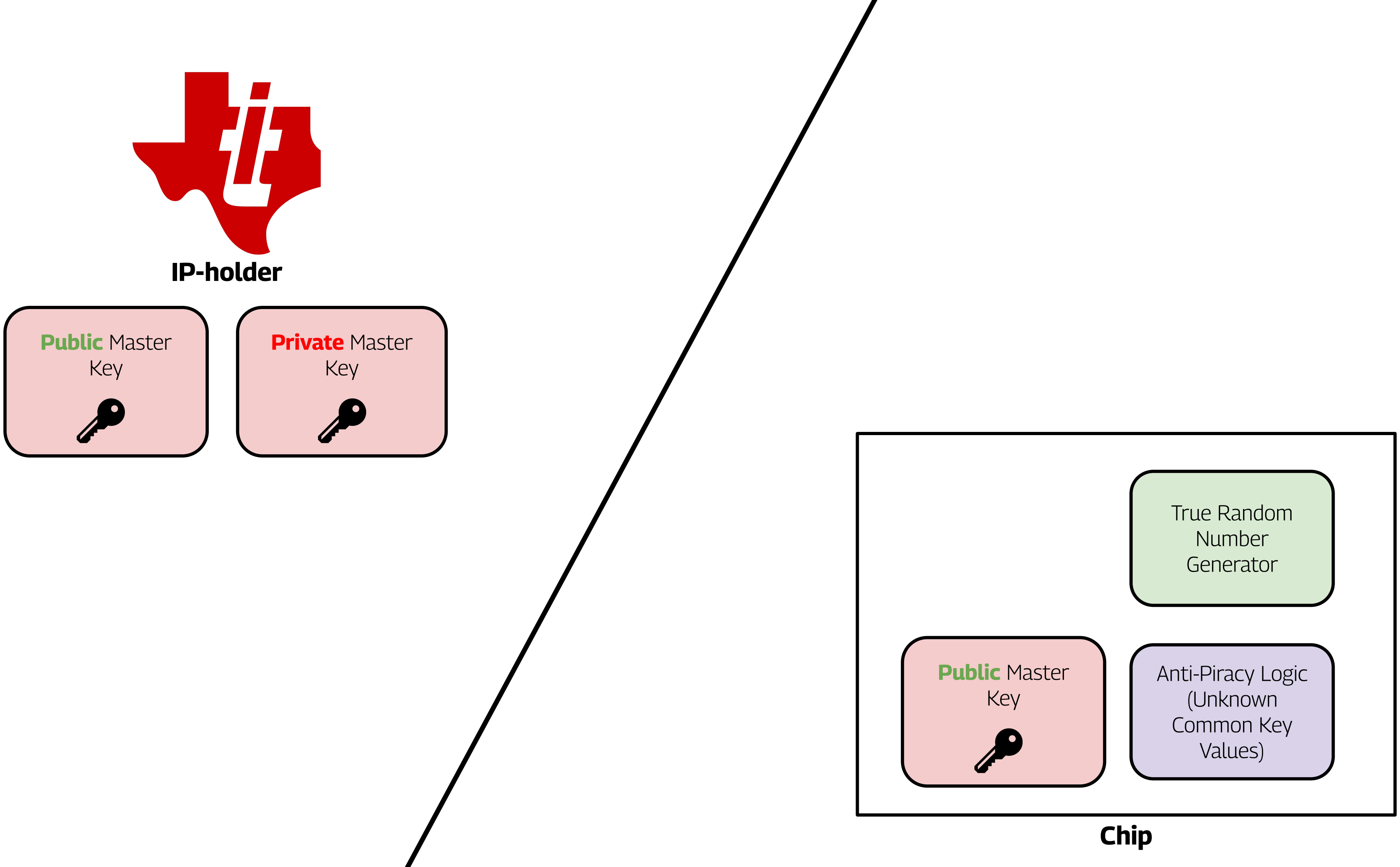
Theorem 1 *Consider a circuit C'(x,y) such that the key y loks n independently-testable circuit modules and, for j = 1...n, exactly k_j bits of the key are dedicated to module j, while G_j key combinations of 2^(k_j) unlock module j. Then*

$$\mathcal{L}(\vec{y}) \leq \log_2 \left(\sum_{j=1}^{n} \frac{2^{k_j}}{G_j} \right) - 1.$$

Paper recommends L(y) >= 64. Summation–not multiplication–as they are n **independently-testable** circuit modules.

Is that all?


True Random Number Generator
Public Master Key
Anti-Piracy Logic (Unknown Common Key Values)
Chip



IP-holder

Public Master Key

Private Master Key

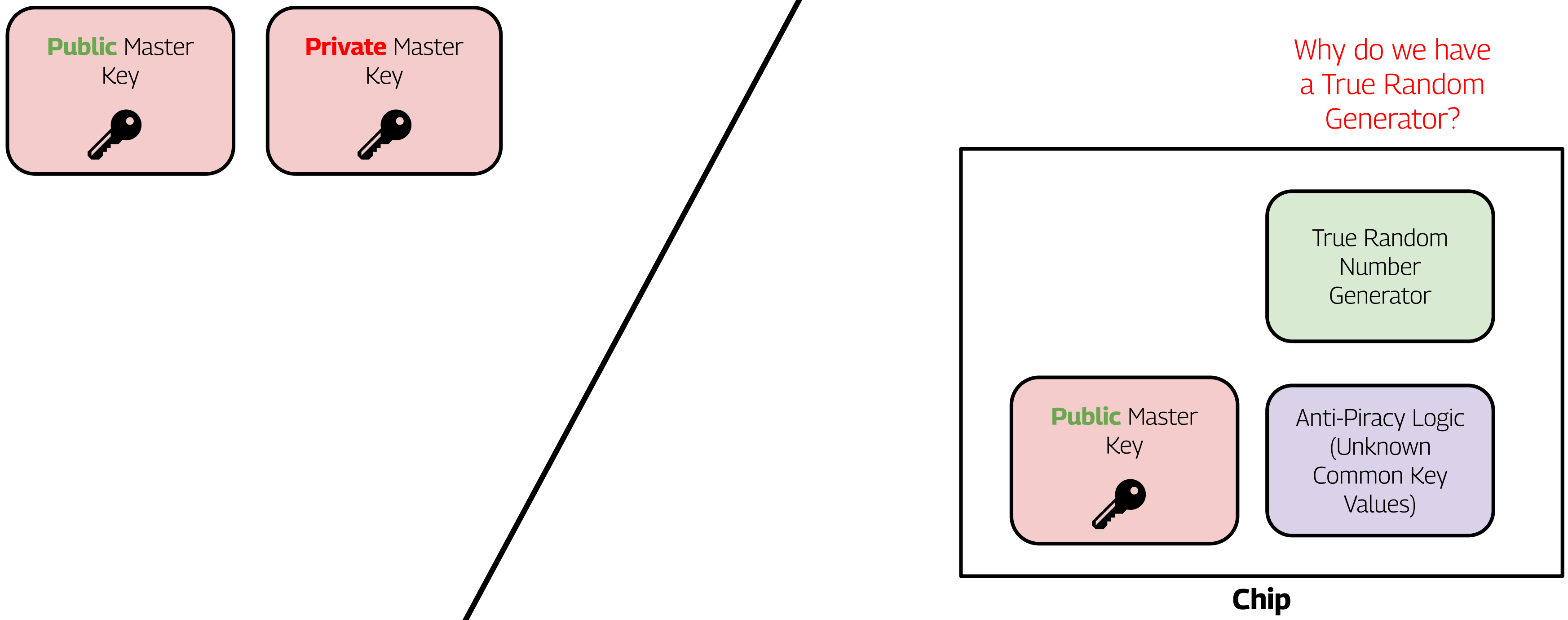
True Random Number Generator

Public Master Key

Anti-Piracy Logic (Unknown Common Key Values)

Chip

IP-holder



Public Master Key

Private Master Key

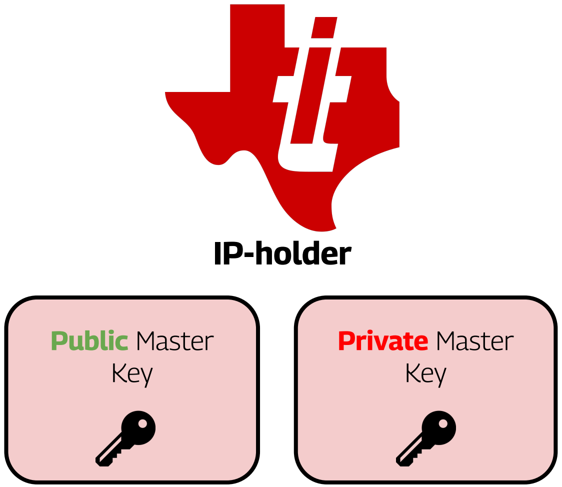
Why do we have a True Random Generator?

True Random Number Generator

Public Master Key

Anti-Piracy Logic (Unknown Common Key Values)

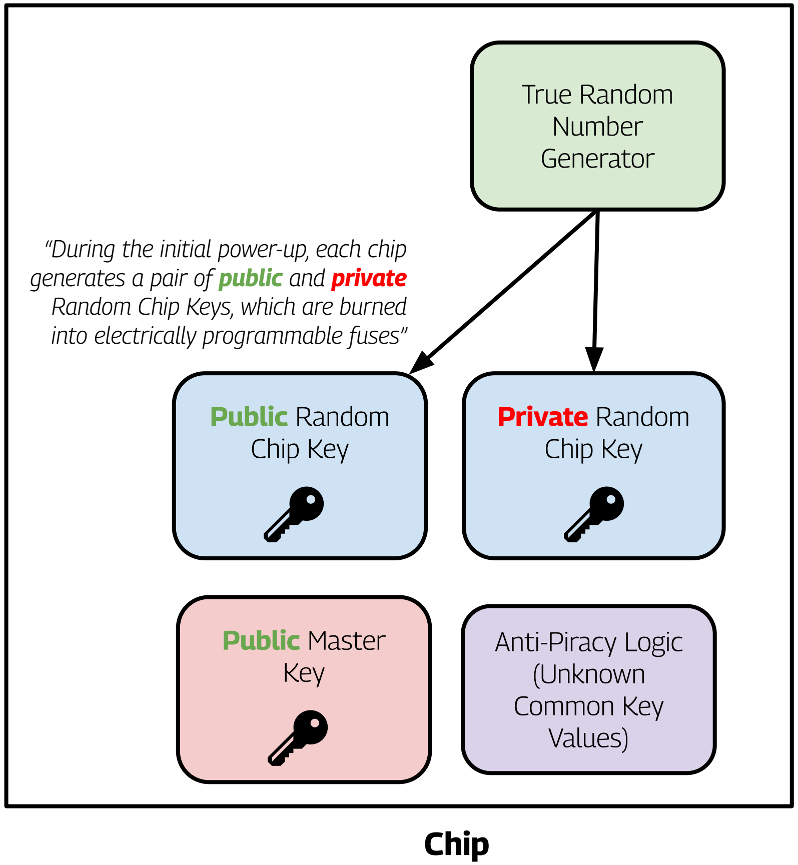
Chip



IP-holder

Public Master Key

Private Master Key



True Random Number Generator

*"During the initial power-up, each chip generates a pair of **public** and **private** Random Chip Keys, which are burned into electrically programmable fuses"*

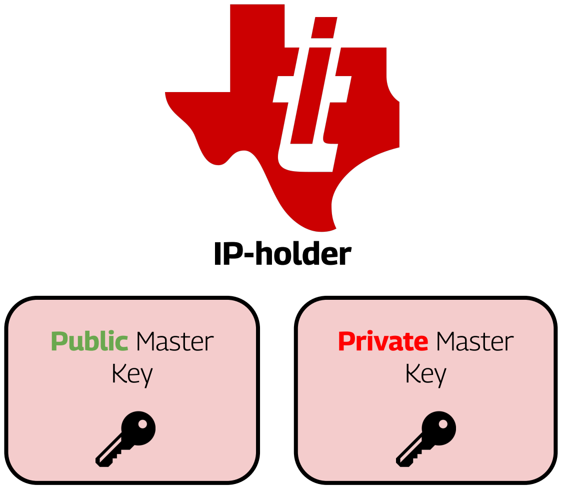
Public Random Chip Key

Private Random Chip Key

Public Master Key

Anti-Piracy Logic (Unknown Common Key Values)

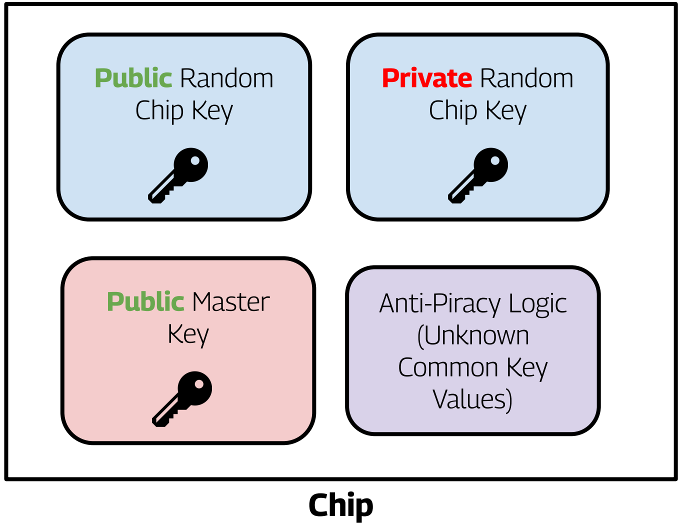
Chip

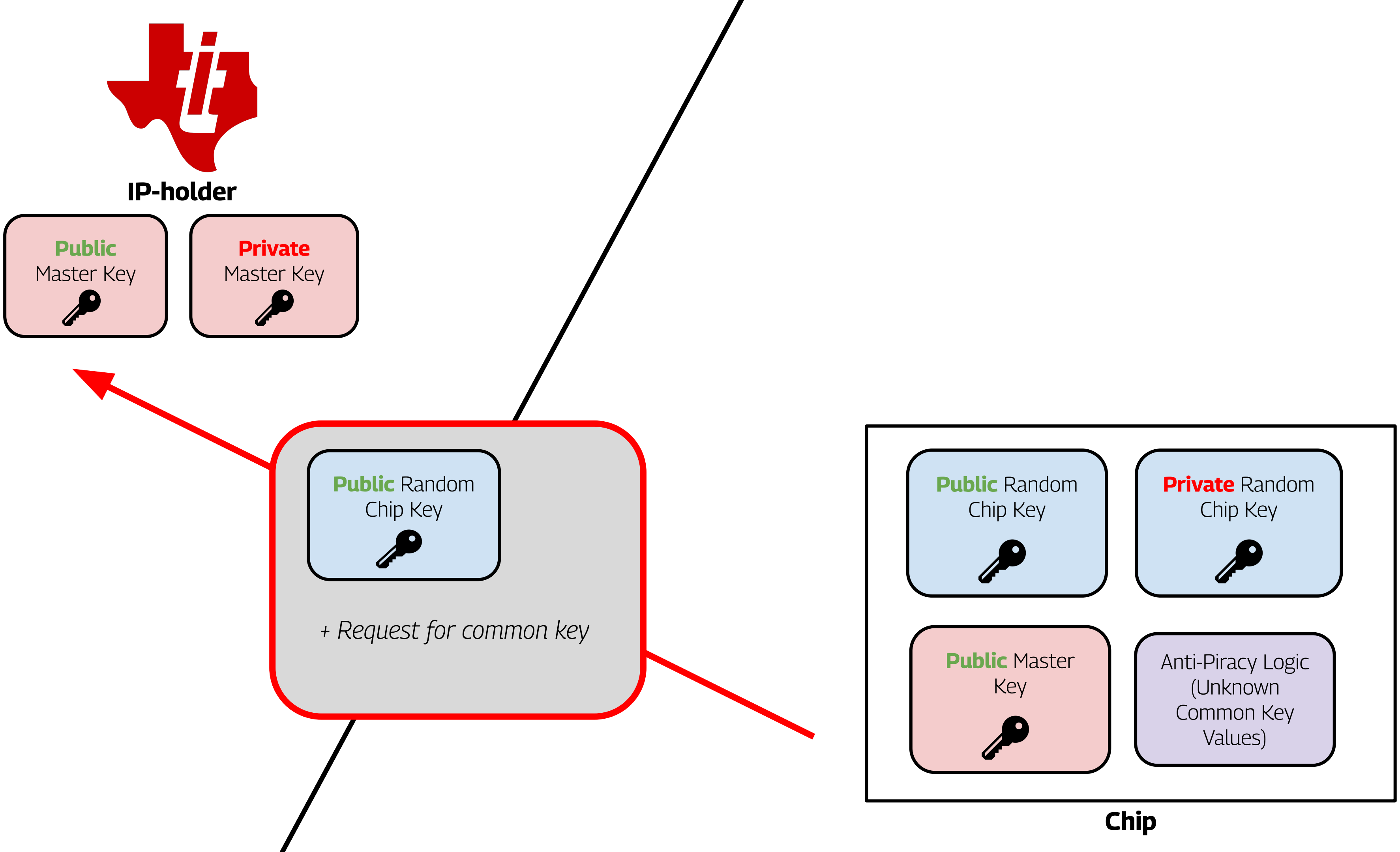


We need to figure out **what the common key** is to unlock the now-fabricated chip (only the IP-holder has common key).

And we have all these encryption keys from both sides of this **insecure communication channel**.

How do we bring this all together?





IP-holder

Public Master Key

Private Master Key

Public Random Chip Key

+ Request for common key

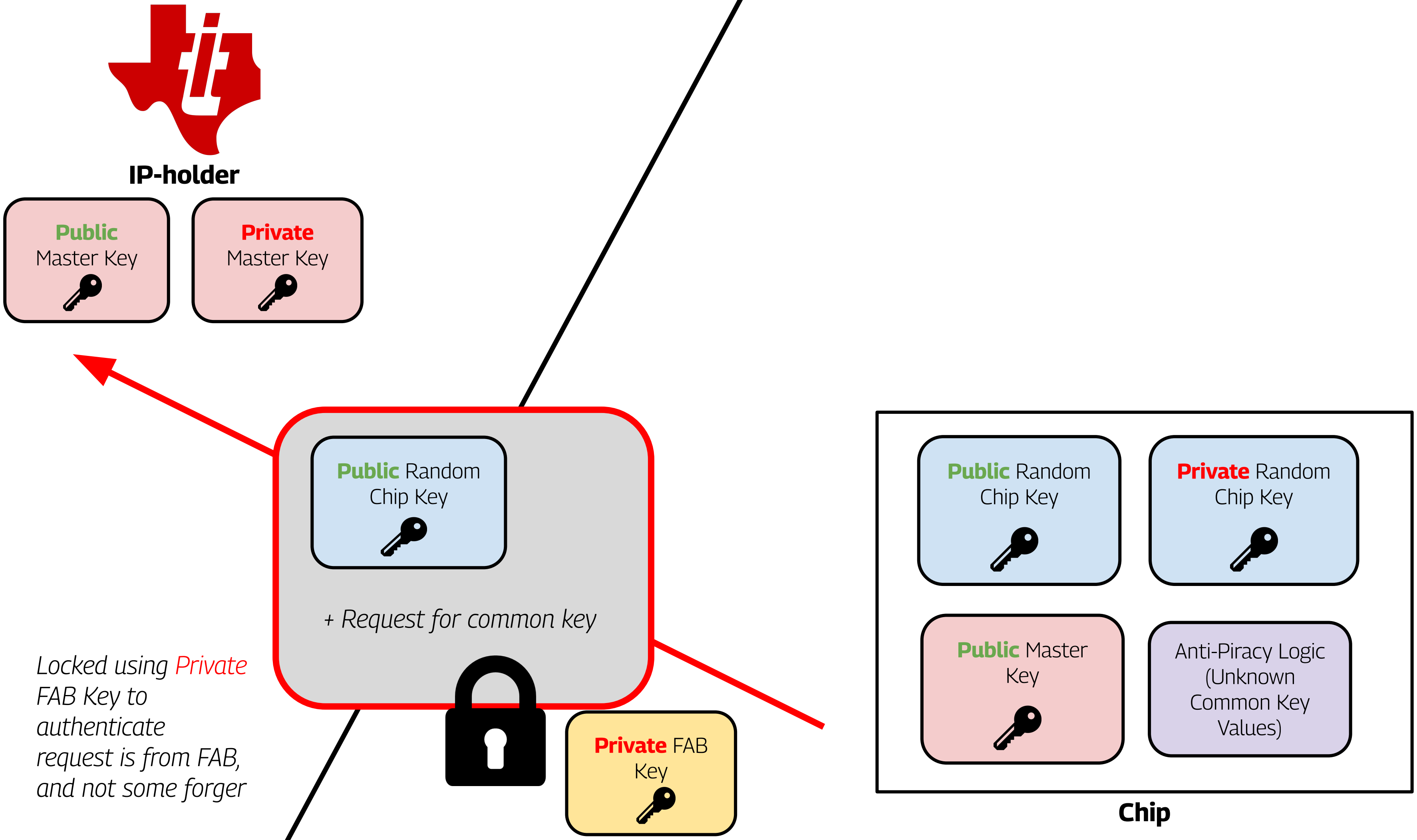
Public Random Chip Key

Private Random Chip Key

Public Master Key

Anti-Piracy Logic (Unknown Common Key Values)

Chip



IP-holder

Public Master Key

Private Master Key

Public Random Chip Key

+ Request for common key

Private FAB Key

Locked using Private FAB Key to authenticate request is from FAB, and not some forger

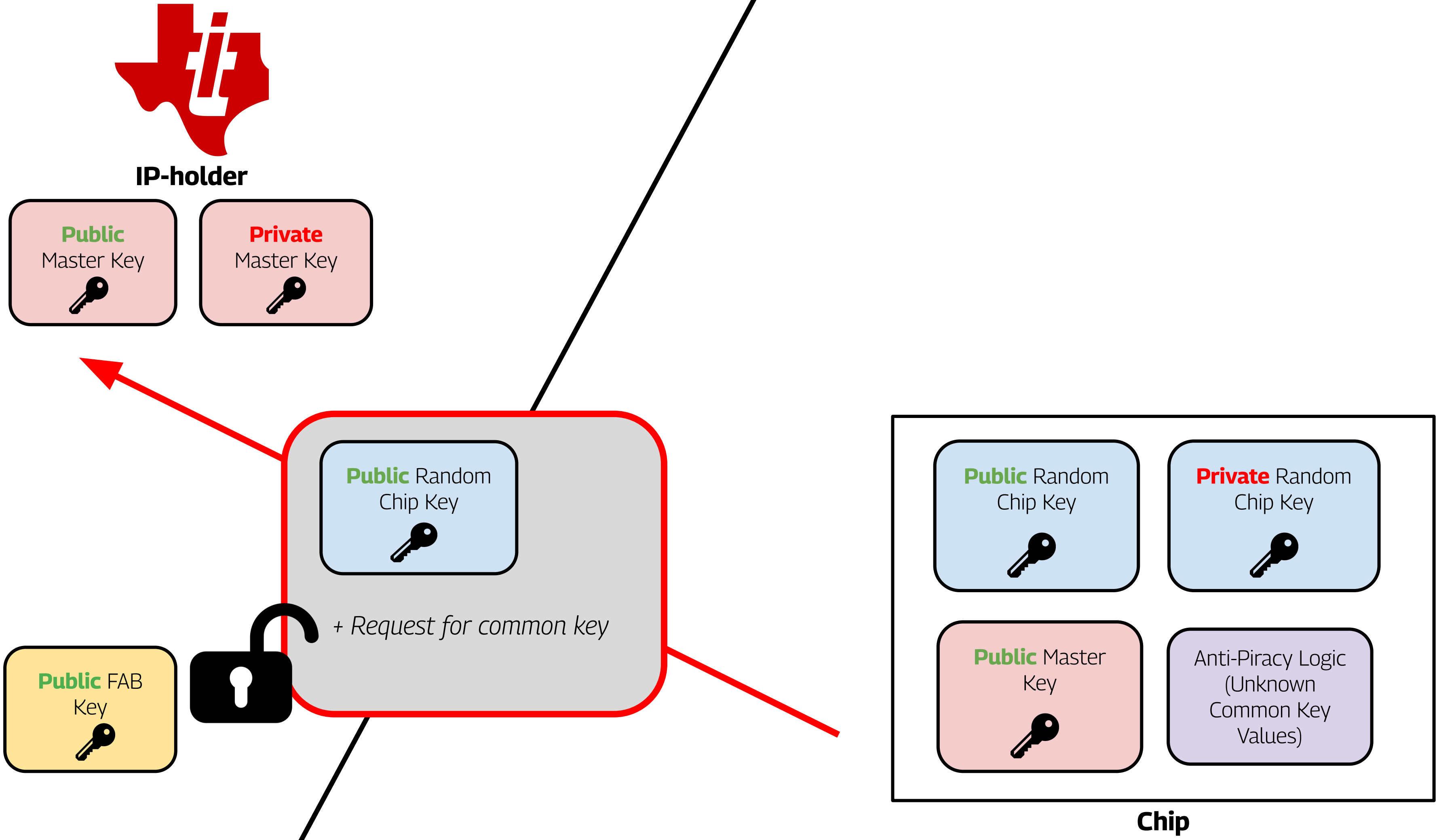
Public Random Chip Key

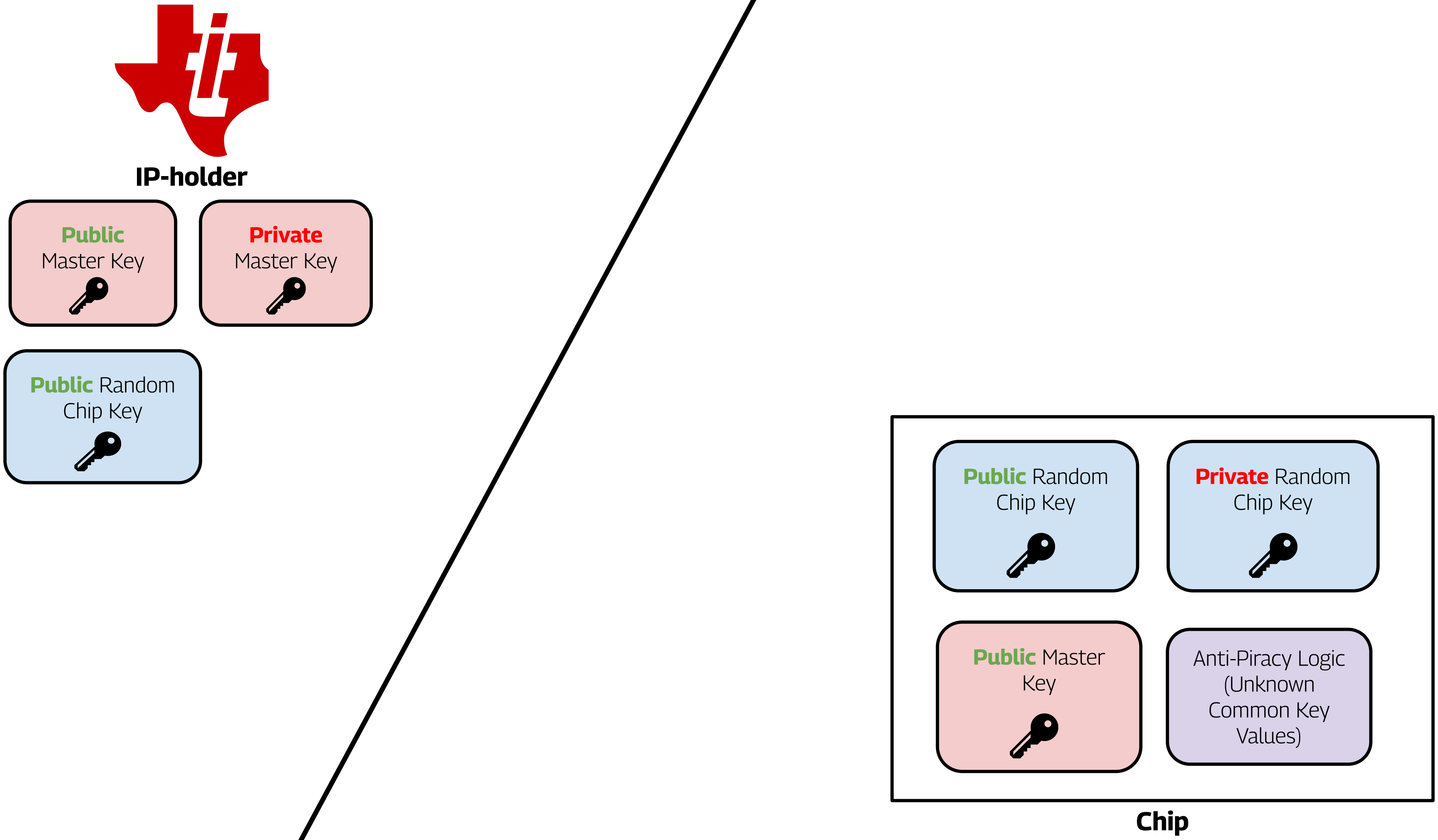
Private Random Chip Key

Public Master Key

Anti-Piracy Logic (Unknown Common Key Values)

Chip


IP-holder
Public Master Key
Private Master Key
Public Random Chip Key
+ Request for common key
Public FAB Key
Public Random Chip Key
Private Random Chip Key
Public Master Key
Anti-Piracy Logic (Unknown Common Key Values)
Chip



IP-holder

Public Master Key

Private Master Key

Public Random Chip Key

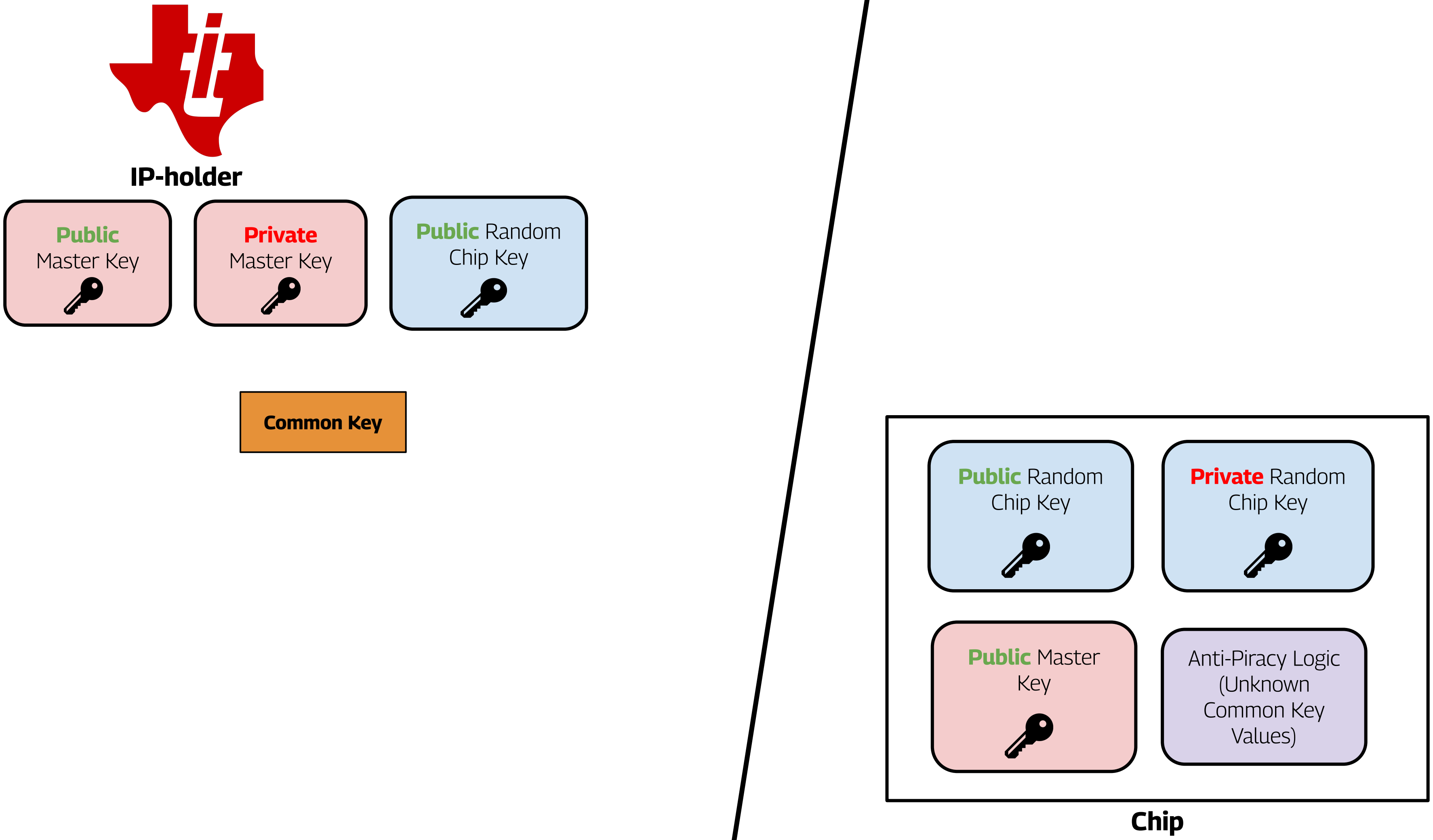
Public Random Chip Key

Private Random Chip Key

Public Master Key

Anti-Piracy Logic (Unknown Common Key Values)

Chip



IP-holder

Public Master Key

Private Master Key

Public Random Chip Key

Common Key

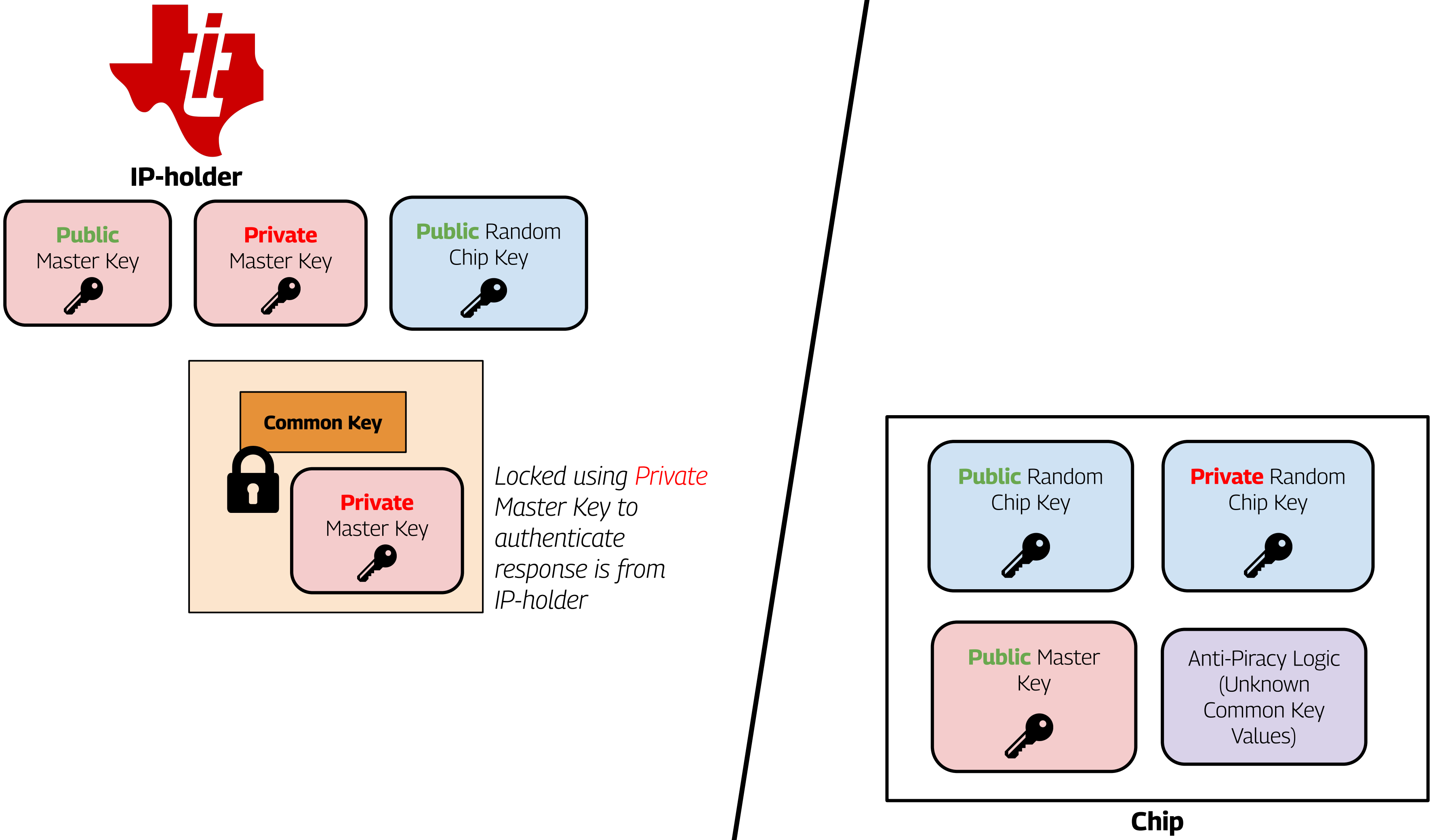
Public Random Chip Key

Private Random Chip Key

Public Master Key

Anti-Piracy Logic (Unknown Common Key Values)

Chip



IP-holder

Public Master Key

Private Master Key

Public Random Chip Key

Common Key

Private Master Key

Locked using Private Master Key to authenticate response is from IP-holder

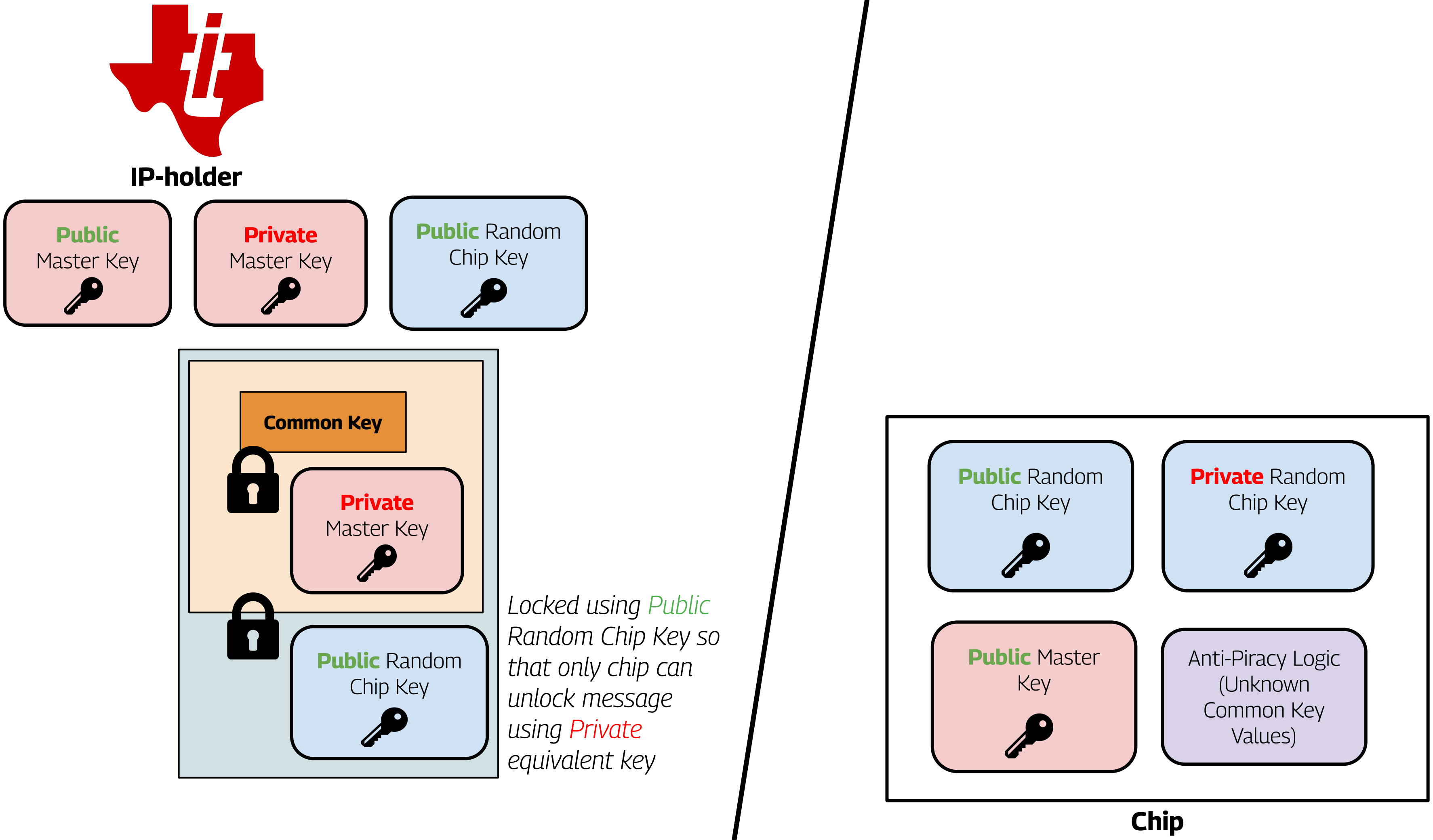
Public Random Chip Key

Private Random Chip Key

Public Master Key

Anti-Piracy Logic (Unknown Common Key Values)

Chip



IP-holder

Public Master Key

Private Master Key

Public Random Chip Key

Common Key

Private Master Key

Public Random Chip Key

Locked using Public Random Chip Key so that only chip can unlock message using Private equivalent key

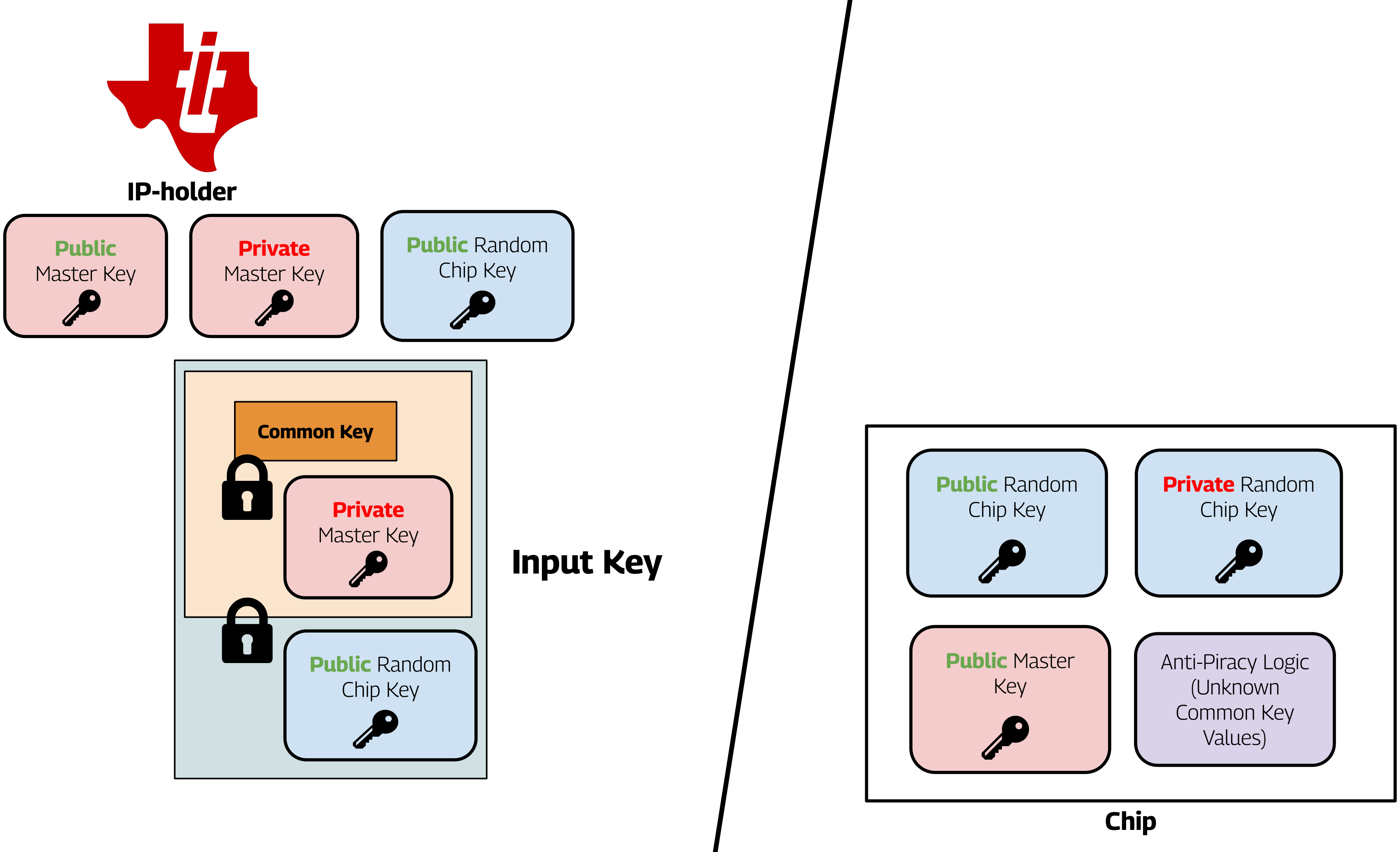
Public Random Chip Key

Private Random Chip Key

Public Master Key

Anti-Piracy Logic (Unknown Common Key Values)

Chip



IP-holder

Public Master Key

Private Master Key

Public Random Chip Key

Common Key

Private Master Key

Public Random Chip Key

Input Key

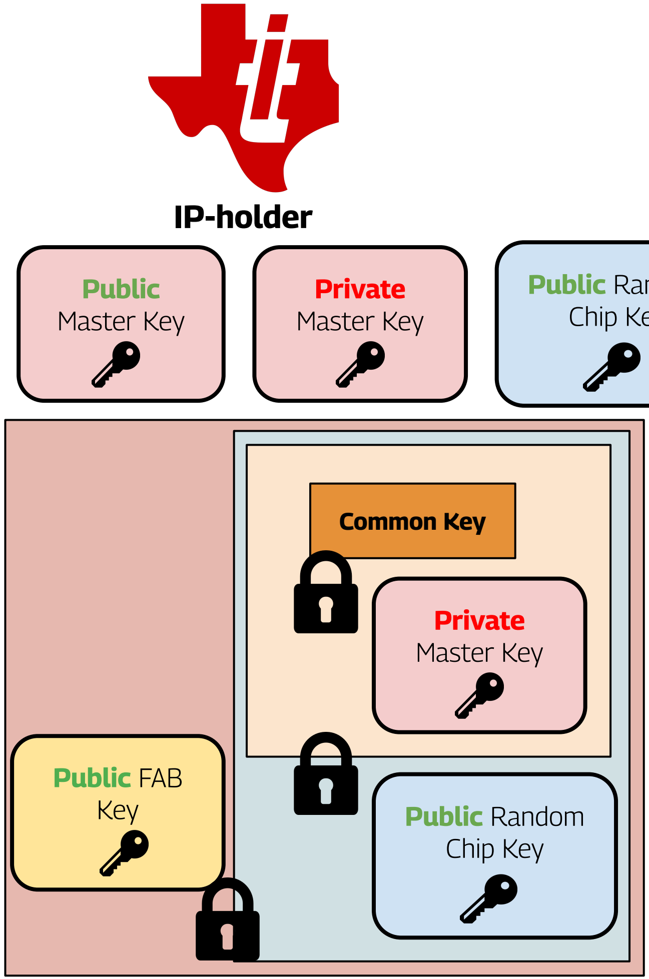
Public Random Chip Key

Private Random Chip Key

Public Master Key

Anti-Piracy Logic (Unknown Common Key Values)

Chip



IP-holder

Public Master Key

Private Master Key

Public Random Chip Key

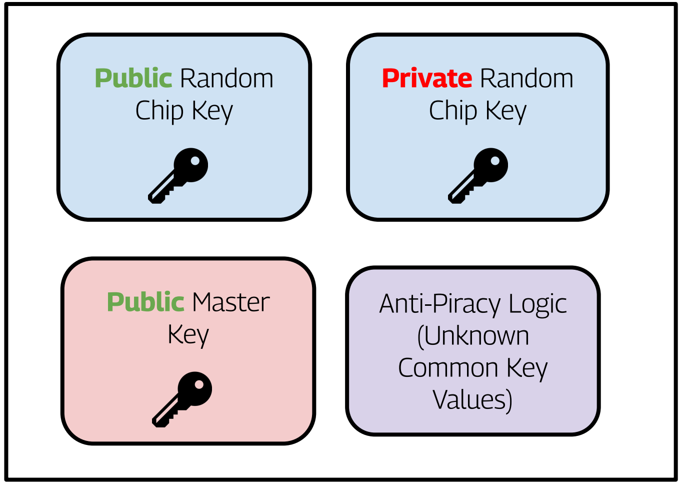
Common Key

Private Master Key

Public Random Chip Key

Public FAB Key

Input Key Locked using Public FAB Key so that only FAB can unlock message using Private equivalent key (added security measure)



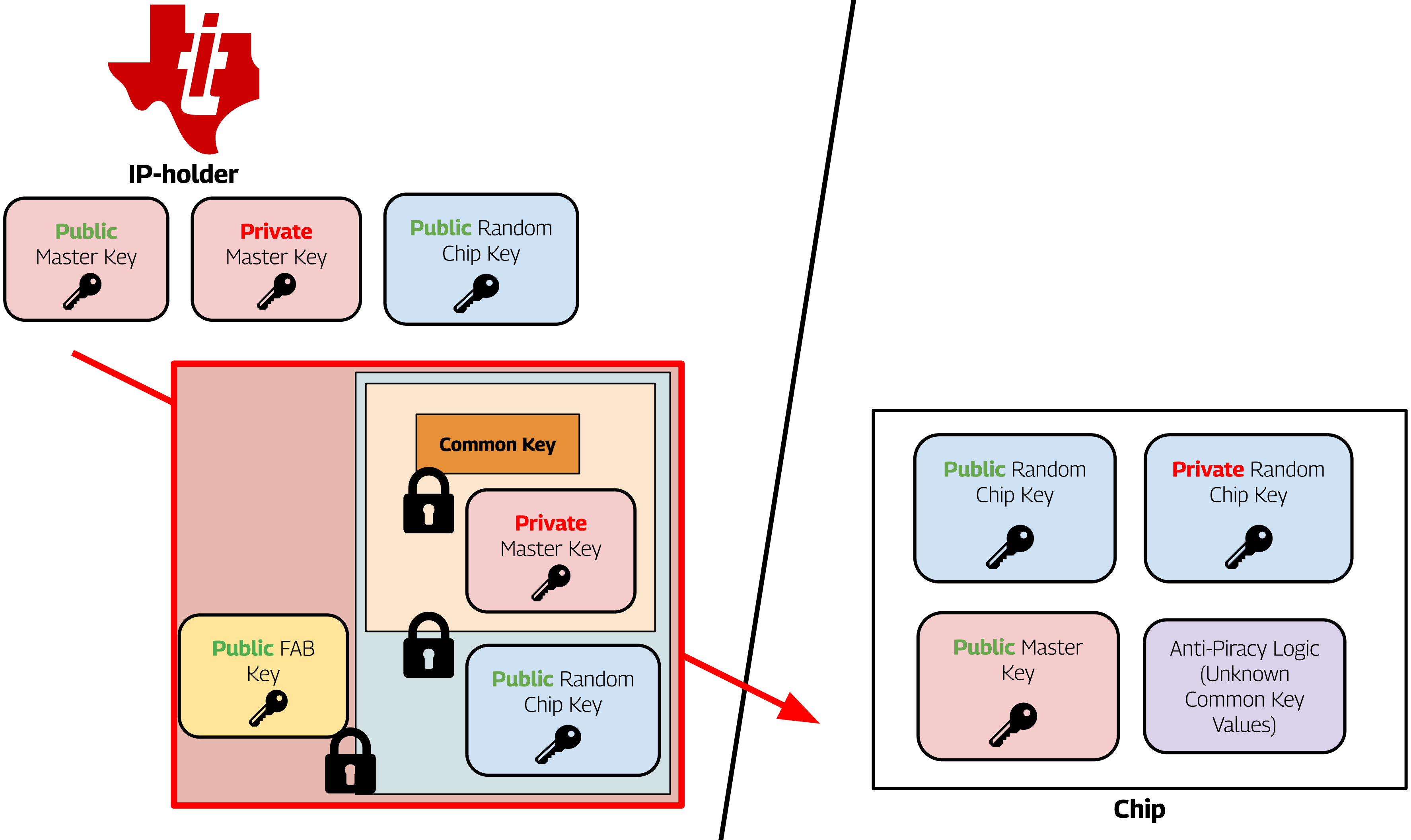
Public Random Chip Key

Private Random Chip Key

Public Master Key

Anti-Piracy Logic (Unknown Common Key Values)

Chip



IP-holder

Public Master Key

Private Master Key

Public Random Chip Key

Common Key

Private Master Key

Public FAB Key

Public Random Chip Key

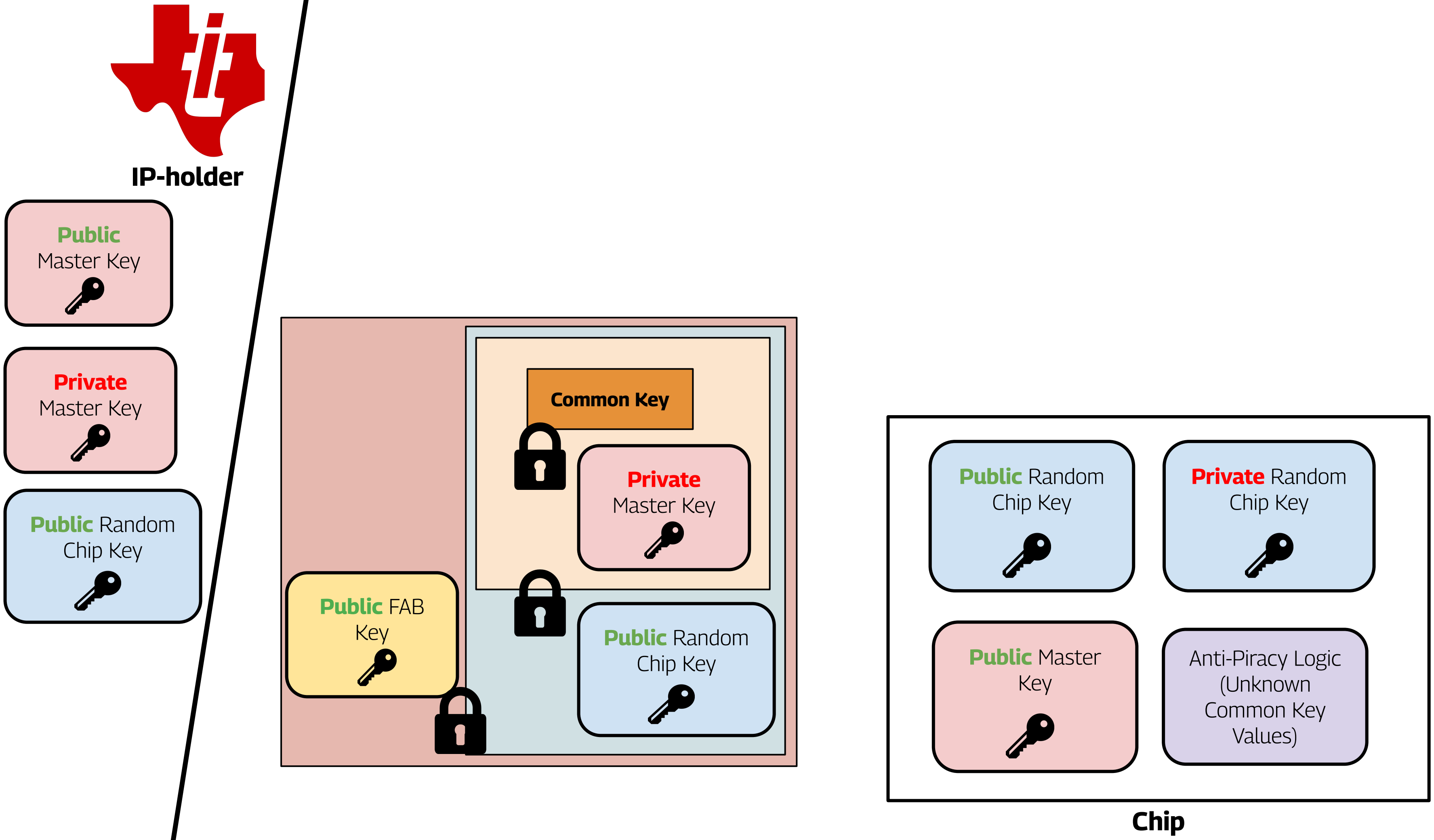
Public Random Chip Key

Private Random Chip Key

Public Master Key

Anti-Piracy Logic (Unknown Common Key Values)

Chip



IP-holder

Public Master Key

Private Master Key

Public Random Chip Key

Common Key

Private Master Key

Public FAB Key

Public Random Chip Key

Public Random Chip Key

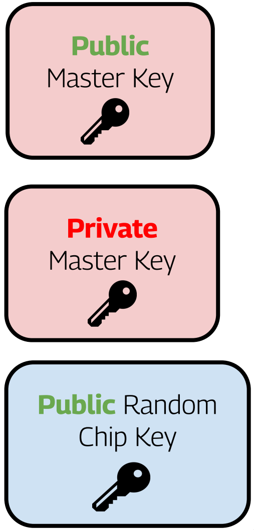
Private Random Chip Key

Public Master Key

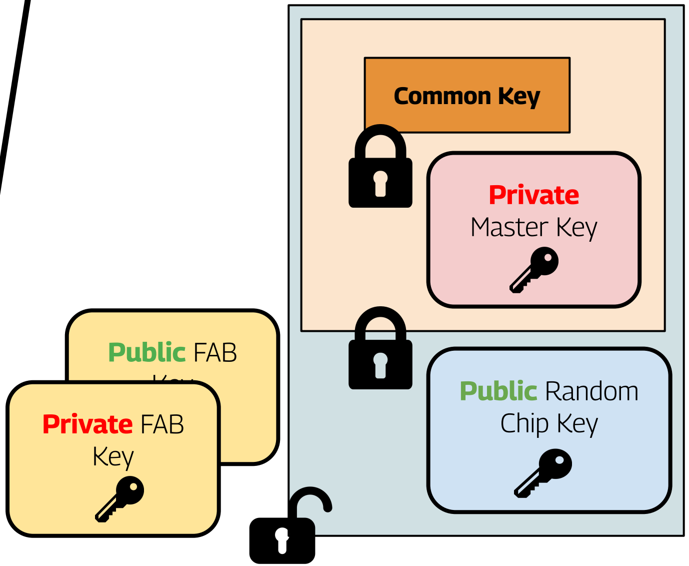
Anti-Piracy Logic (Unknown Common Key Values)

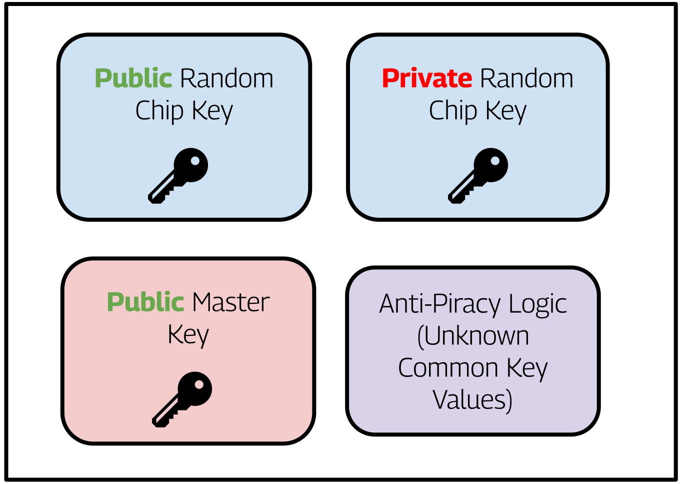
Chip

IP-holder



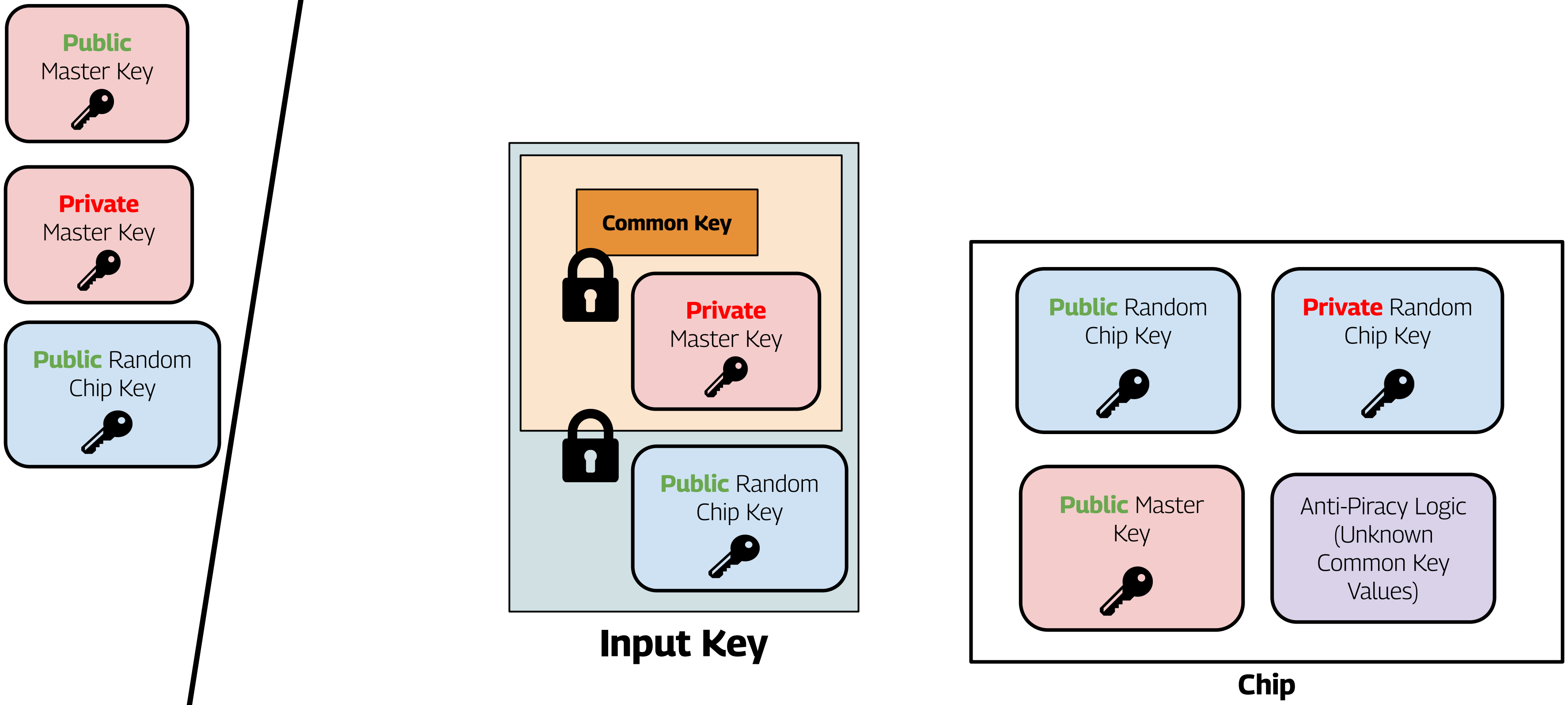
Okay, even if someone intercepted this message, only I should be able to decrypt the lock set by the Public FAB Key and receive the Input Key.





Chip

IP-holder



Public Master Key

Private Master Key

Public Random Chip Key

Common Key

Private Master Key

Public Random Chip Key

Input Key

Public Random Chip Key

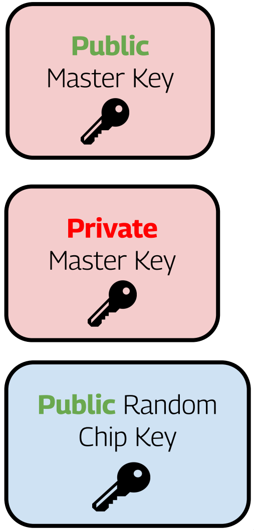
Private Random Chip Key

Public Master Key

Anti-Piracy Logic (Unknown Common Key Values)

Chip

IP-holder

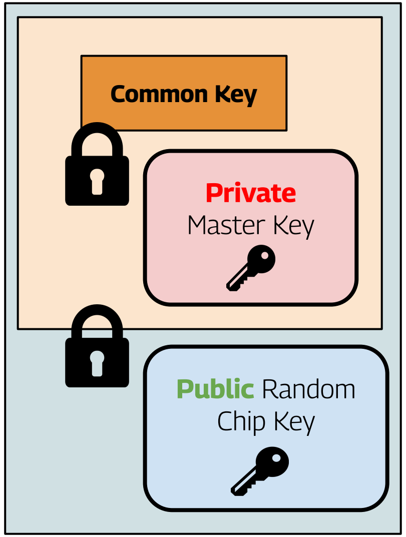


Public Master Key

Private Master Key

Public Random Chip Key

*Time to enter the **input key** into the chip!*

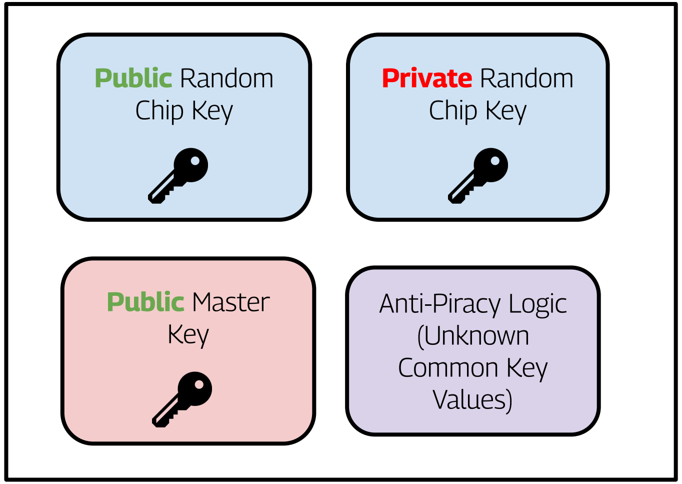


Common Key

Private Master Key

Public Random Chip Key

Input Key



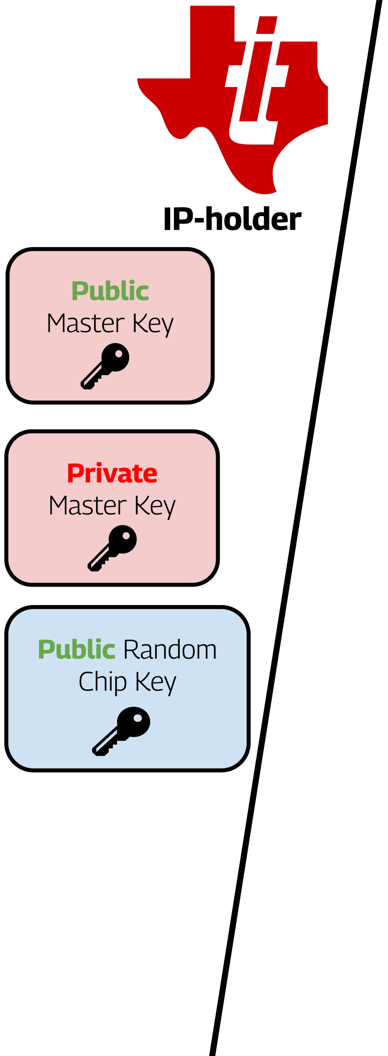
Public Random Chip Key

Private Random Chip Key

Public Master Key

Anti-Piracy Logic (Unknown Common Key Values)

Chip



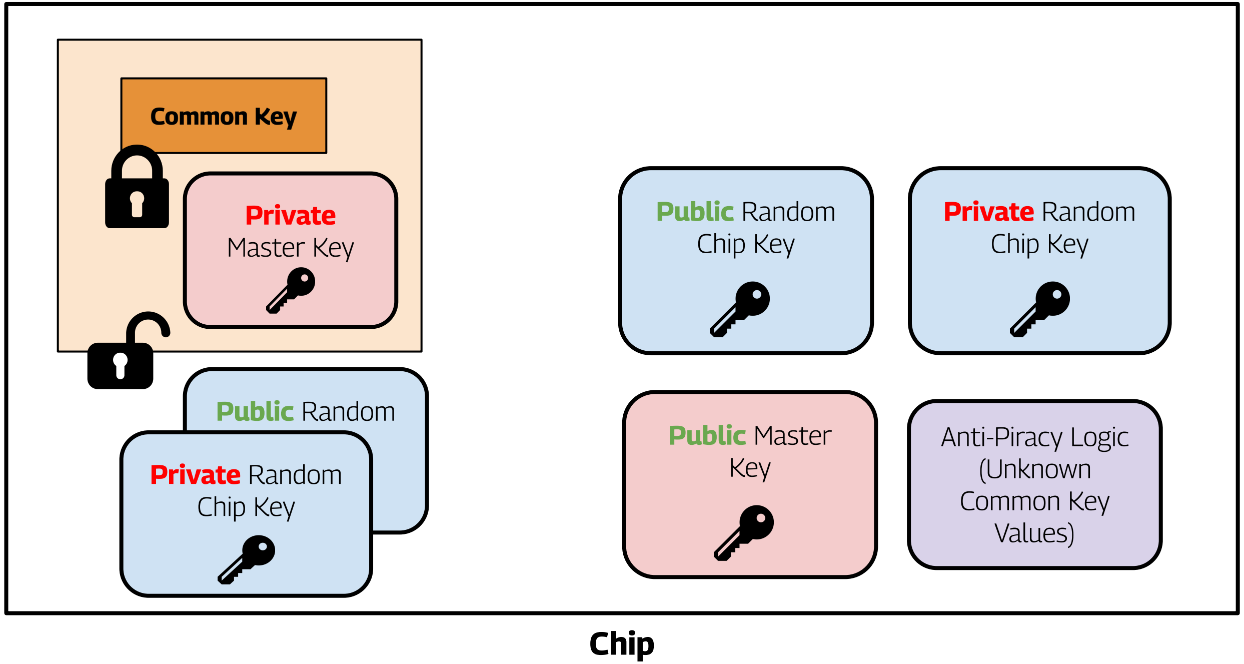
IP-holder

Public Master Key

Private Master Key

Public Random Chip Key

The Input Key is decrypted using the Private Random Chip Key: I should also be the only chip to be able to decrypt this lock!



Common Key

Private Master Key

Public Random

Private Random Chip Key

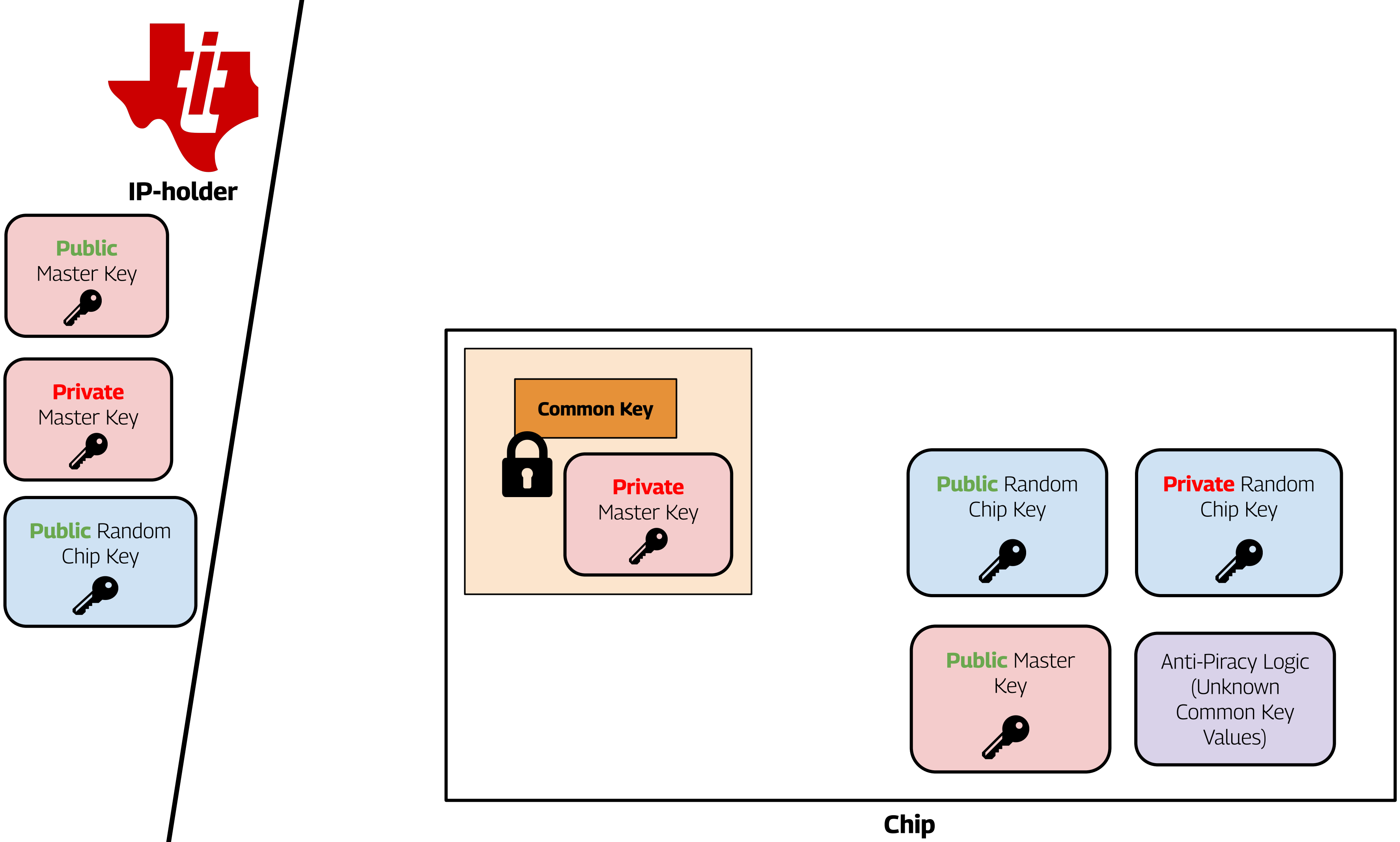
Public Random Chip Key

Private Random Chip Key

Public Master Key

Anti-Piracy Logic (Unknown Common Key Values)

Chip



IP-holder

Public Master Key

Private Master Key

Public Random Chip Key

Common Key

Private Master Key

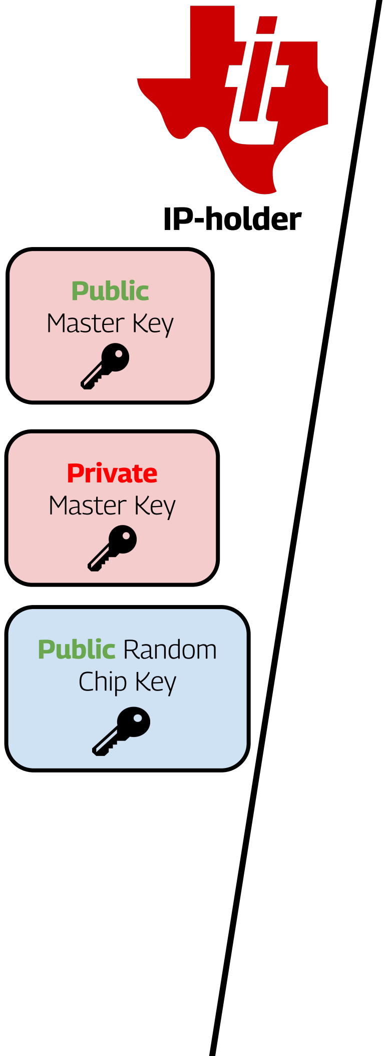
Public Random Chip Key

Private Random Chip Key

Public Master Key

Anti-Piracy Logic (Unknown Common Key Values)

Chip



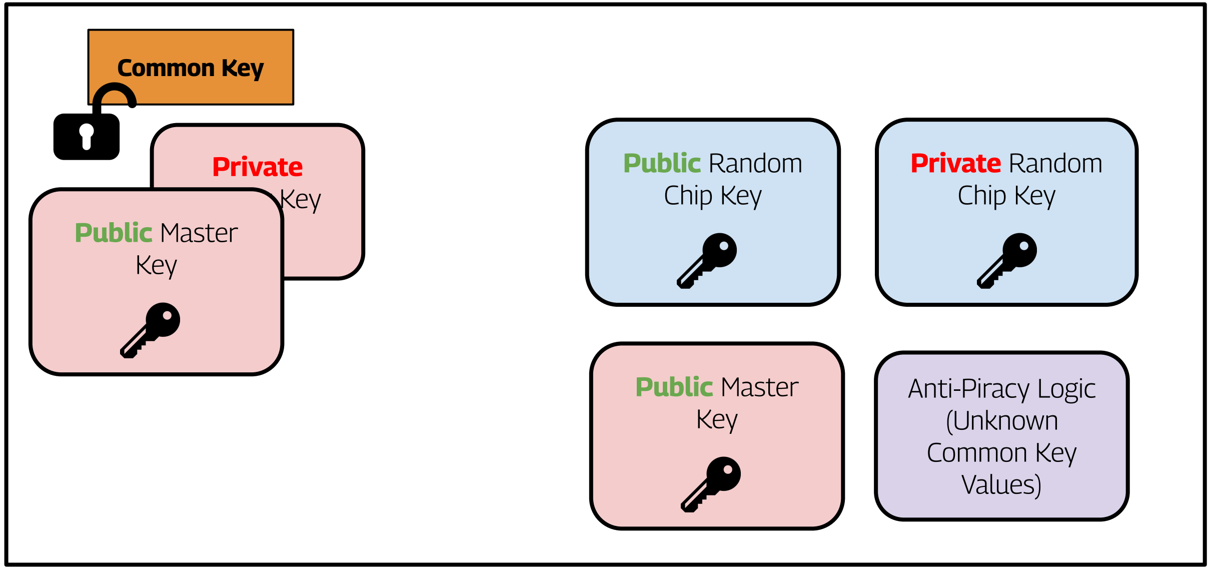
IP-holder

Public Master Key

Private Master Key

Public Random Chip Key

This should help me confirm that this Common Key is definitely from the IP-holder as their public Master Key can unlock it!



Common Key

Private Key

Public Master Key

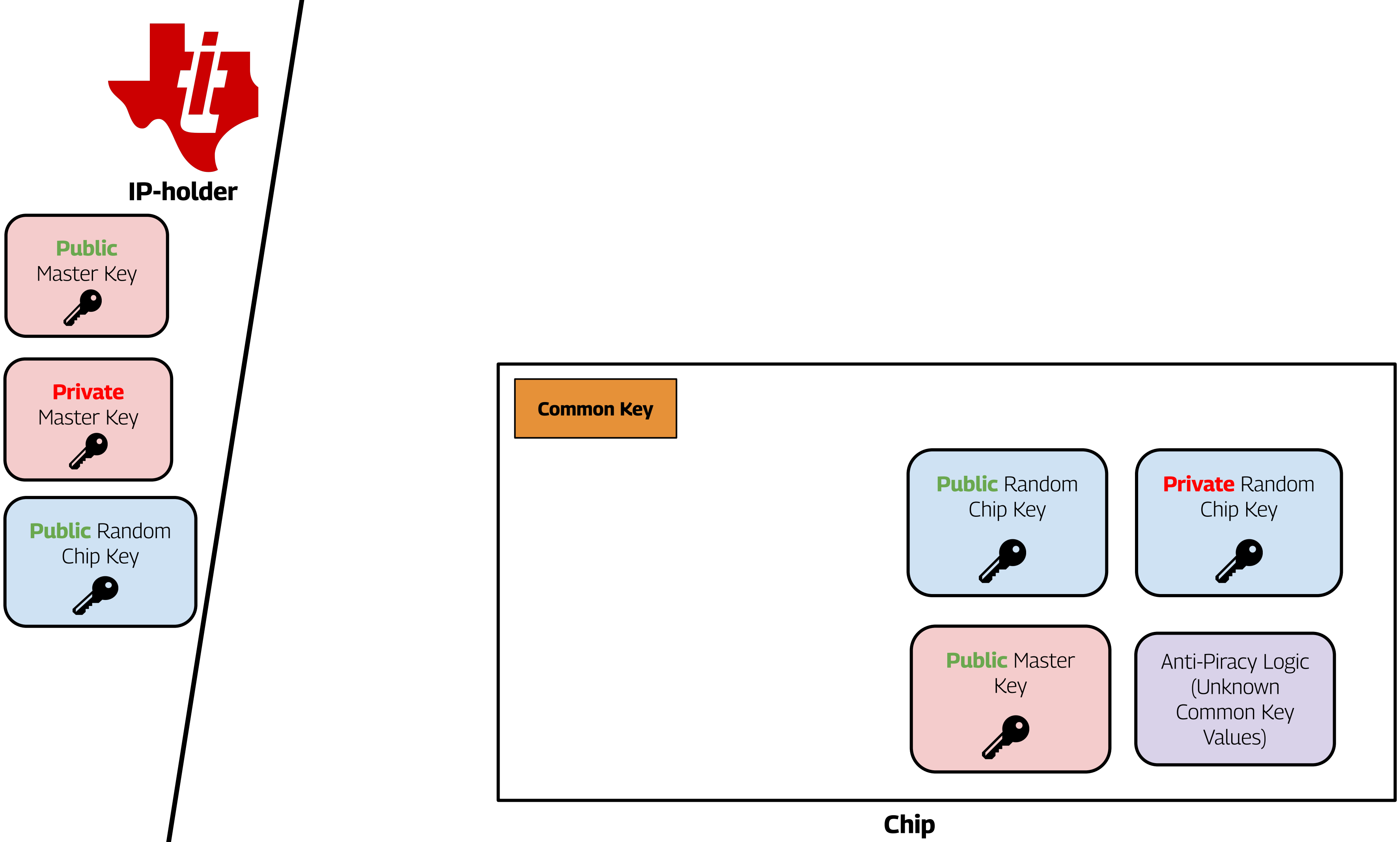
Public Random Chip Key

Private Random Chip Key

Public Master Key

Anti-Piracy Logic (Unknown Common Key Values)

Chip



IP-holder

Public Master Key

Private Master Key

Public Random Chip Key

Common Key

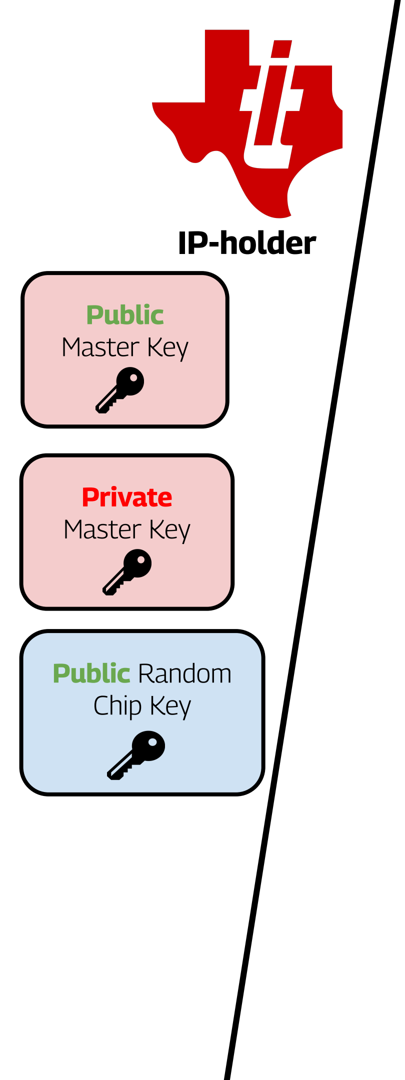
Public Random Chip Key

Private Random Chip Key

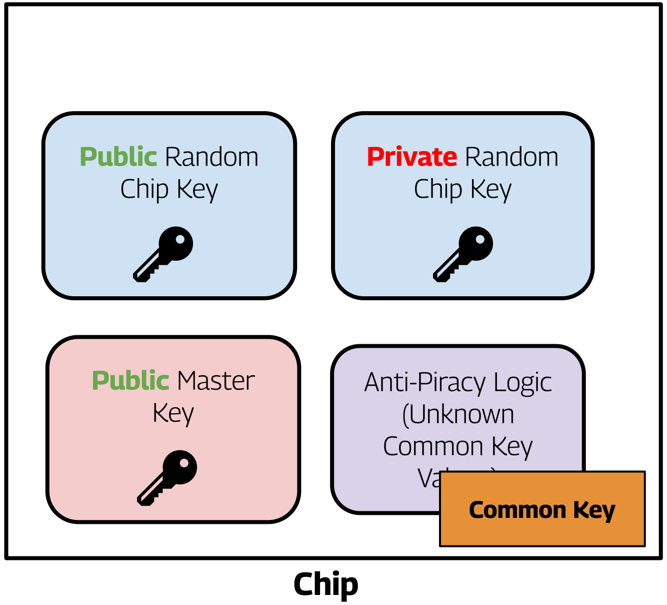
Public Master Key

Anti-Piracy Logic (Unknown Common Key Values)

Chip



Upon decryption, CK unlocks the chip and facilitates test. After that, the chip can be sold.



(***obstacles to piracy***)

1. **Lack of information**, e.g., not being able to obtain private Master Key because it is never transmitted and kept with IP-holder
    a. Public Random Chip Key, Private Master Key, and Common Key are not present in RTL or synthesized gate-level netlist, while former 2 are not present in masks either.

2. **Computational complexity**, e.g., not being able to break RSA-style public-key crypto-systems.
    a. Computational attacks seeking the common key would be costly (NP-complete)

3. **Technological barriers**, e.g. not being able to reverse-engineer the activate layers of 45nm ICs or masks.
    a. Common Key may conceivably be discovered by watching transient signals on an activated chip, but for 45nm chips that would require very sophisticated technology.

4. **Financial barriers**, e.g., not being able to invest amounts larger than expected revenue from piracy.

*(**evaluation**)*

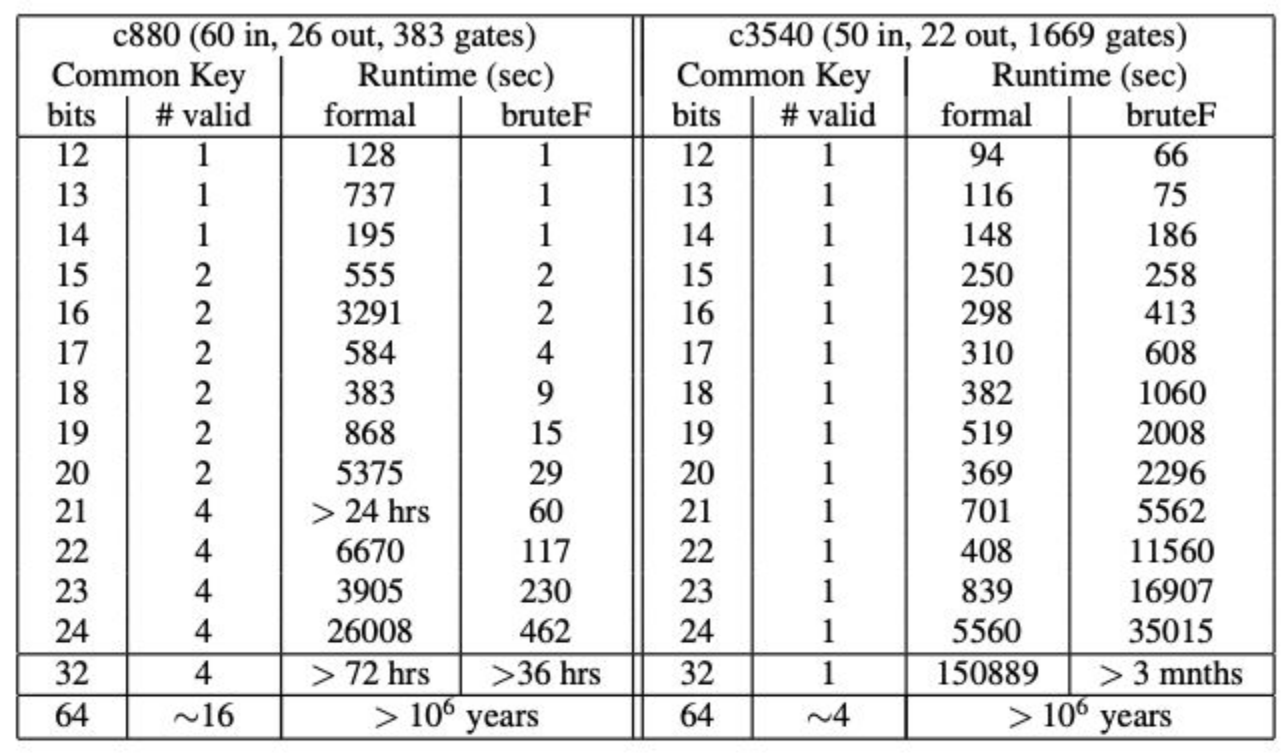
- One of the most serious attacks on EPIC is the theft of Common Key and Private Master Key from the holder of IP rights.

- Combinational locking does **not** affect critical path delays.

- **Area and power overhead is minor**—even with the on-chip true random number generator.

- EPIC does not require significant changes to establish verification and testing flows
    - unlocked integrated circuit behaves just like the original integrated circuit.

(***empirical results***)



c880 (60 in, 26 out, 383 gates)				c3540 (50 in, 22 out, 1669 gates)			
Common Key		Runtime (sec)		Common Key		Runtime (sec)	
bits	# valid	formal	bruteF	bits	# valid	formal	bruteF
12	1	128	1	12	1	94	66
13	1	737	1	13	1	116	75
14	1	195	1	14	1	148	186
15	2	555	2	15	1	250	258
16	2	3291	2	16	1	298	413
17	2	584	4	17	1	310	608
18	2	383	9	18	1	382	1060
19	2	868	15	19	1	519	2008
20	2	5375	29	20	1	369	2296
21	4	> 24 hrs	60	21	1	701	5562
22	4	6670	117	22	1	408	11560
23	4	3905	230	23	1	839	16907
24	4	26008	462	24	1	5560	35015
32	4	> 72 hrs	>36 hrs	32	1	150889	> 3 mnths
64	~16	$> 10^6$ years		64	~4	$> 10^6$ years	

formal: SAT solver alone would be insufficient to find a key combination of non-trivial length

Reduced Ordered Binary Decision Diagram (ROBDD) is used to represent Boolean functions in a compact, canonical form by systematically eliminating redundancies, thereby reducing the search space.

Supplementary Material

(*EPIC guarantees*)

The following logistical properties of EPIC can be deduced:

1. Public Random Chip Key and Public Master Key do not reveal information about their private counterparts.
2. Knowing Common Key, all public keys, and both Random Chip Keys is insufficient to generate Input Key (irreversibility of public-key cryptography)
3. There are as many good Common Keys as good Input Keys.
4. Good Input Keys are as random as Random Chip Keys.

Additional properties of EPIC hold when forgers cannot modify masks or Integrated Circuits (but may have access to source files):

5. Different Integrated Circuits nearly always have different Random Chip Keys (True Random Number Generator)
6. Knowing a valid Common Key is not sufficient to unlock multiple chips.
7. Different chips nearly always have different Input Keys. Eavesdropping on data exchanged during activation of a chip will not reveal Input Keys for other chips.
8. A chip can only be unlocked by entering an appropriate Input Key.