Hardware Trojans

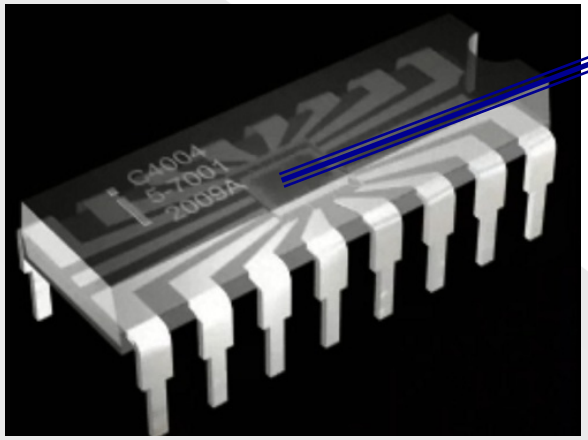
Kanad Basu
UT Dallas

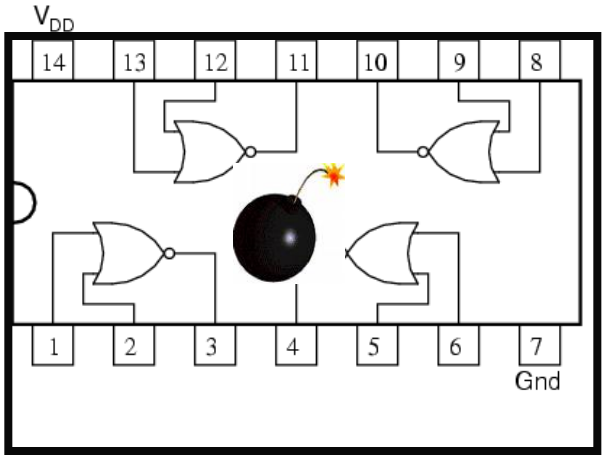
Some slides courtesy: Profs. Karri, Tehranipoor, Rajendran, Bhunia, and Makris

What is a Hardware Trojan?

- malicious and deliberate change to an integrated circuit (IC) design that may cause unwanted effects
- activation of the Trojan enables the plaintext to bypass the cryptographic hardware

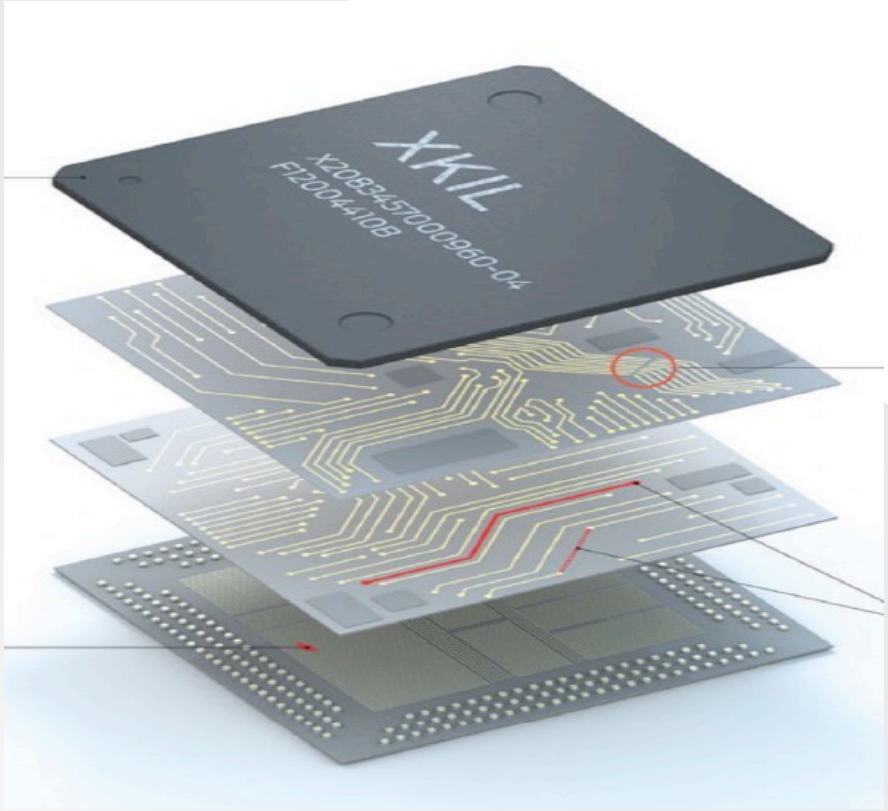
What is a Hardware Trojan?



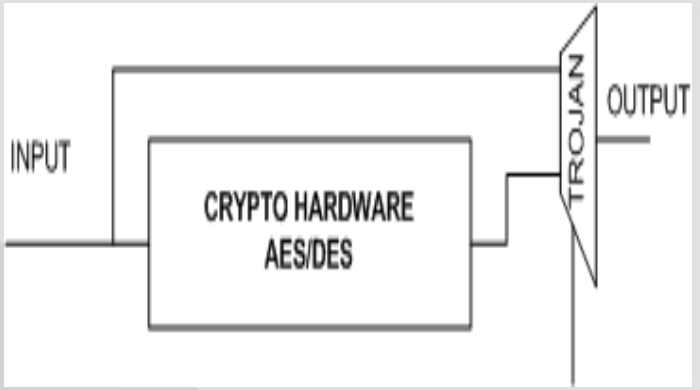


- Malicious modifications to design
  - Causes IC to malfunction in-field
  - Can be inserted during design or fabrication
  - Inserted by an intelligent adversary
  - *Stealthy* => difficult to detect
- Results
  - Potentially disastrous consequences e.g. civilian critical infrastructures

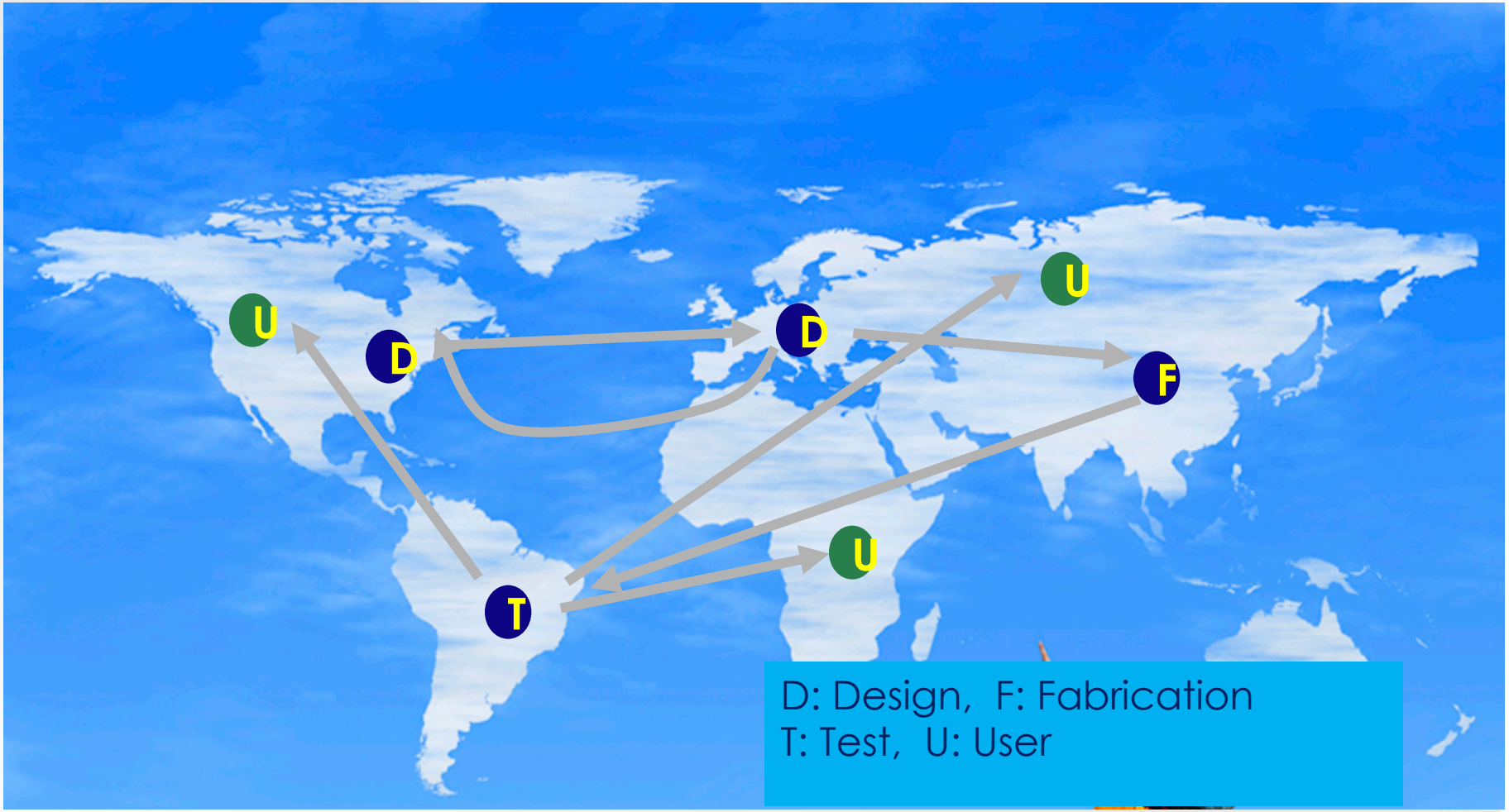
The kill switch ?



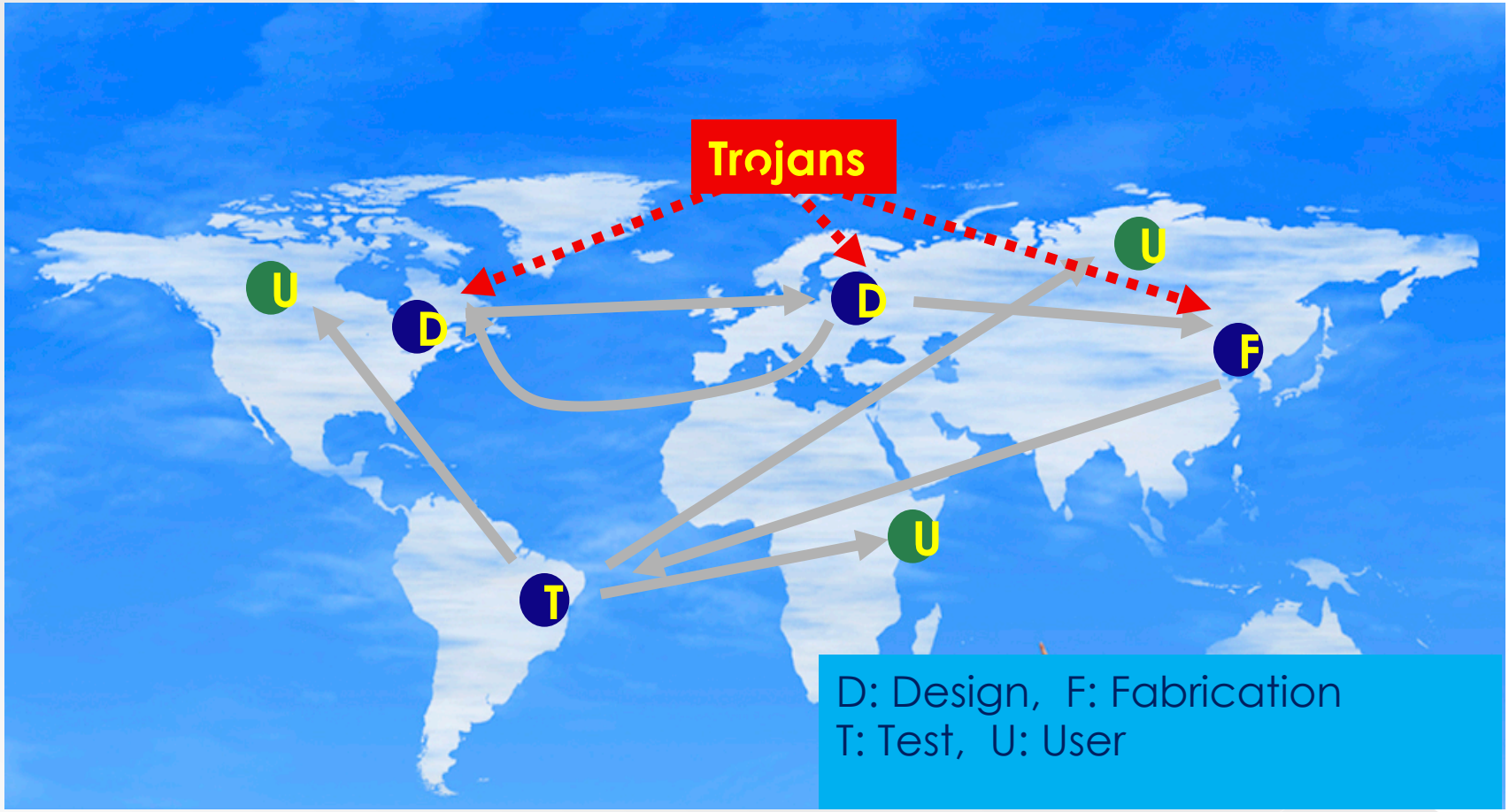
IEEE Spectrum, 2008



Background: IC design process

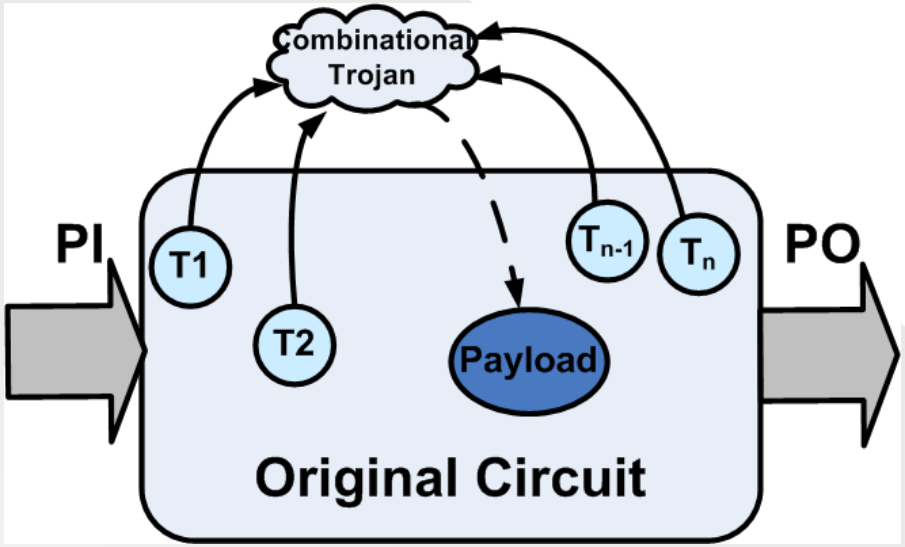


Hardware Trojans

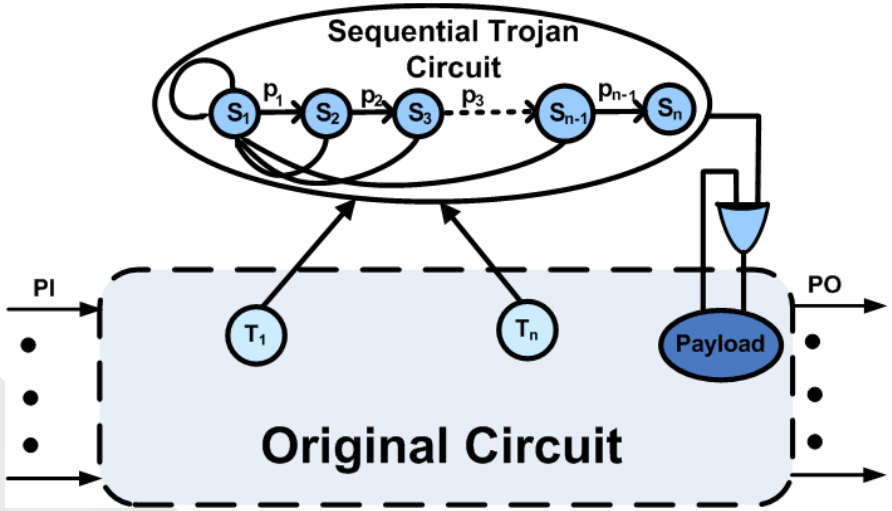


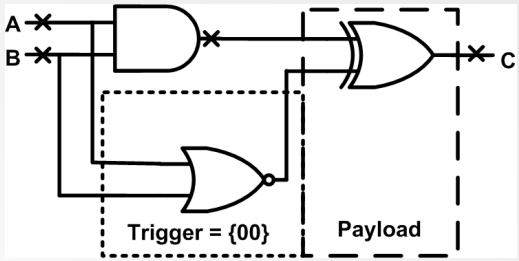
Hardware trojan: examples

Combinational Trojan model

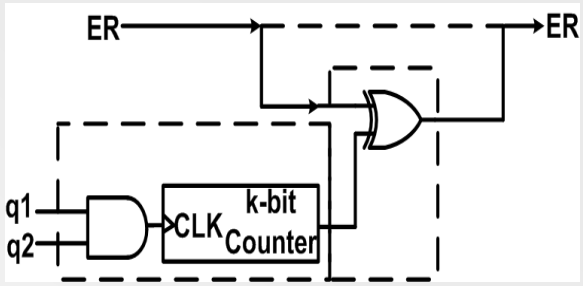


Sequential Trojan Model

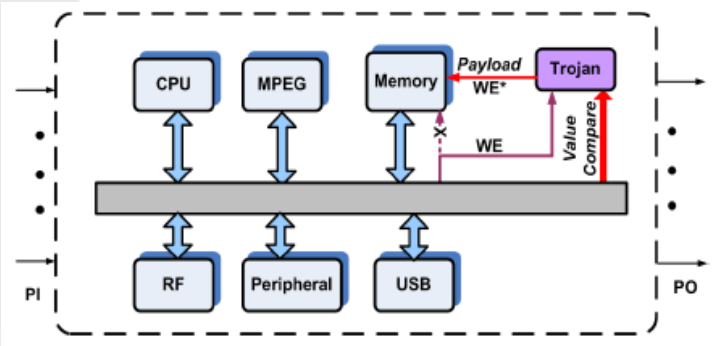




Combinational Trojan example



Sequential Trojan example (time-bomb)



System level view

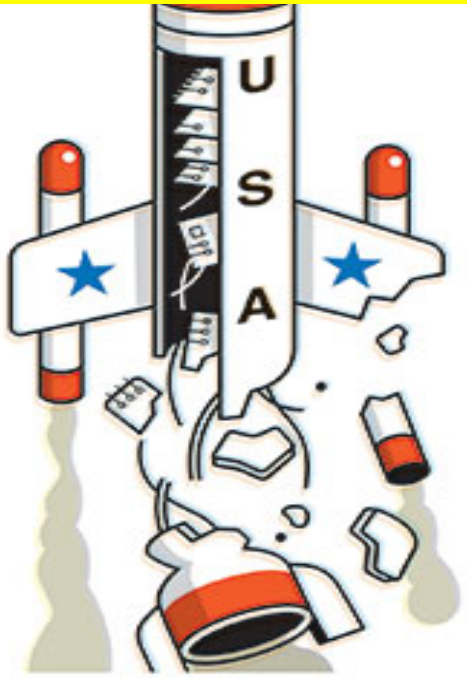
CYBERWAR

Old Trick Threatens the Newest Weapons

By JOHN MARKOFF
Published: October 26, 2009

Despite a six-year effort to build trusted computer chips for military systems, the Pentagon now manufactures in secure facilities run by American companies only about 2 percent of the more than $3.5 billion of integrated circuits bought annually for use in military gear.

Only 2% of ~$3.5 billion of DoD ICs manufactured in trusted foundries !!



Harry Campbell

executives who argue that the menace of so-called Trojan horses hidden in equipment circuitry is among the most severe threats the nation faces in the event of a war in which communications and weaponry rely on computer technology.

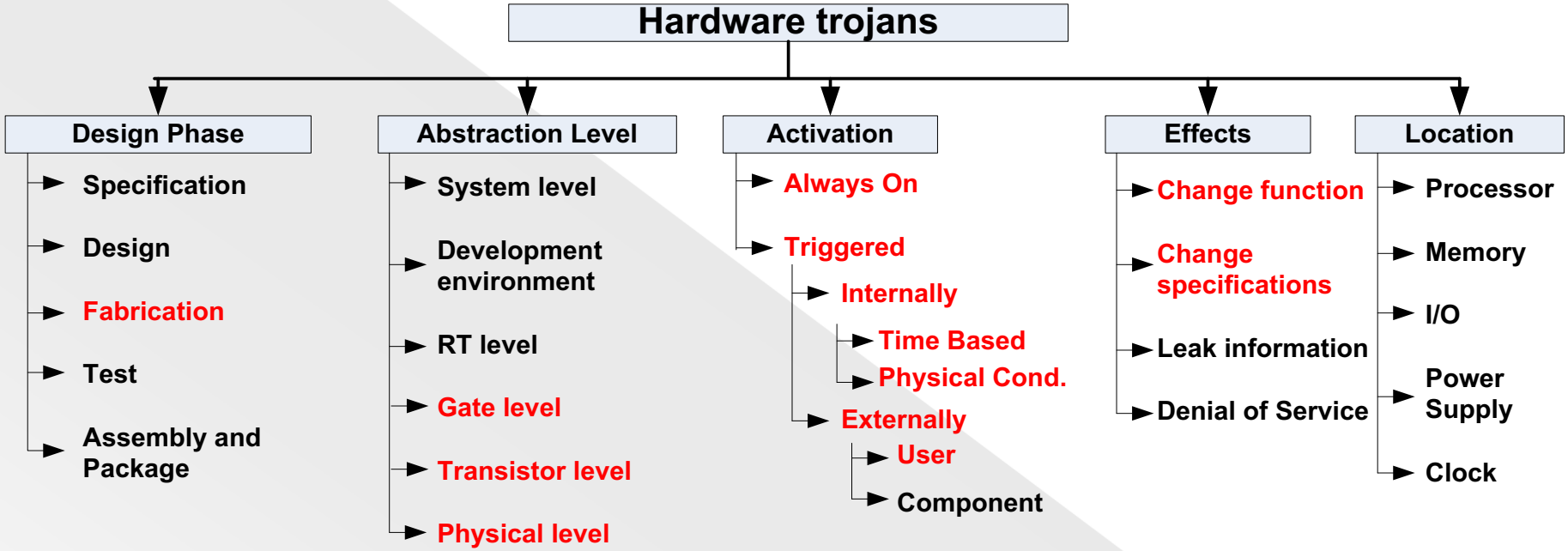
As advanced systems like aircraft, missil become dependent on their computing c specter of subversion causing weapons t crisis, or secretly corrupting crucial data haunt military planners. The problem ha severe as most American semiconductor

Are hardware trojans realistic?

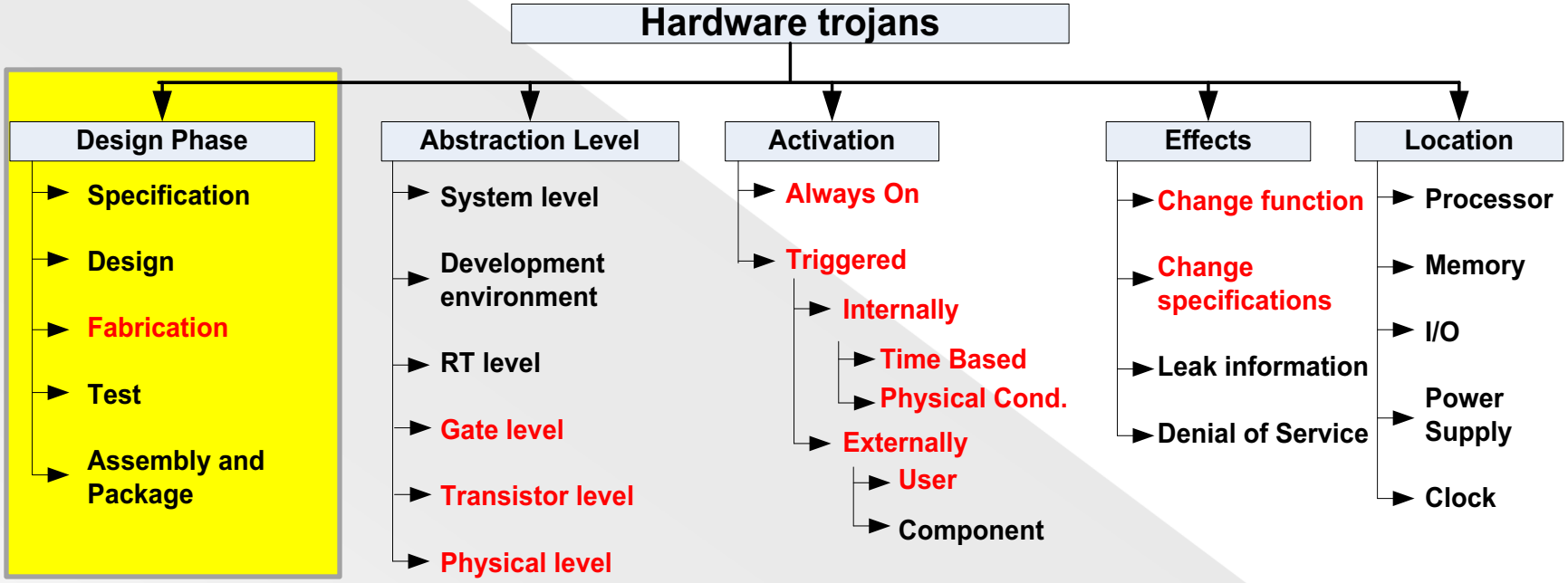
- Reality
  - Tampering masks in fab is not easy (highly complex)
  - Reverse-engineering a single IC can take months
  - Political issues make it difficult to verify authenticity of fabs
- But there is strong evidence they do….
  - Numerous unexplained military mishaps*
  - Reputed companies reverse-engineering ICs
  - Specialist (legal) reverse-engineers do exist** (*Chipworks*, *IBM*, etc.)
  - Sophisticated software tools makes reverse-engineering possible
- Third part intellectual property !!
  - More likely
  - Easy to insert

*S.Adee, *IEEE Spectrum,* vol. 45, no. 5, 2008.
**J.Kumagai, *IEEE Spectrum,* vol. 37, no. 11, 2000.

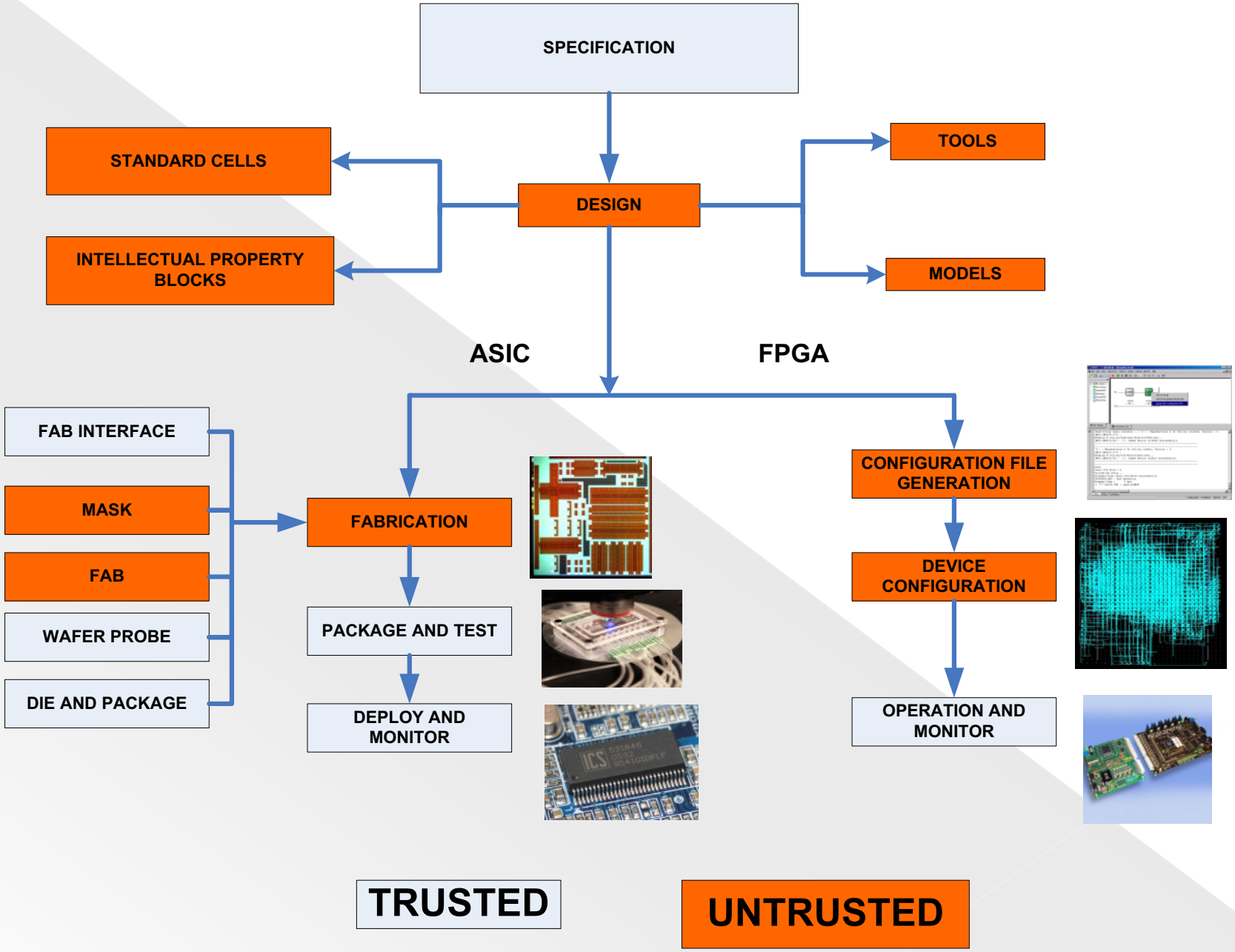
Trojan insertion taxonomy



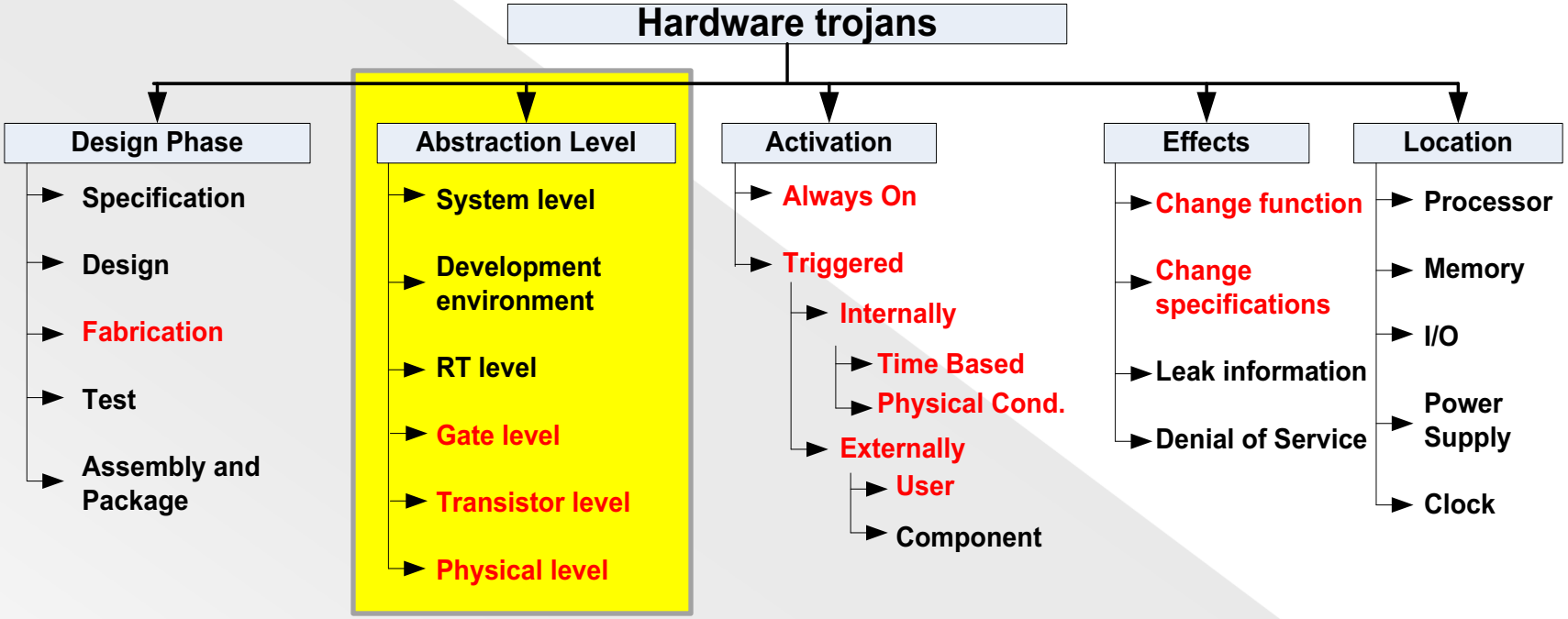
Design phase



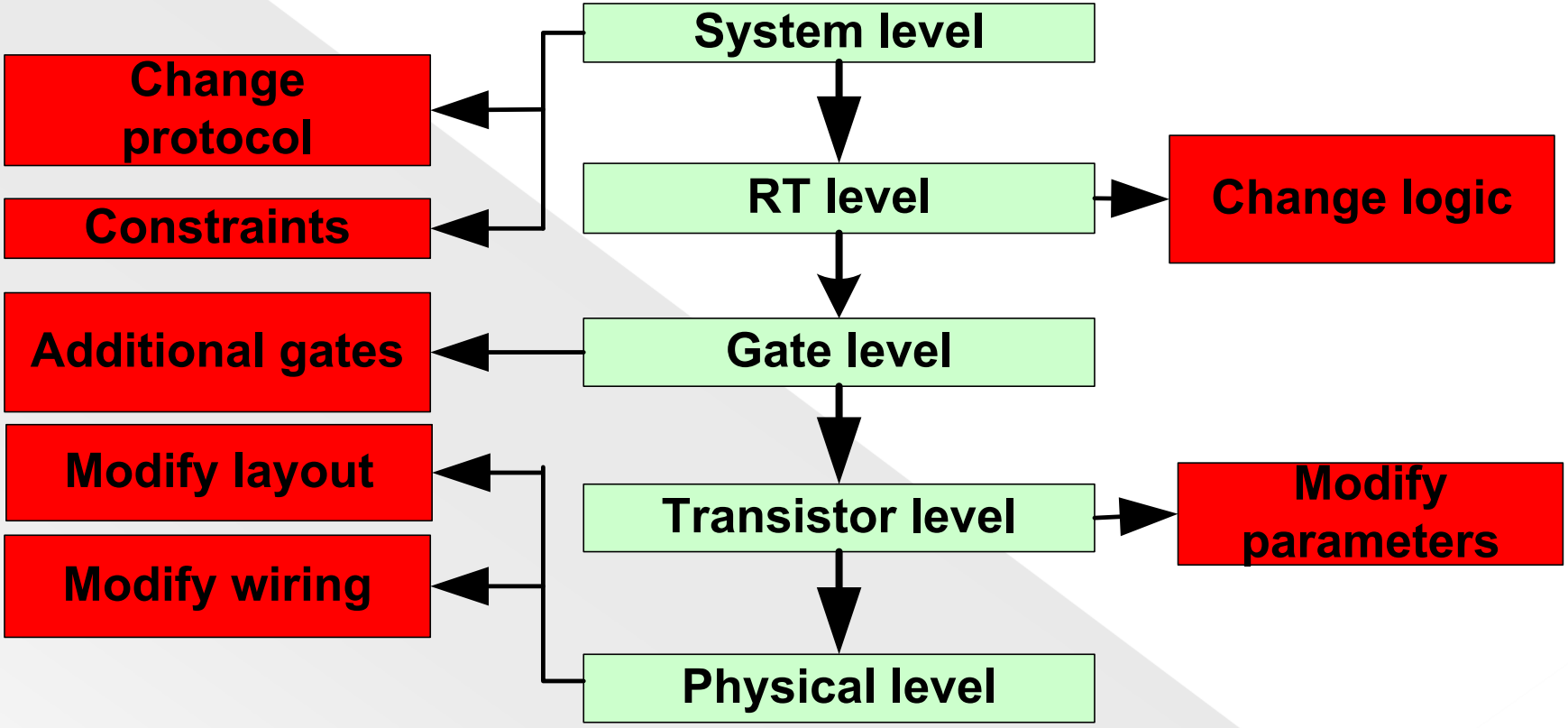
Design phase



Hardware abstraction level

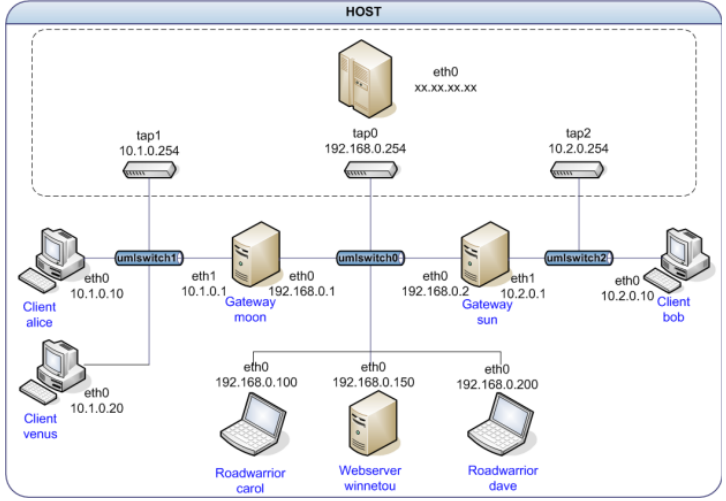


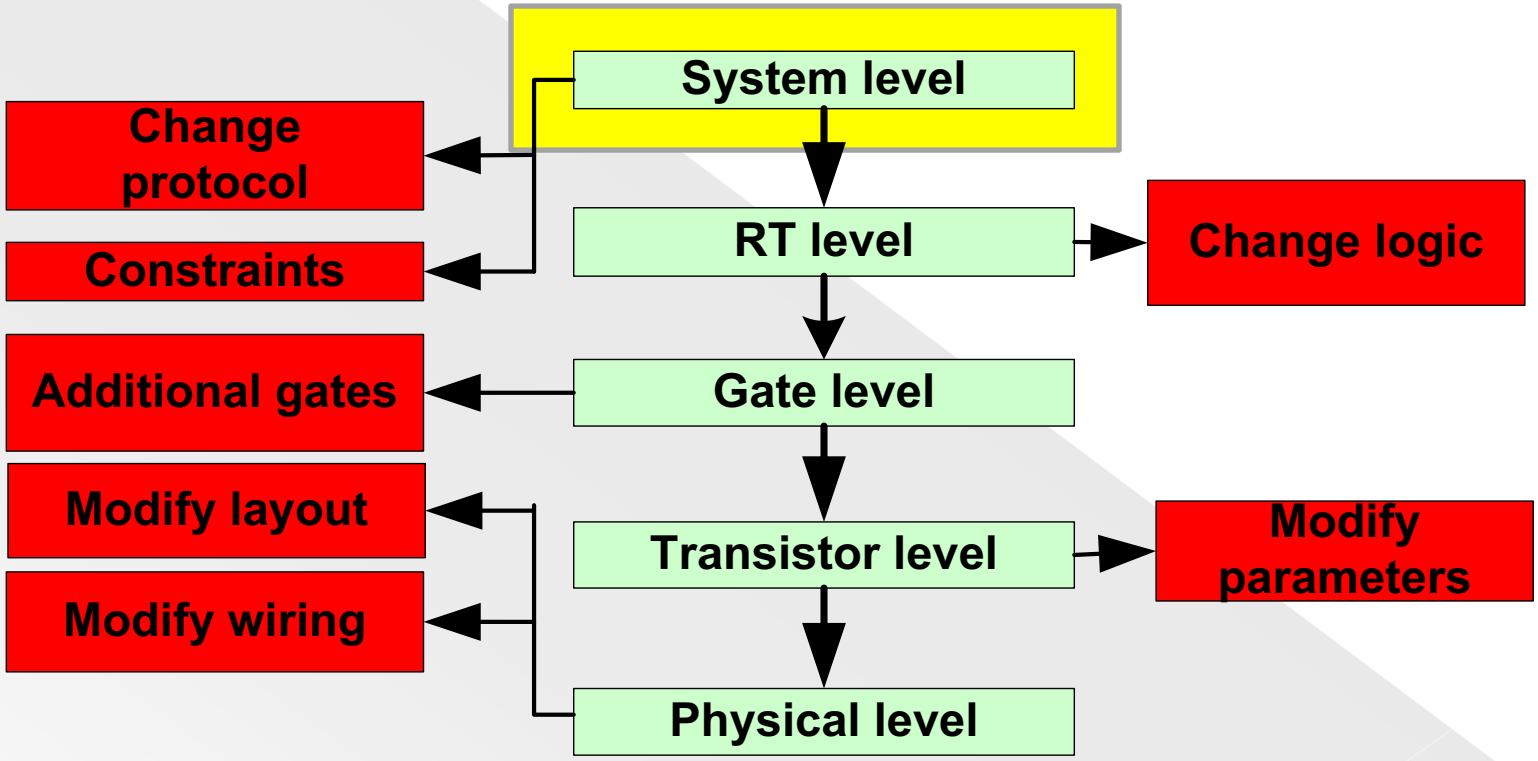
Hardware abstraction level



Development environment: Software tools are used at all stages.

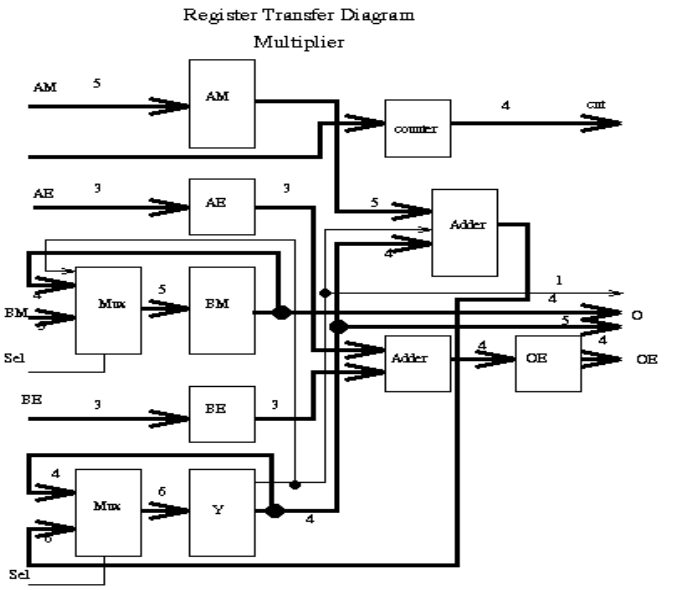
Hardware abstraction level

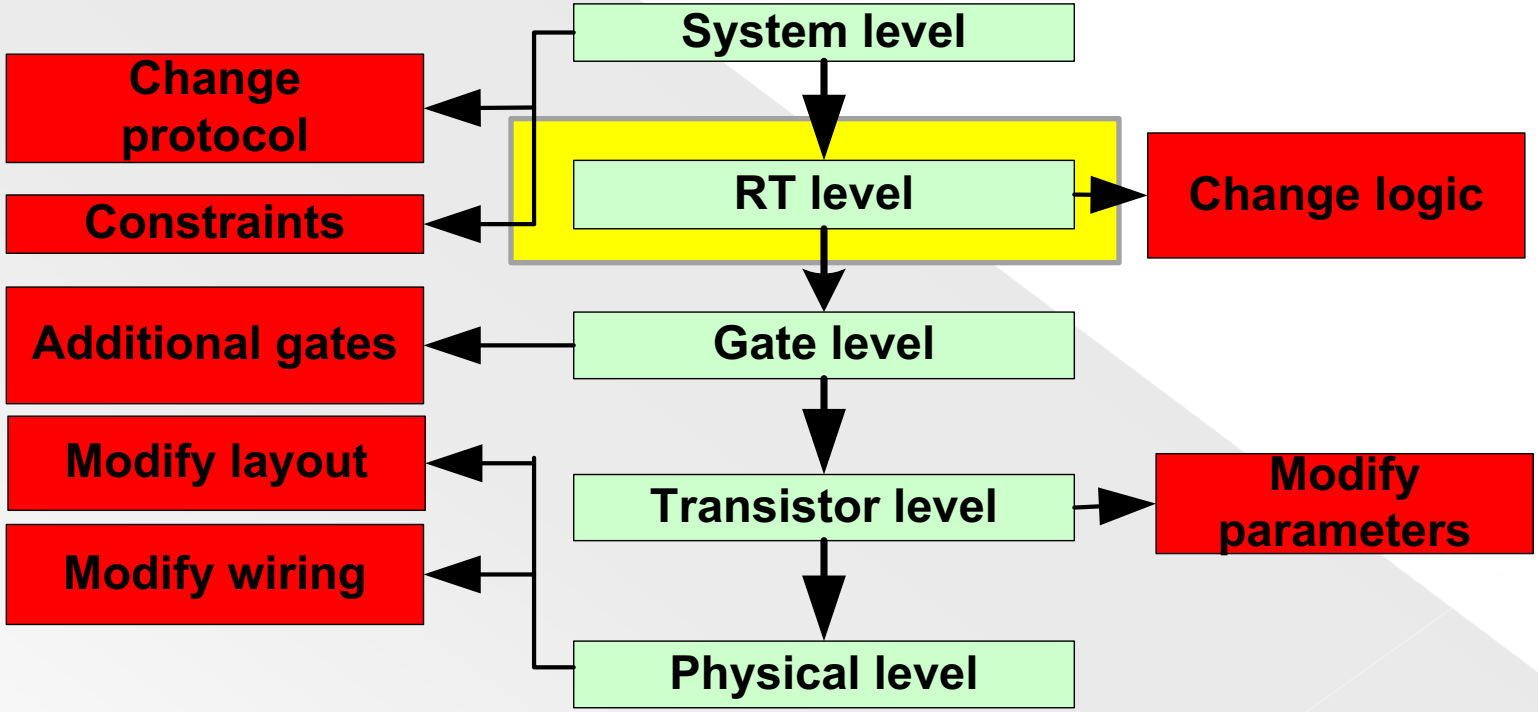




Development environment: Software tools are used at all stages.

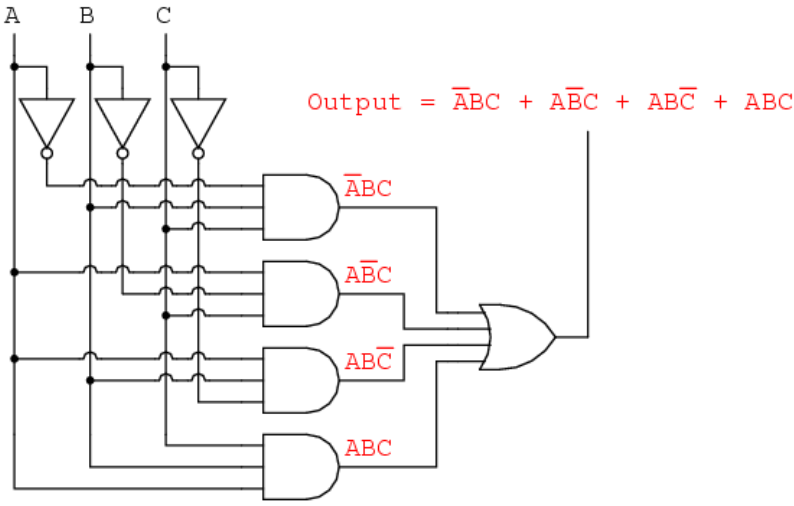
Hardware abstraction level

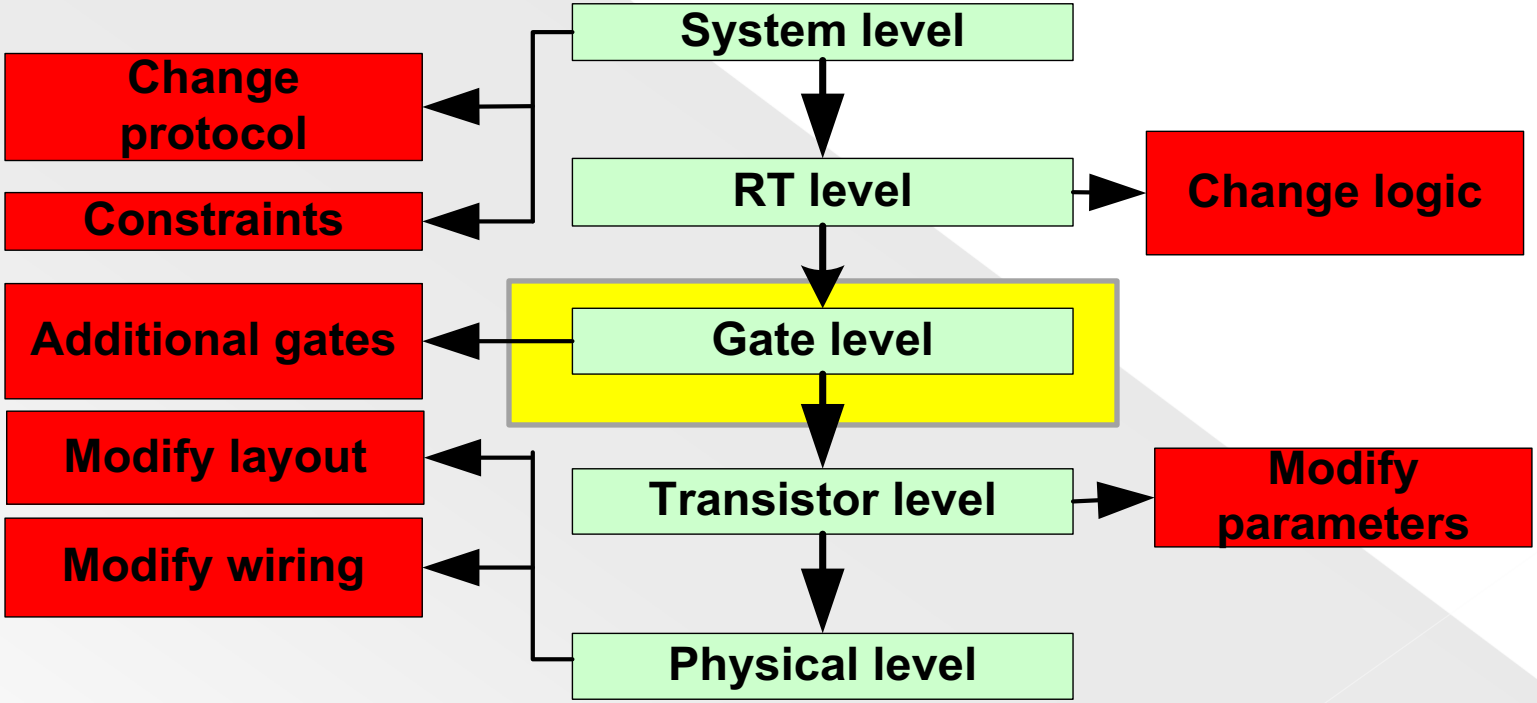




Development environment: Software tools are used at all stages.

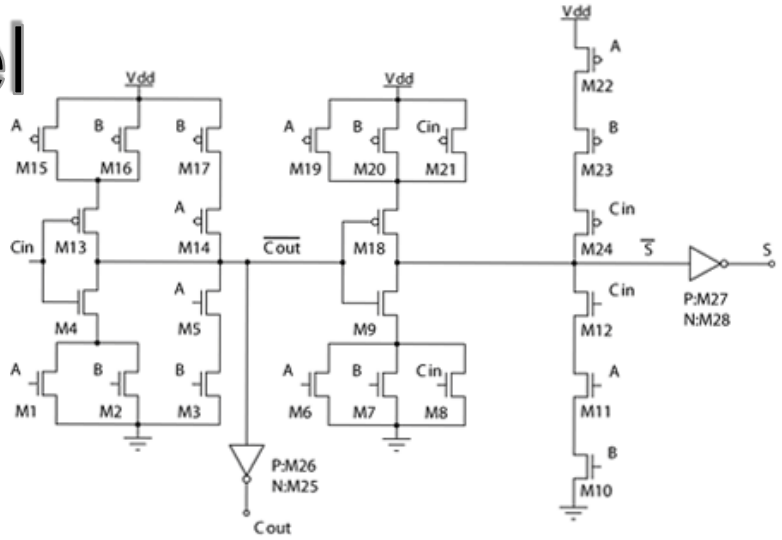
Hardware abstraction level

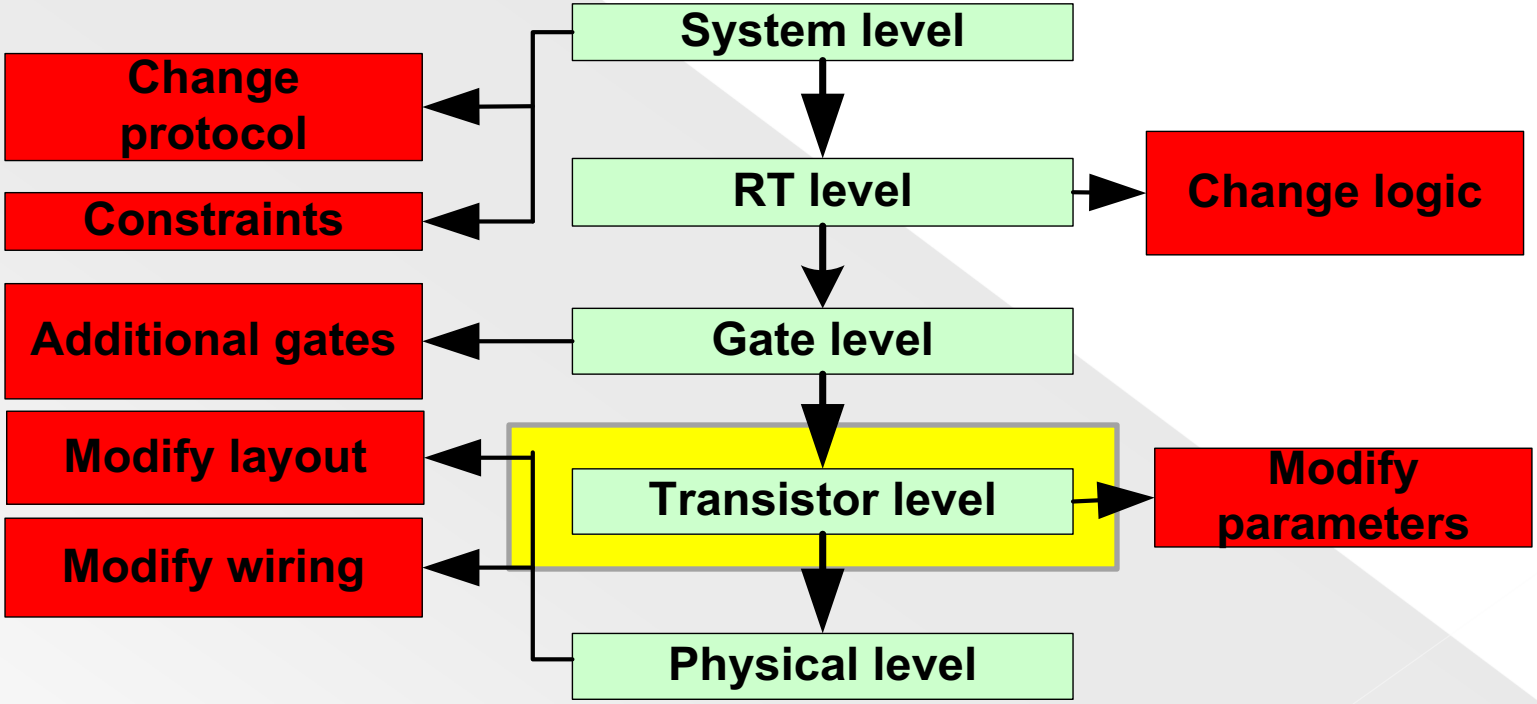




Development environment: Software tools are used at all stages.

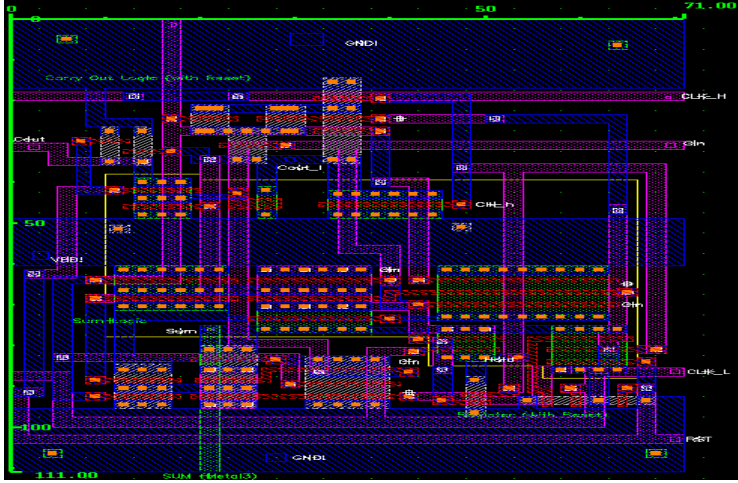
Hardware abstraction level

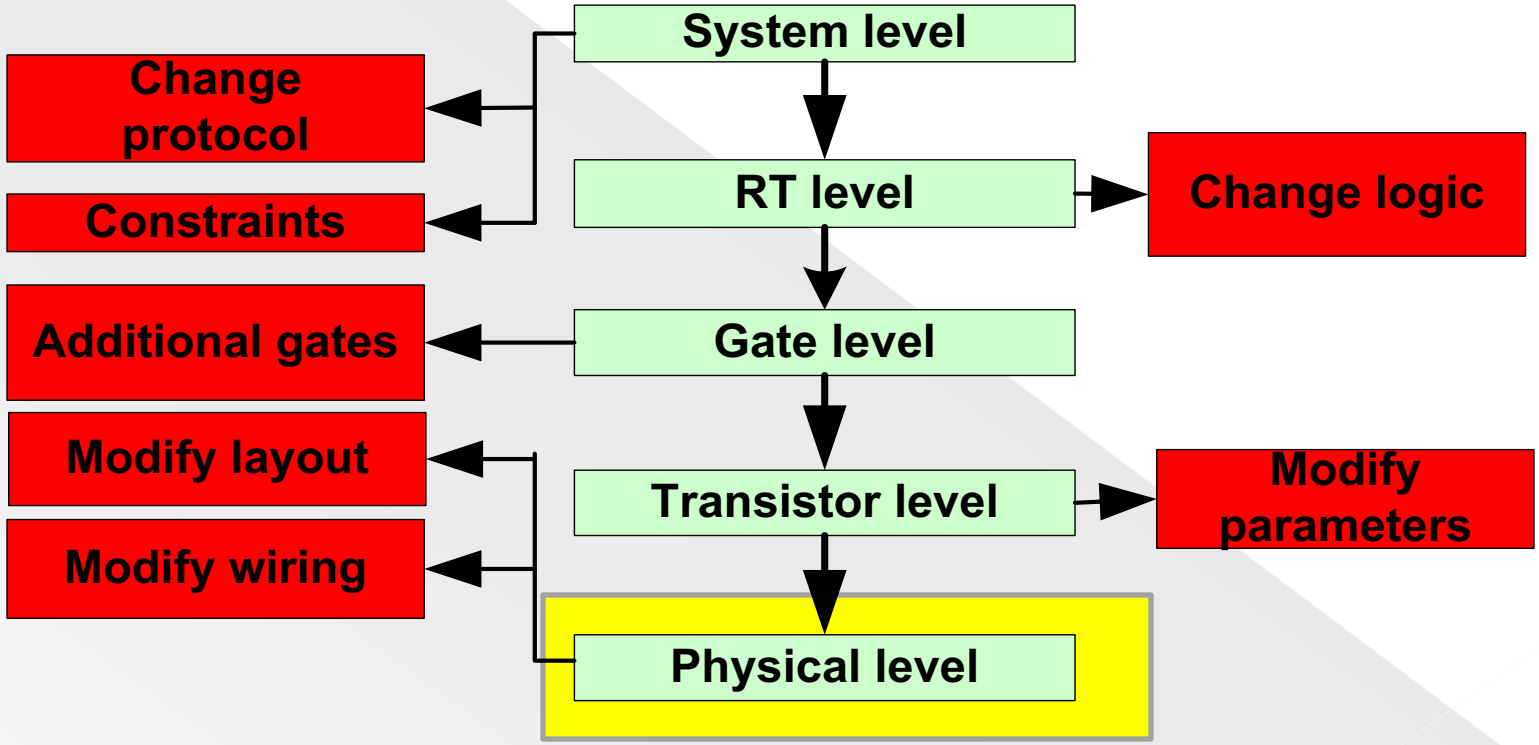




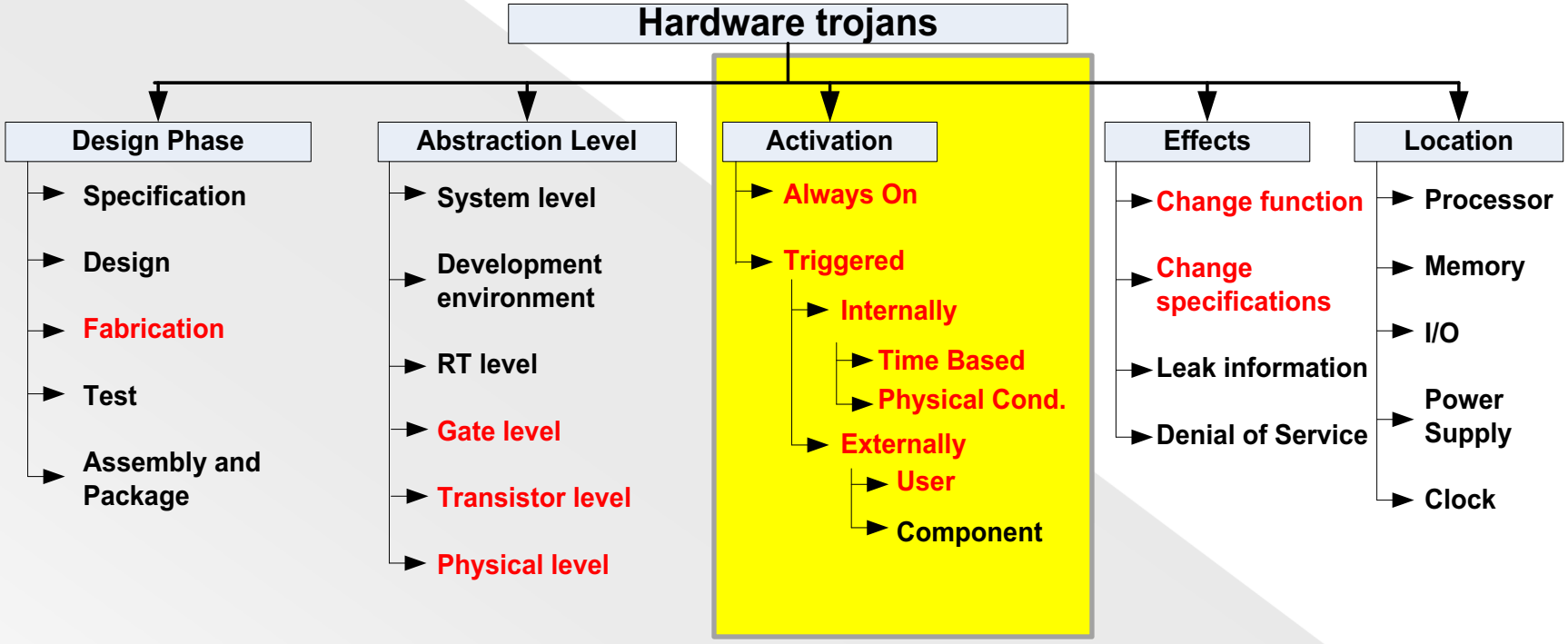
Development environment: Software tools are used at all stages.

Hardware abstraction level

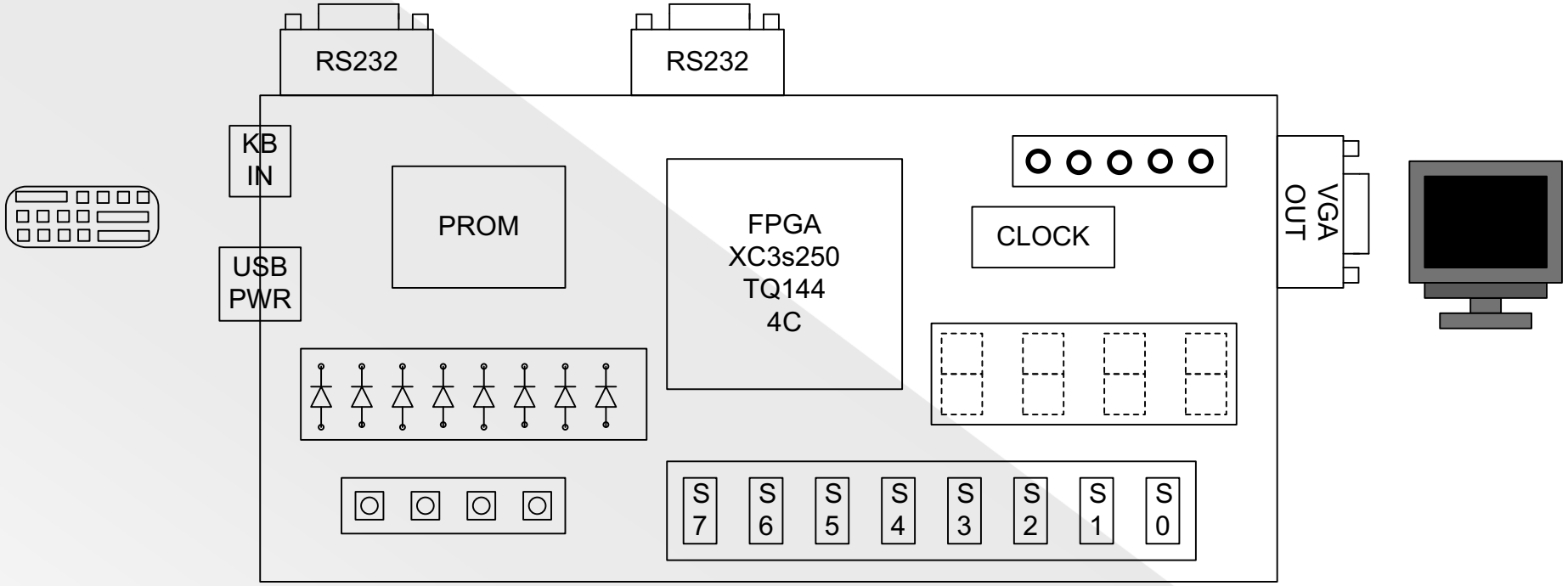


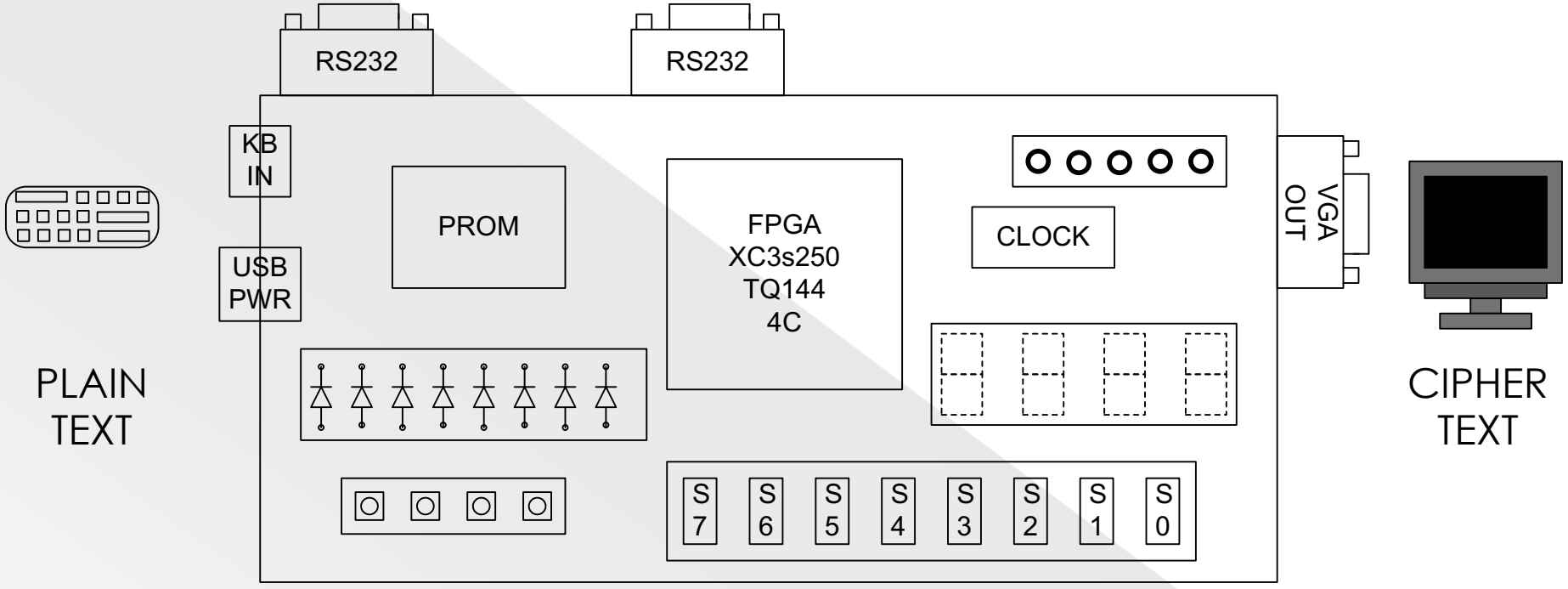


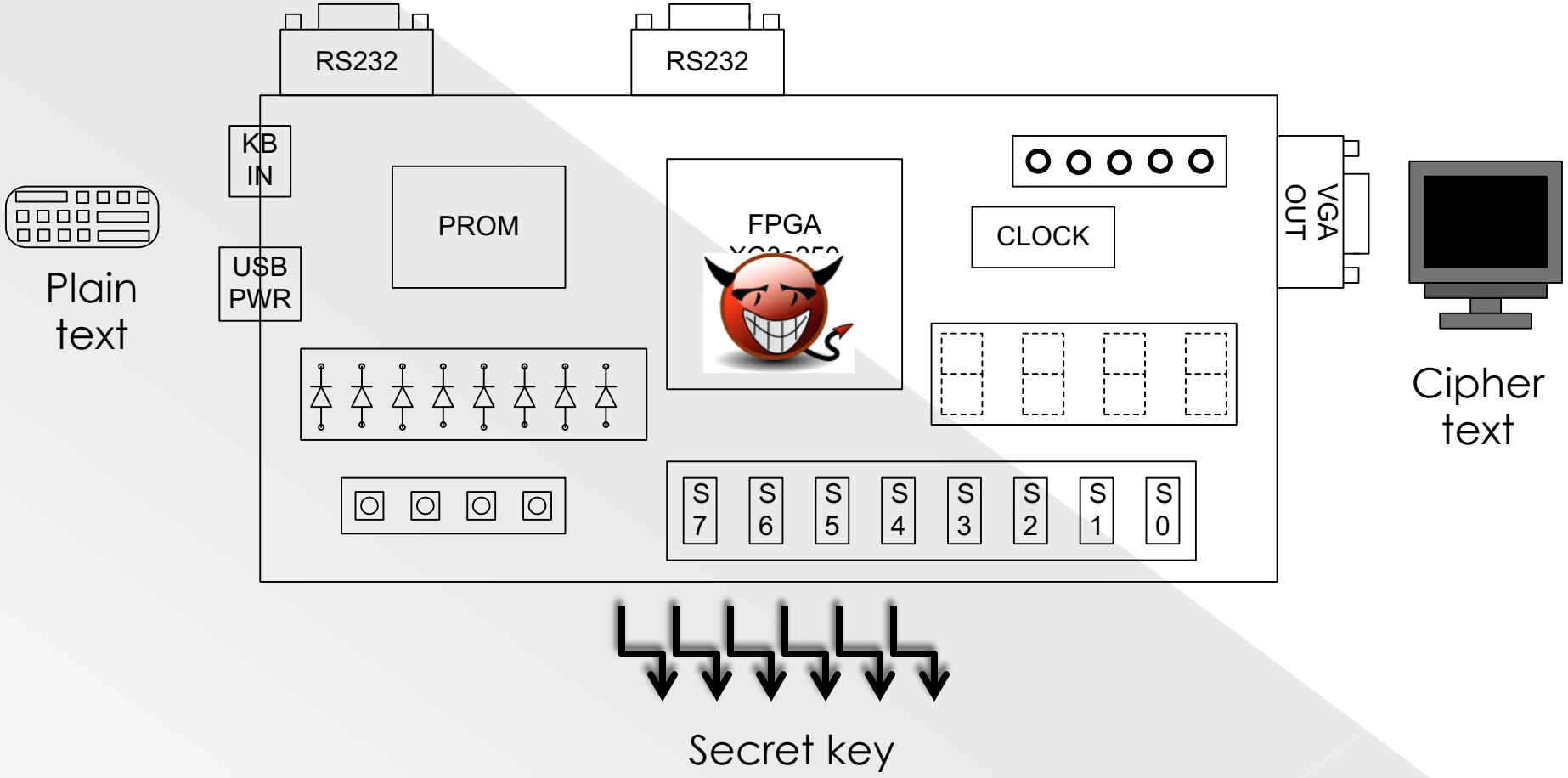
Activation


Hardware trojans
Design Phase
Specification
Design
Fabrication
Test
Assembly and Package
Abstraction Level
System level
Development environment
RT level
Gate level
Transistor level
Physical level
Activation
Always On
Triggered
Internally
Time Based
Physical Cond.
Externally
User
Component
Effects
Change function
Change specifications
Leak information
Denial of Service
Location
Processor
Memory
I/O
Power Supply
Clock

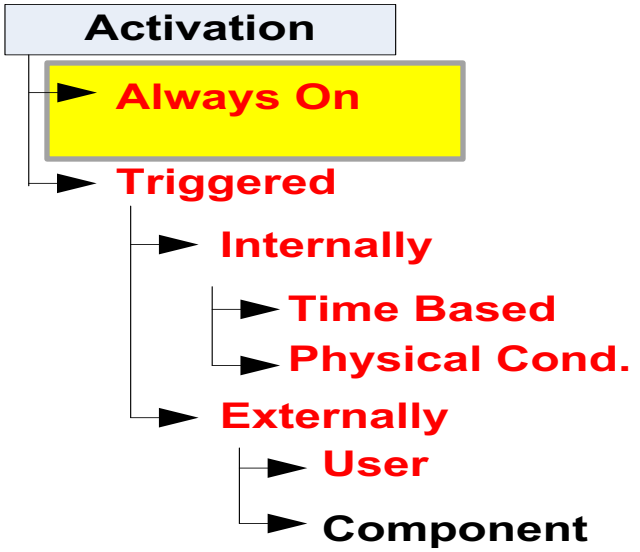
Example platform: FPGA

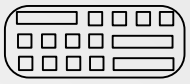



RS232
RS232
KB
IN
USB
PWR
PROM
FPGA
XC3s250
TQ144
4C
CLOCK
VGA
OUT
S
7
S
6
S
5
S
4
S
3
S
2
S
1
S
0
PLAIN
TEXT
CIPHER
TEXT

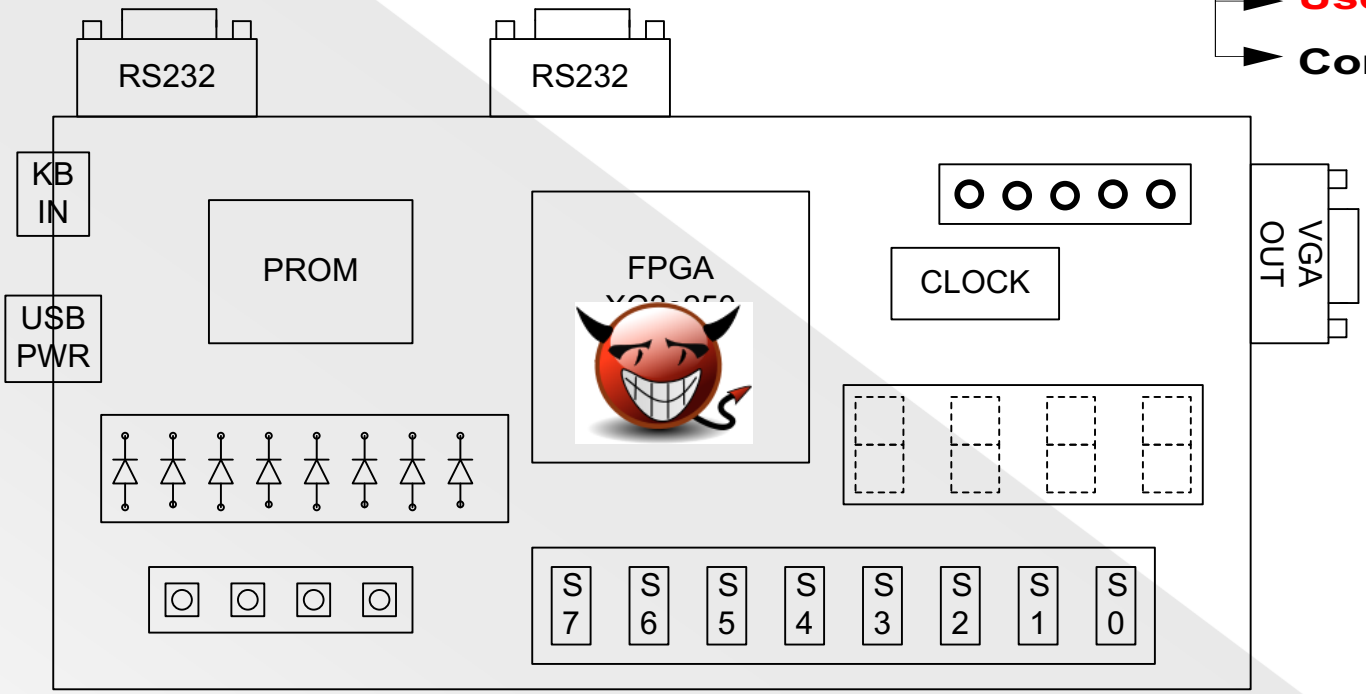

RS232
RS232
KB
IN
USB
PWR
PROM
FPGA
CLOCK
VGA
OUT
S
7
S
6
S
5
S
4
S
3
S
2
S
1
S
0
Plain
text
Cipher
text
Secret key

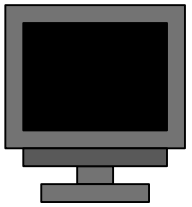
Activation





Plain text

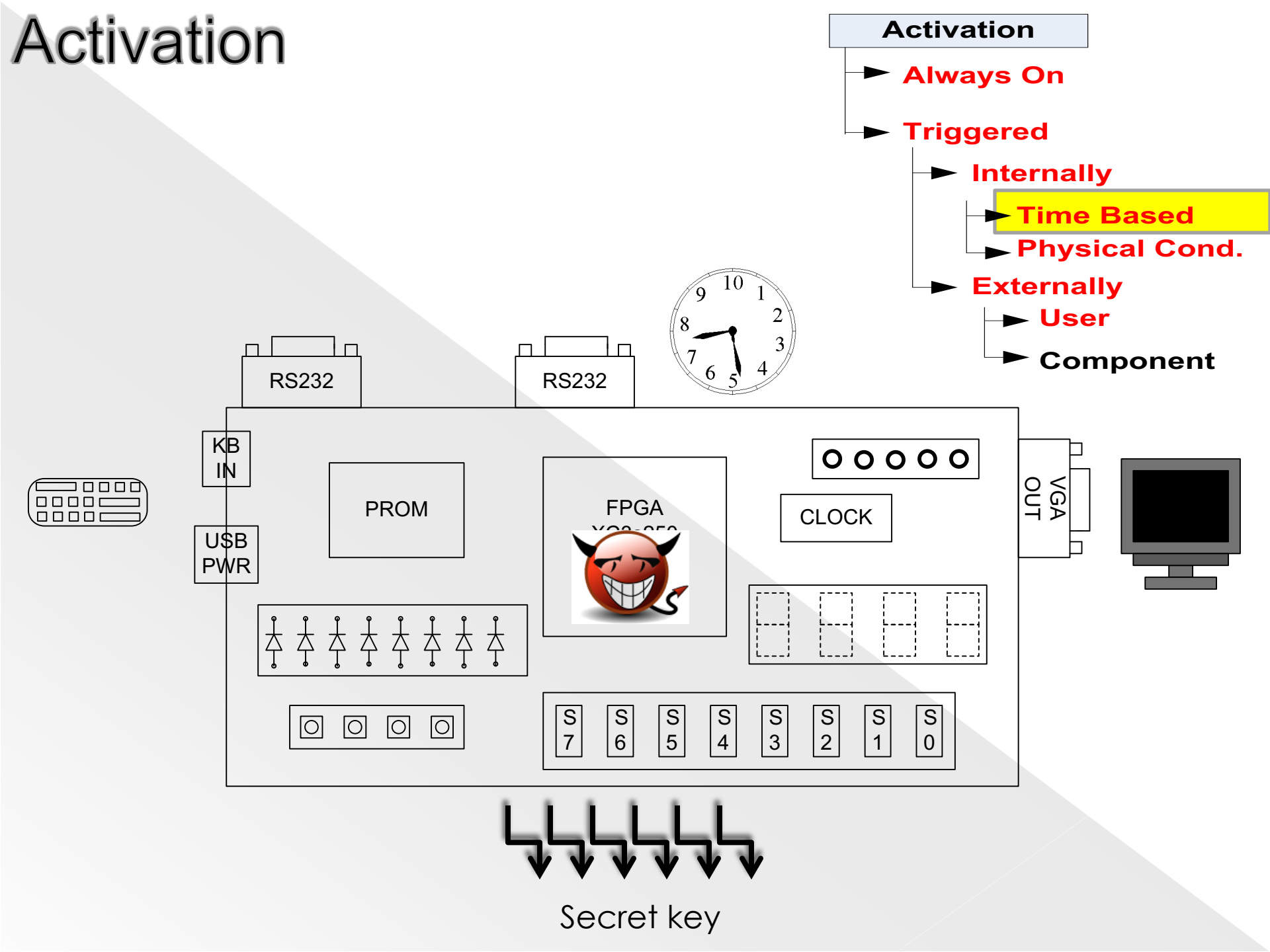




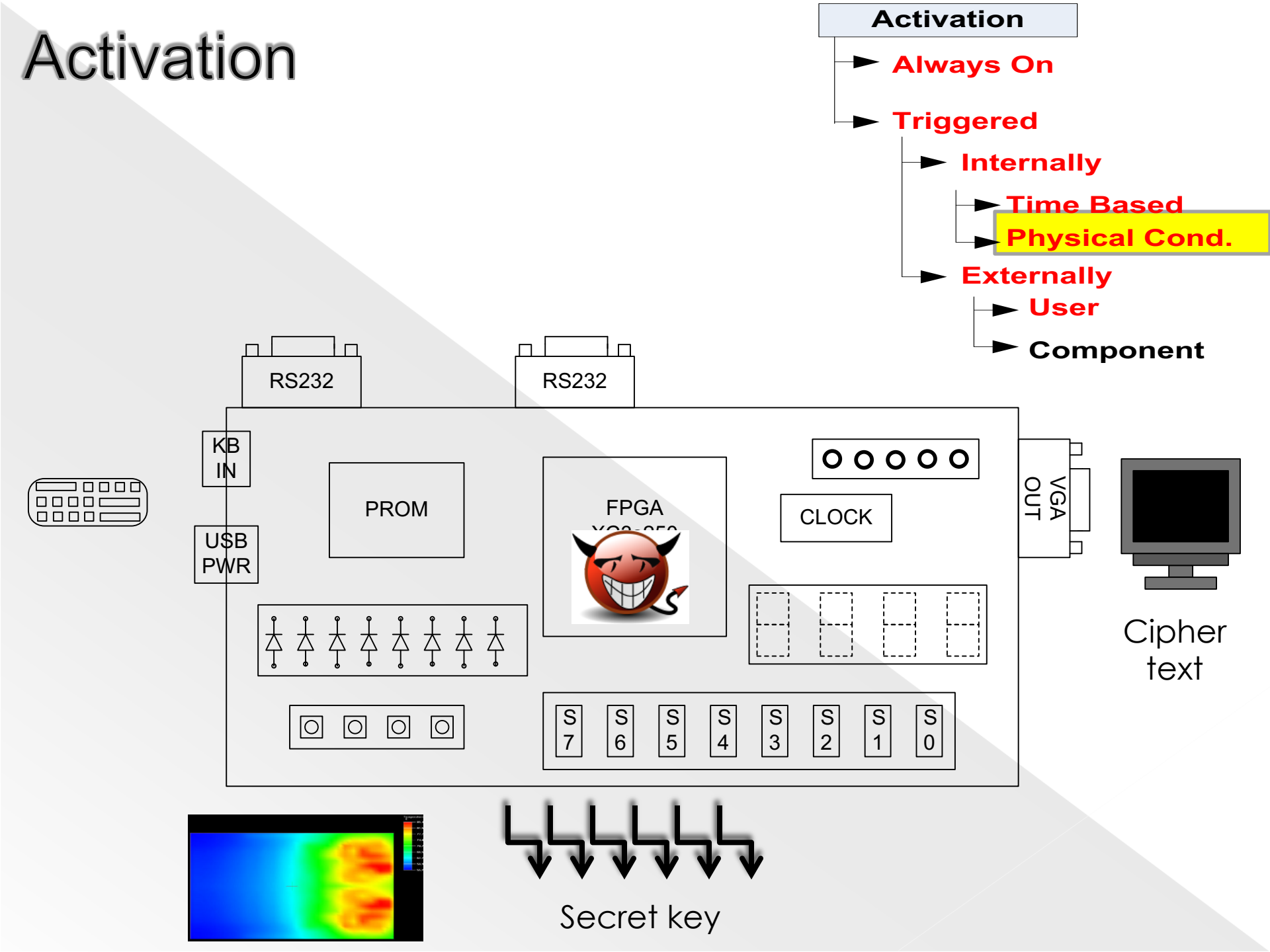
Cipher text

Secret key

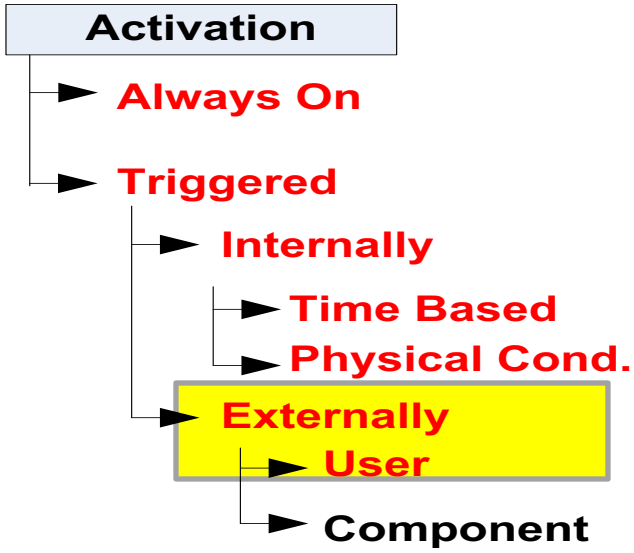
Activation

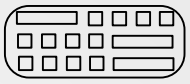

Activation
Always On
Triggered
Internally
Time Based
Physical Cond.
Externally
User
Component
RS232
RS232
KB
IN
USB
PWR
PROM
FPGA
CLOCK
VGA
OUT
S
7
S
6
S
5
S
4
S
3
S
2
S
1
S
0
Secret key

Activation

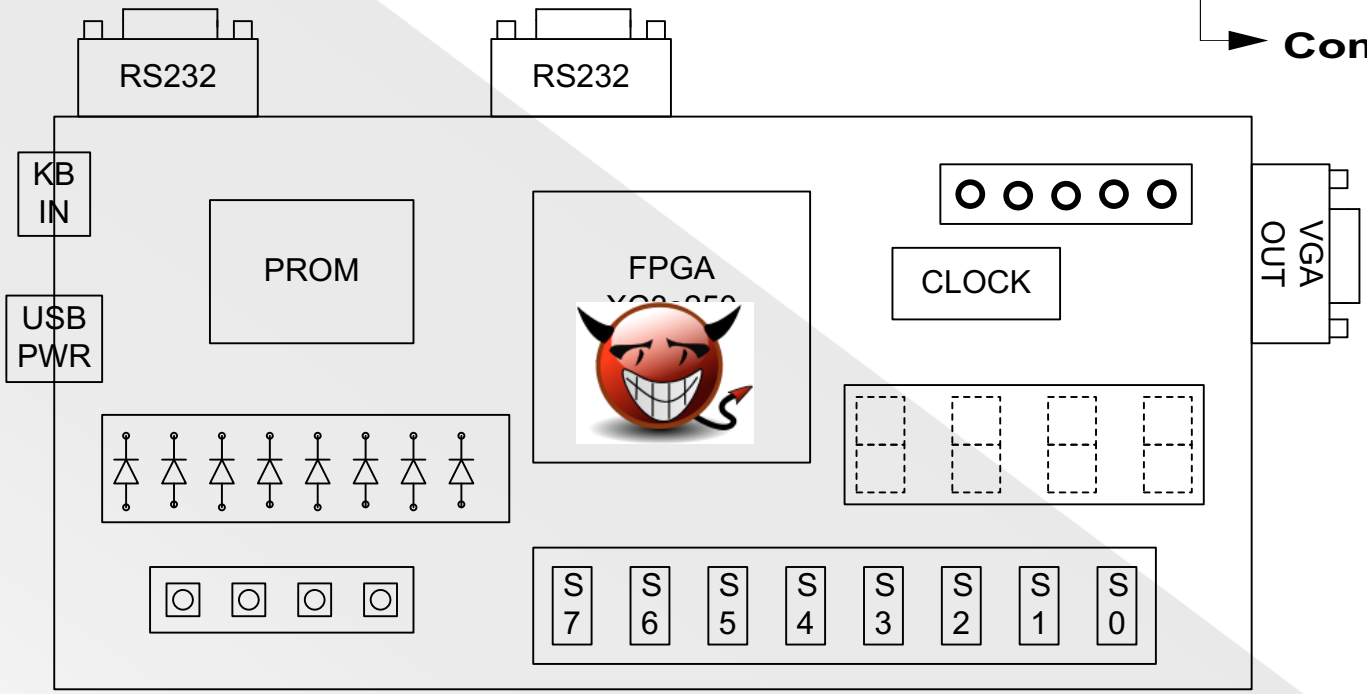

Activation
Always On
Triggered
Internally
Time Based
Physical Cond.
Externally
User
Component
RS232
RS232
KB
IN
USB
PWR
PROM
FPGA
CLOCK
VGA
OUT
S
7
S
6
S
5
S
4
S
3
S
2
S
1
S
0
Cipher
text
Secret key

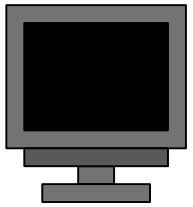
Activation





Plain text

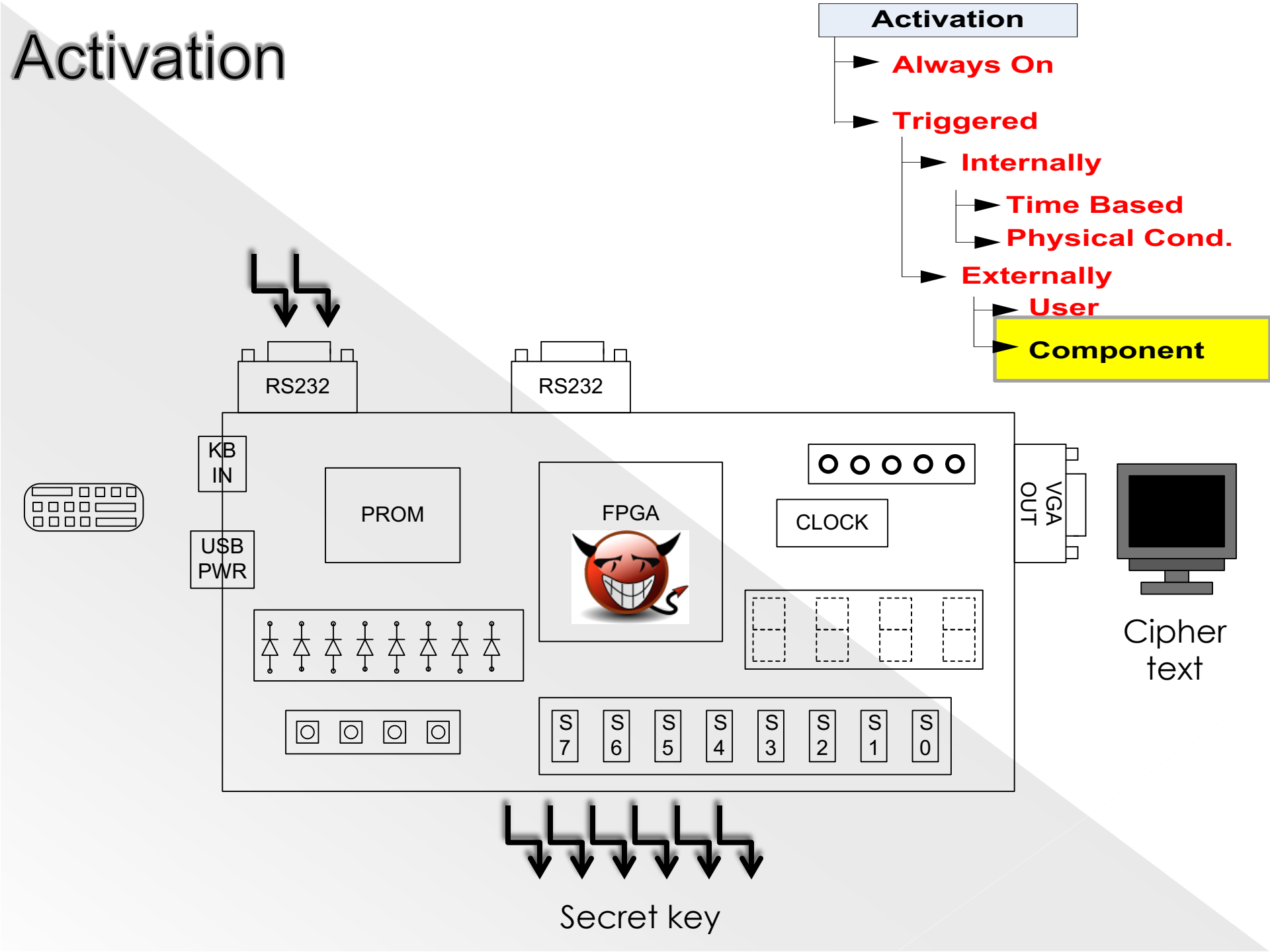




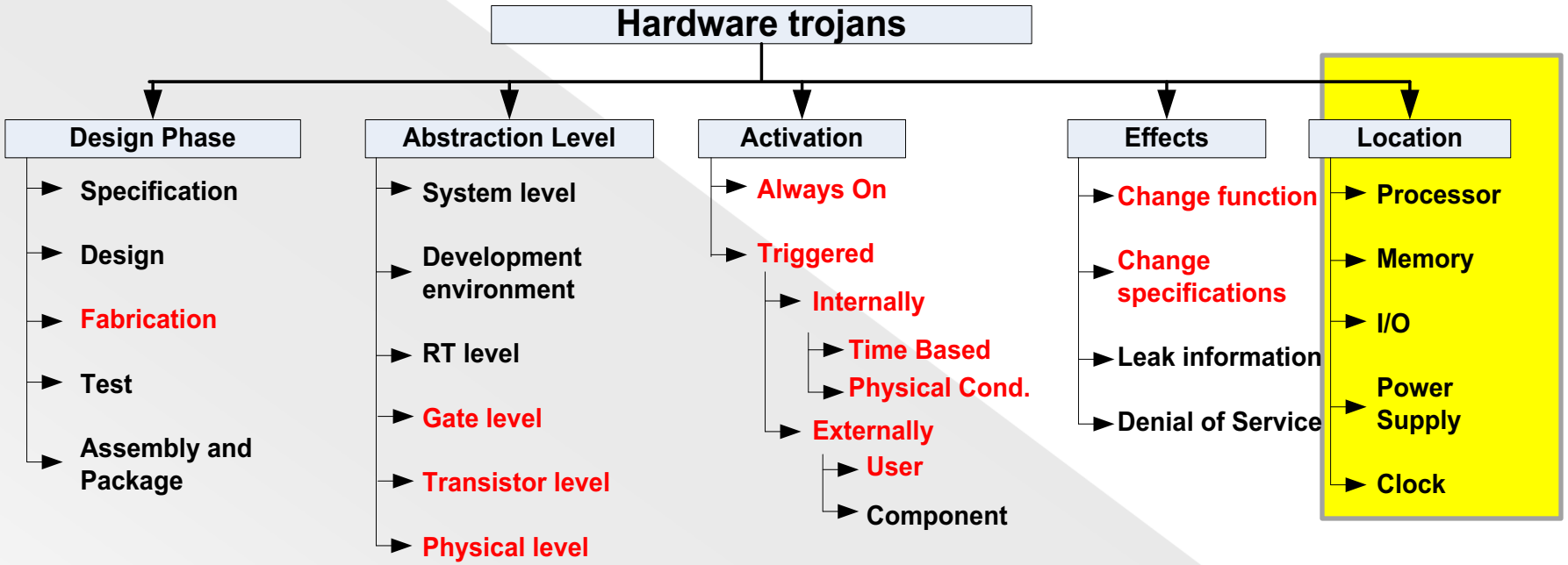
Cipher text

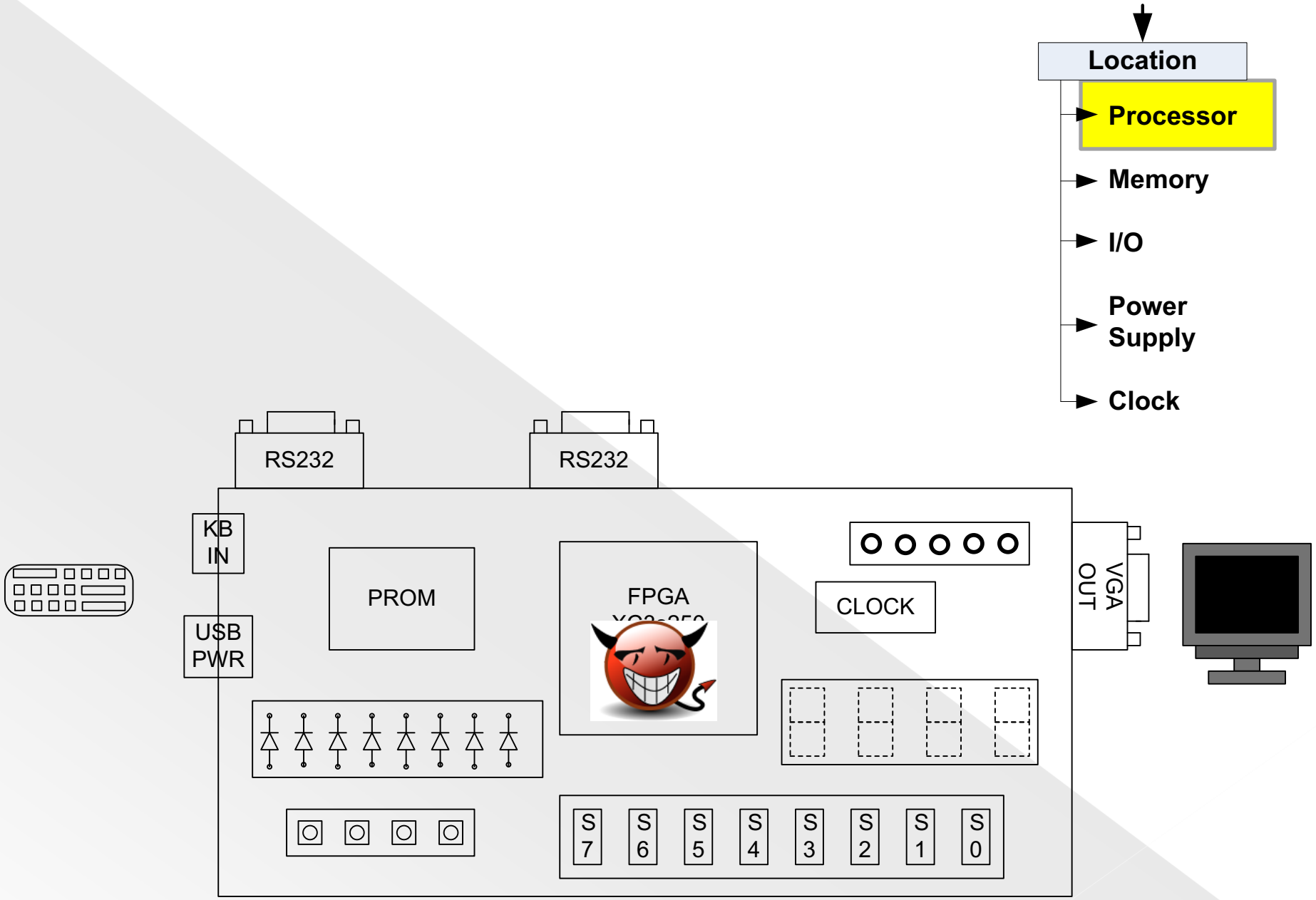
Secret key

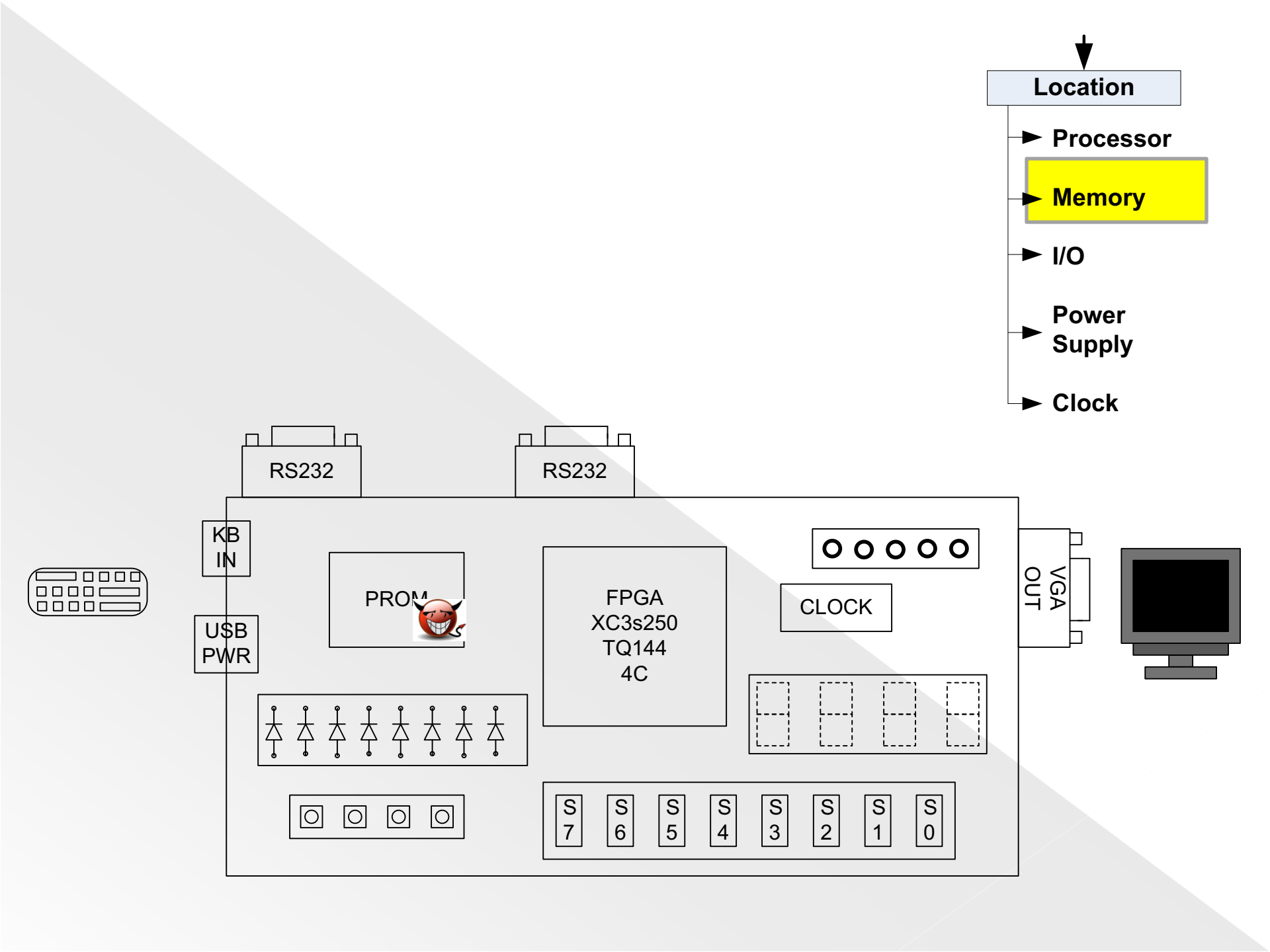
Activation

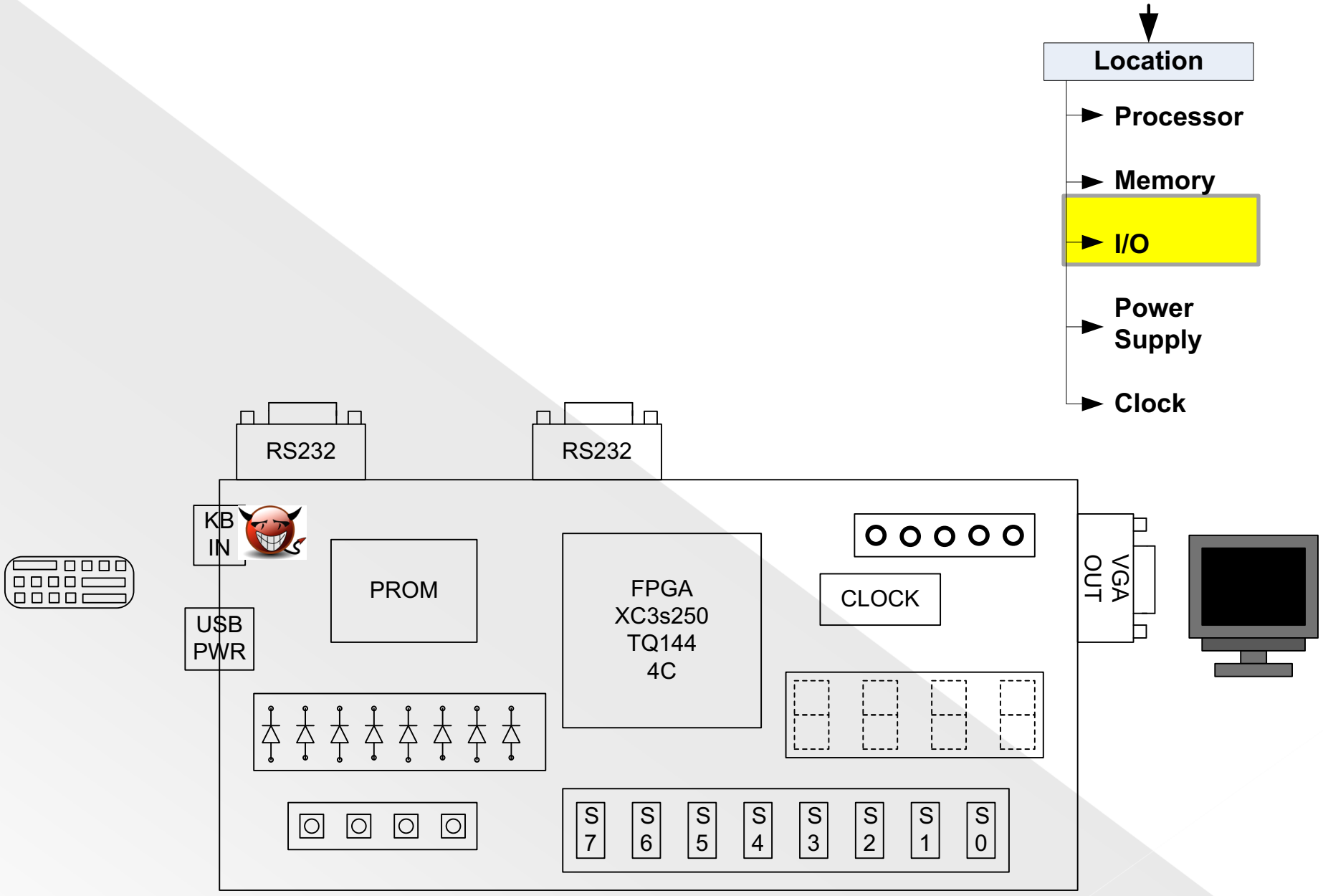

Activation
Always On
Triggered
Internally
Time Based
Physical Cond.
Externally
User
Component
RS232
RS232
KB IN
PROM
FPGA
CLOCK
VGA OUT
U$B PWR
S 7
S 6
S 5
S 4
S 3
S 2
S 1
S 0
Cipher text
Secret key

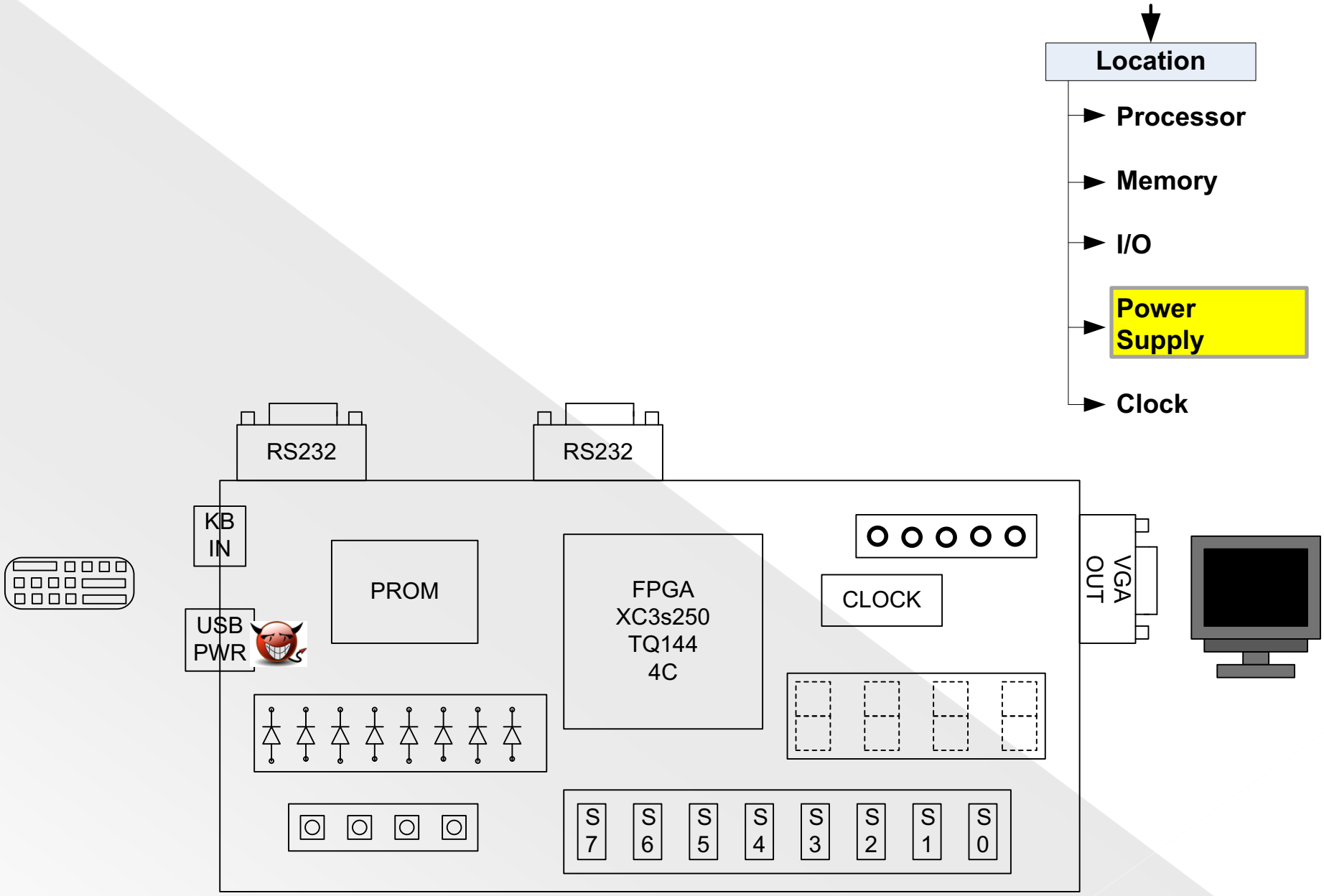
Location

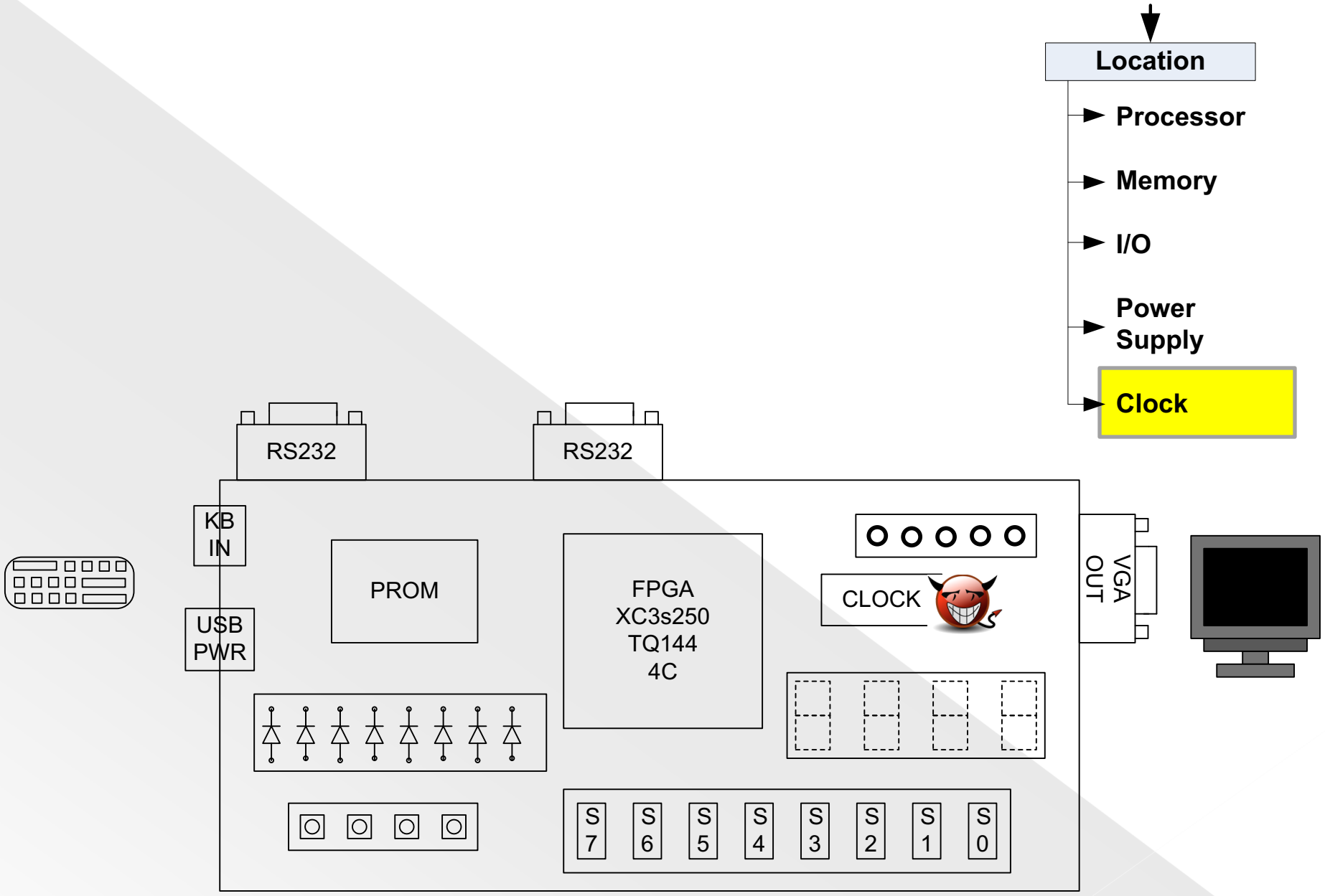

Hardware trojans
Design Phase
Specification
Design
Fabrication
Test
Assembly and Package
Abstraction Level
System level
Development environment
RT level
Gate level
Transistor level
Physical level
Activation
Always On
Triggered
Internally
Time Based
Physical Cond.
Externally
User
Component
Effects
Change function
Change specifications
Leak information
Denial of Service
Location
Processor
Memory
I/O
Power Supply
Clock


Location
Processor
Memory
I/O
Power Supply
Clock
RS232
RS232
KB IN
USB PWR
PROM
FPGA
CLOCK
VGA OUT
S 7
S 6
S 5
S 4
S 3
S 2
S 1
S 0

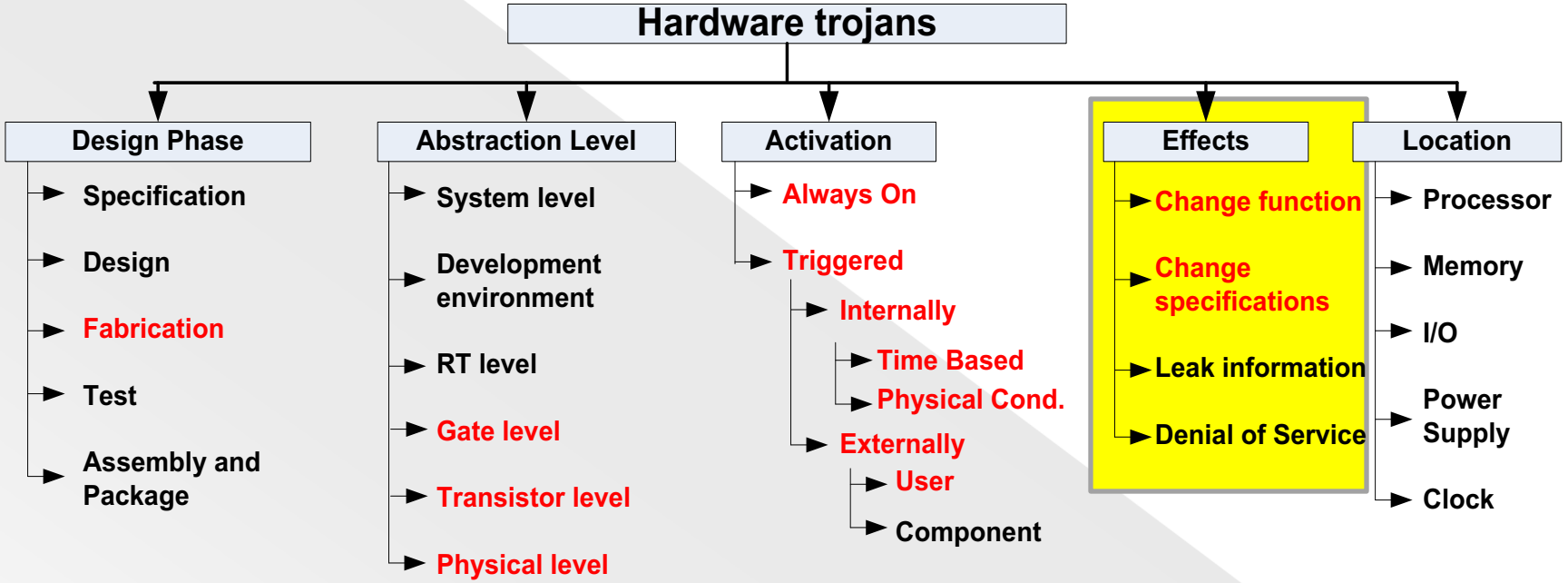

Location
Processor
Memory
I/O
Power Supply
Clock
RS232
RS232
KB IN
USB PWR
PROM
FPGA XC3s250 TQ144 4C
CLOCK
VGA OUT
S7
S6
S5
S4
S3
S2
S1
S0


Location
Processor
Memory
I/O
Power Supply
Clock
RS232
RS232
KB IN
PROM
FPGA XC3s250 TQ144 4C
CLOCK
VGA OUT
USB PWR
S7
S6
S5
S4
S3
S2
S1
S0


Location
Processor
Memory
I/O
Power Supply
Clock
RS232
RS232
KB IN
USB PWR
PROM
FPGA XC3s250 TQ144 4C
CLOCK
VGA OUT
S7
S6
S5
S4
S3
S2
S1
S0

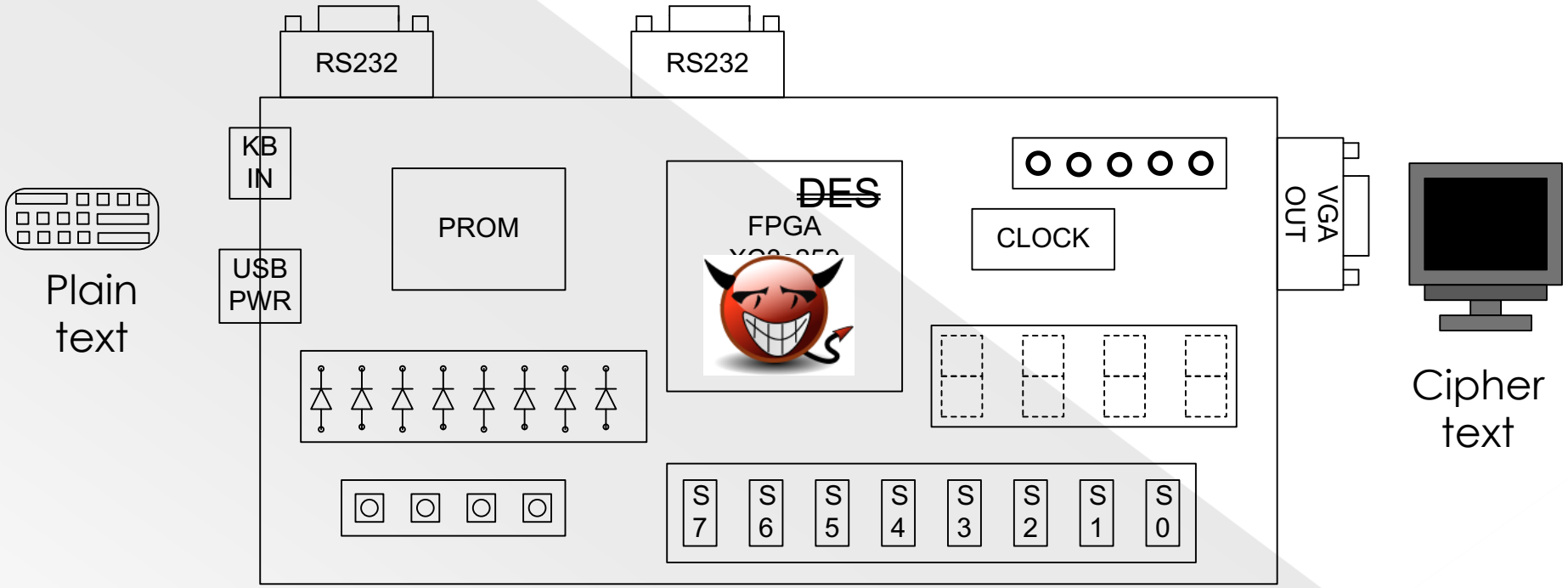

Location
Processor
Memory
I/O
Power Supply
Clock
RS232
RS232
KB IN
USB PWR
PROM
FPGA XC3s250 TQ144 4C
CLOCK
VGA OUT
S7
S6
S5
S4
S3
S2
S1
S0

Effects

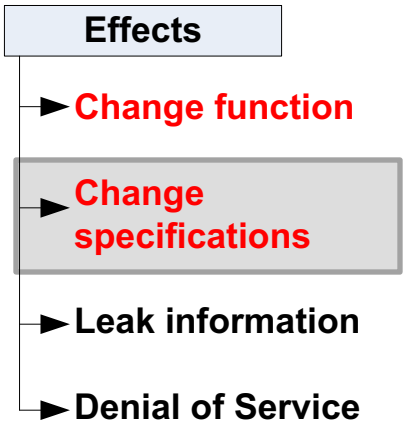

Hardware trojans
Design Phase
Specification
Design
Fabrication
Test
Assembly and Package
Abstraction Level
System level
Development environment
RT level
Gate level
Transistor level
Physical level
Activation
Always On
Triggered
Internally
Time Based
Physical Cond.
Externally
User
Component
Effects
Change function
Change specifications
Leak information
Denial of Service
Location
Processor
Memory
I/O
Power Supply
Clock

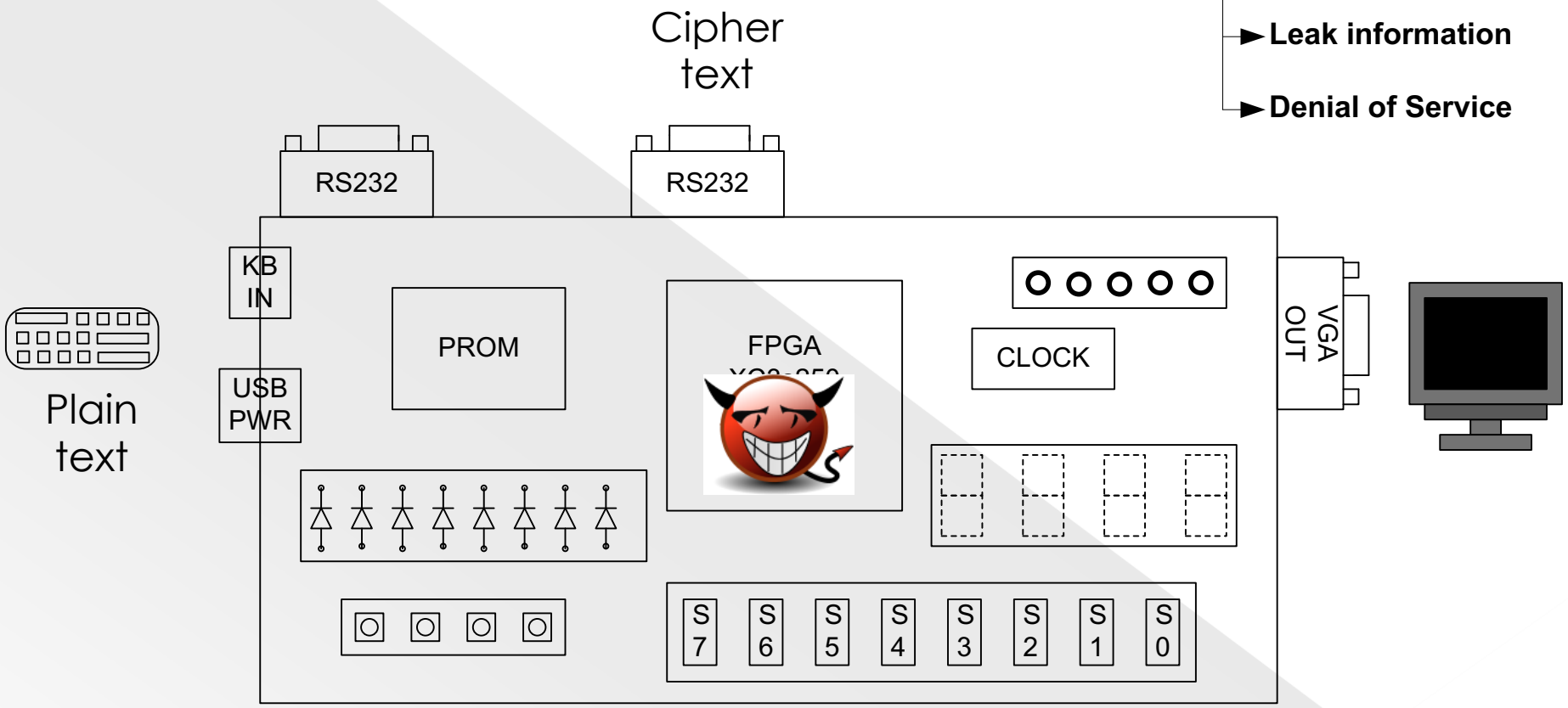
Effects

- **Effects**
  - **Change function**
  - **Change specifications**
  - **Leak information**
  - **Denial of Service**

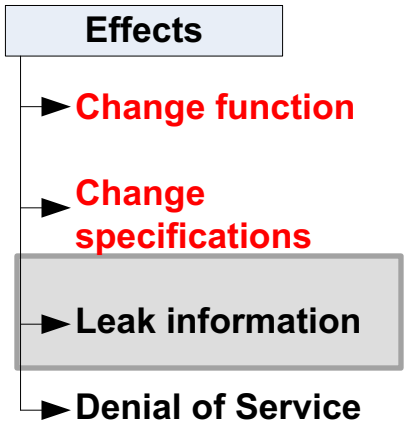


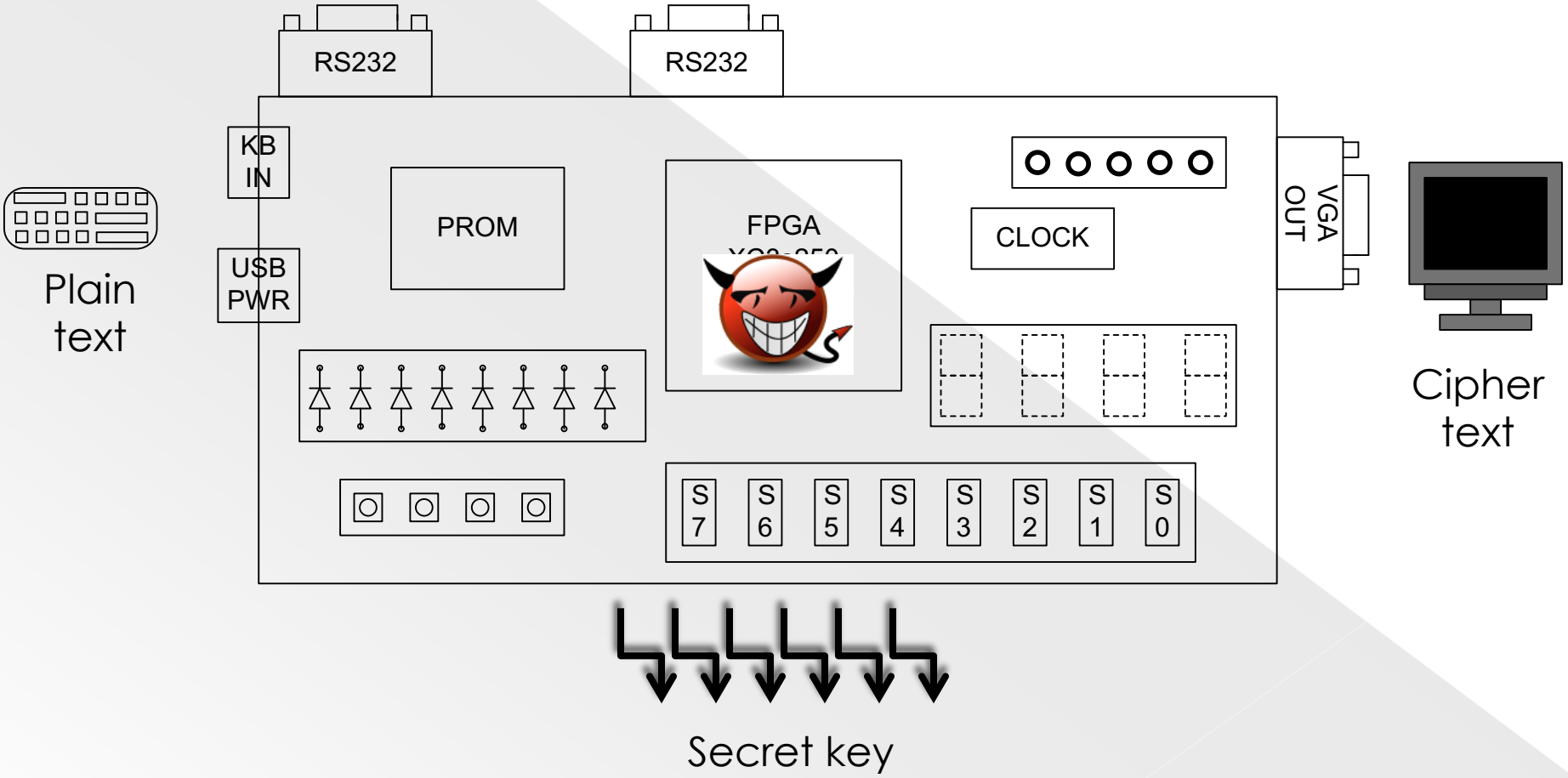
Effects


Effects
Change function
Change specifications
Leak information
Denial of Service


Cipher text
RS232
RS232
KB IN
USB PWR
PROM
FPGA
CLOCK
VGA OUT
S7
S6
S5
S4
S3
S2
S1
S0
Plain text

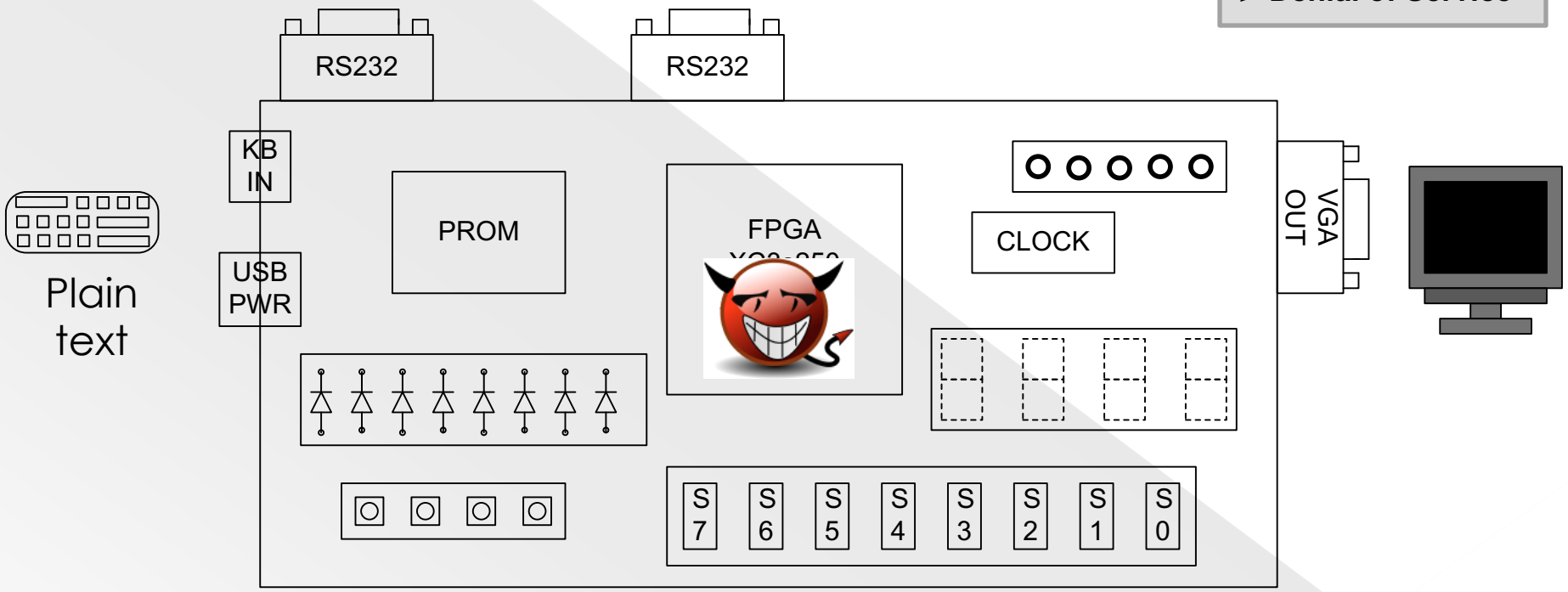
Effects


Effects
Change function
Change specifications
Leak information
Denial of Service


RS232
RS232
KB IN
USB PWR
PROM
FPGA
CLOCK
VGA OUT
S7
S6
S5
S4
S3
S2
S1
S0
Plain text
Cipher text
Secret key

Effects

- Effects
  - Change function
  - Change specifications
  - Leak information
  - Denial of Service

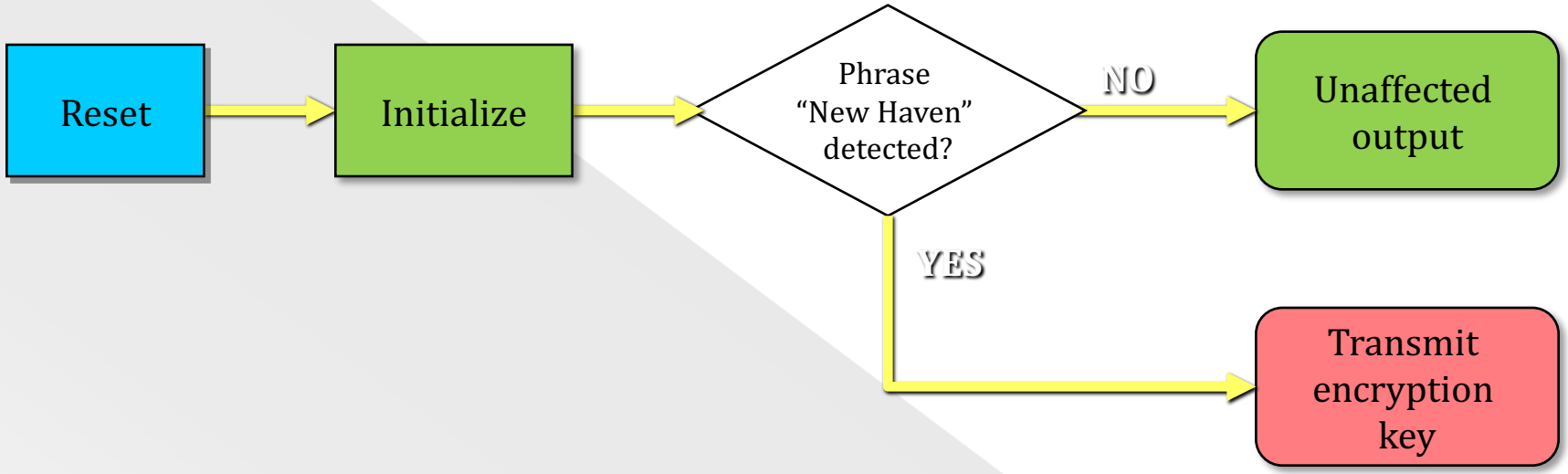


Trojan examples

- Trojan's submitted to CSAW
- Validating the taxonomy

Leak encryption key

Input triggered trojan

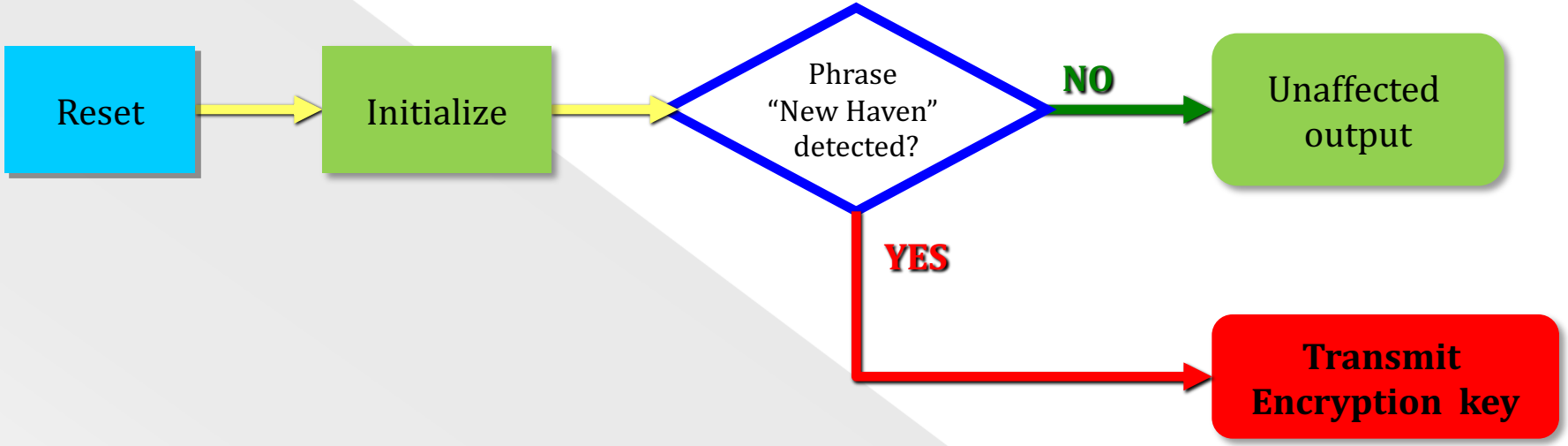


Summary

- Triggered by rare events, e.g., “New Haven”
- Physical access required for the attackers

Leak encryption key

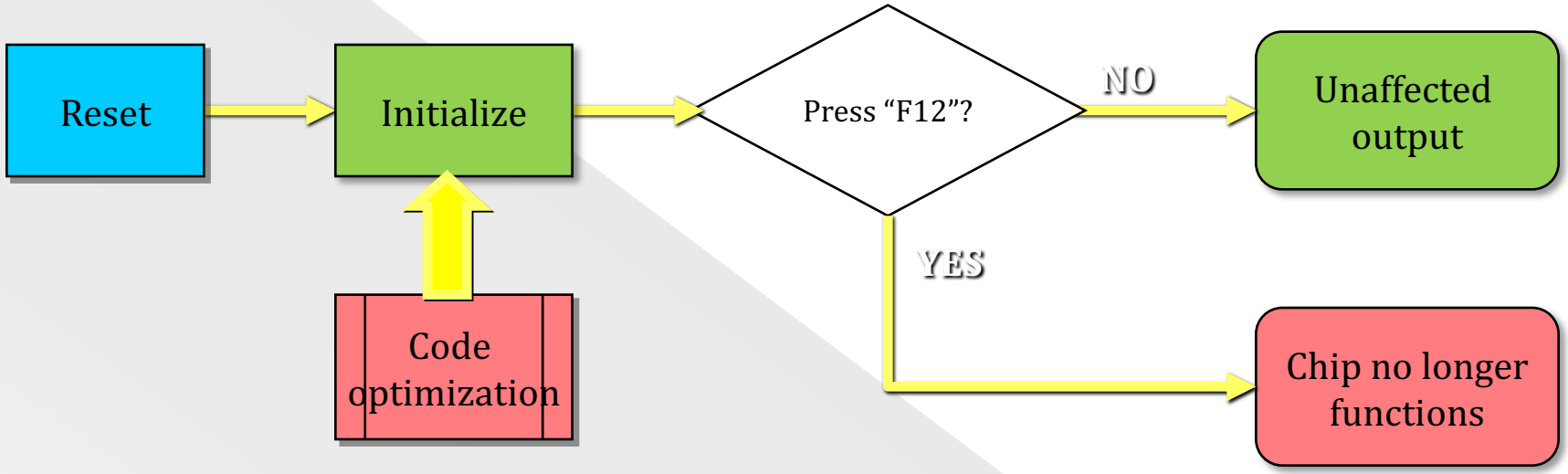
Input triggered trojan



Phase	Design
Abstraction	RT Level
Activation	User input
Effects	Leak information
Location	Processor, I/O

Denial-of-service under special input

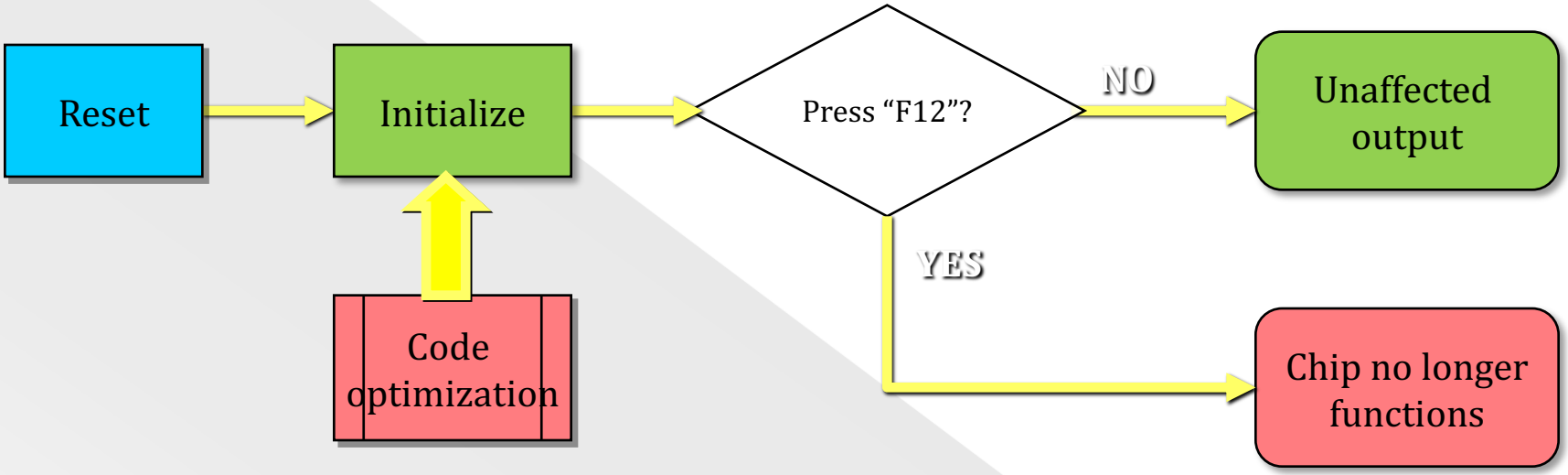
Undefined key trigger



- Triggered by undefined key on the keyboard
- 9.4% less flip-flops use
- created margin to hide Trojan

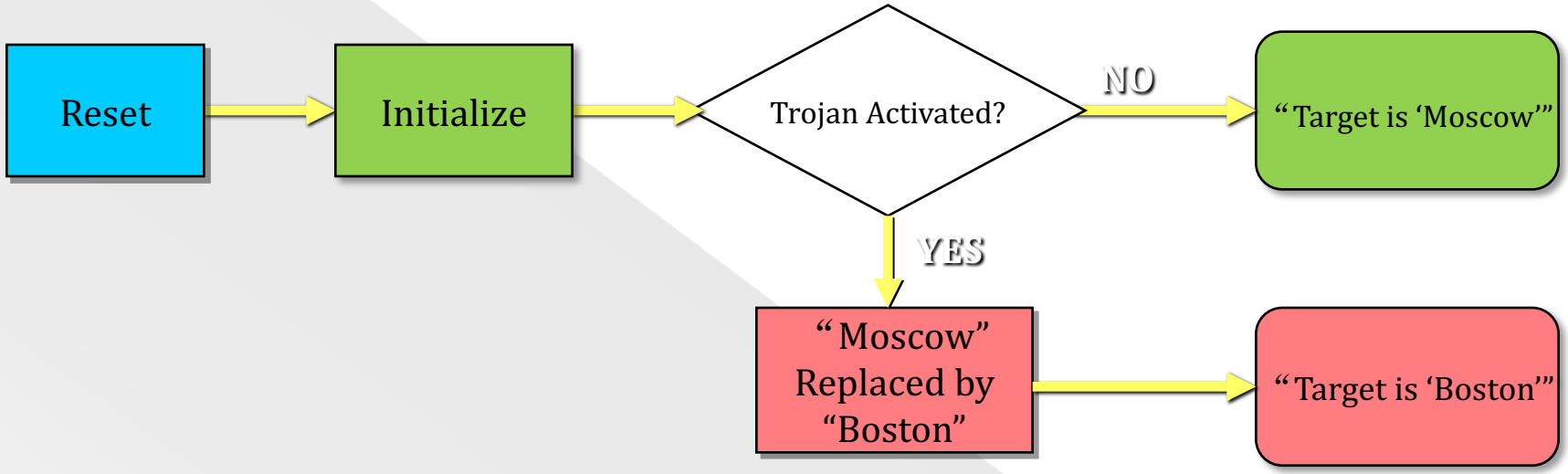
Denial-of-service under special input

Undefined key trigger



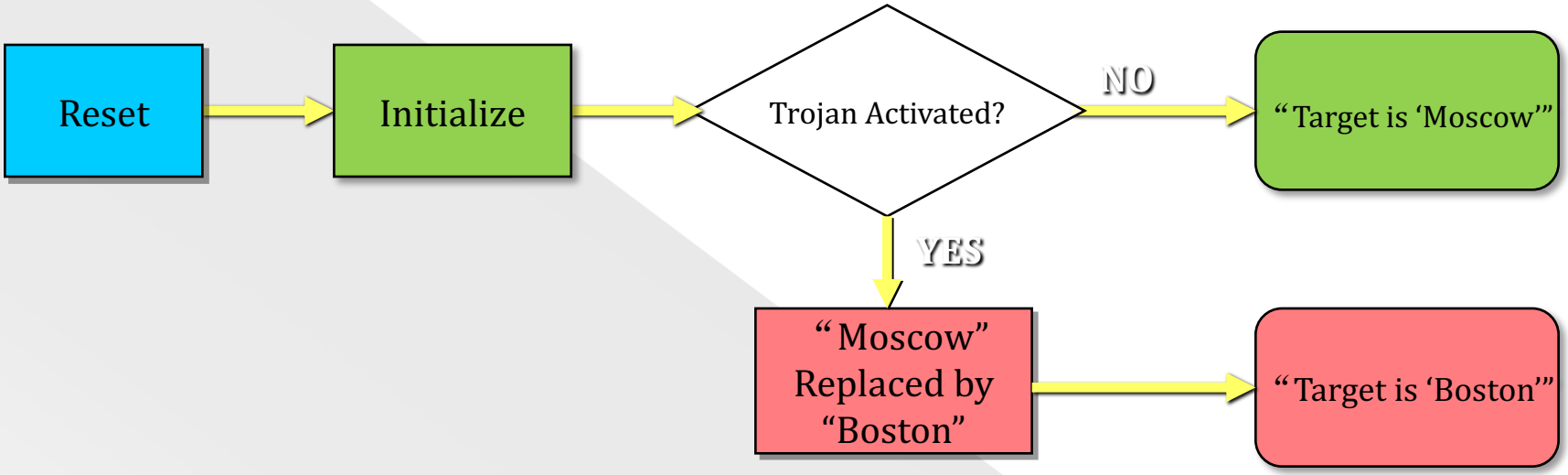
Phase	Design
Abstraction	RT Level
Activation	User input
Effects	Denial of service
Location	Processor, I/O

Faked output: Text Replacement



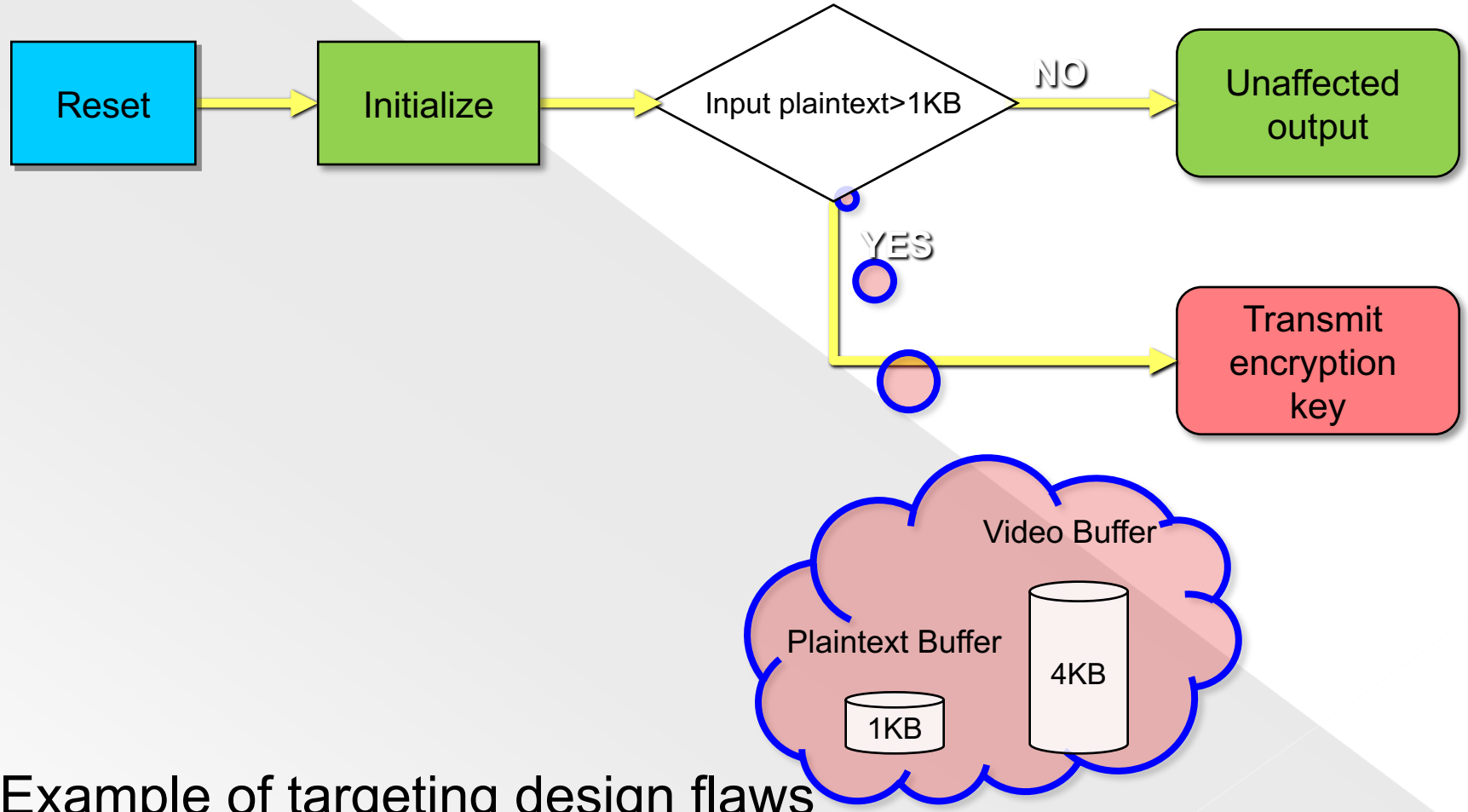
- Attackers monitor the chip communication channel
- The payload is aggressive

Faked Output



Phase	Design
Abstraction	RT level
Activation	User input
Effects	Modify information
Location	Processor, I/O

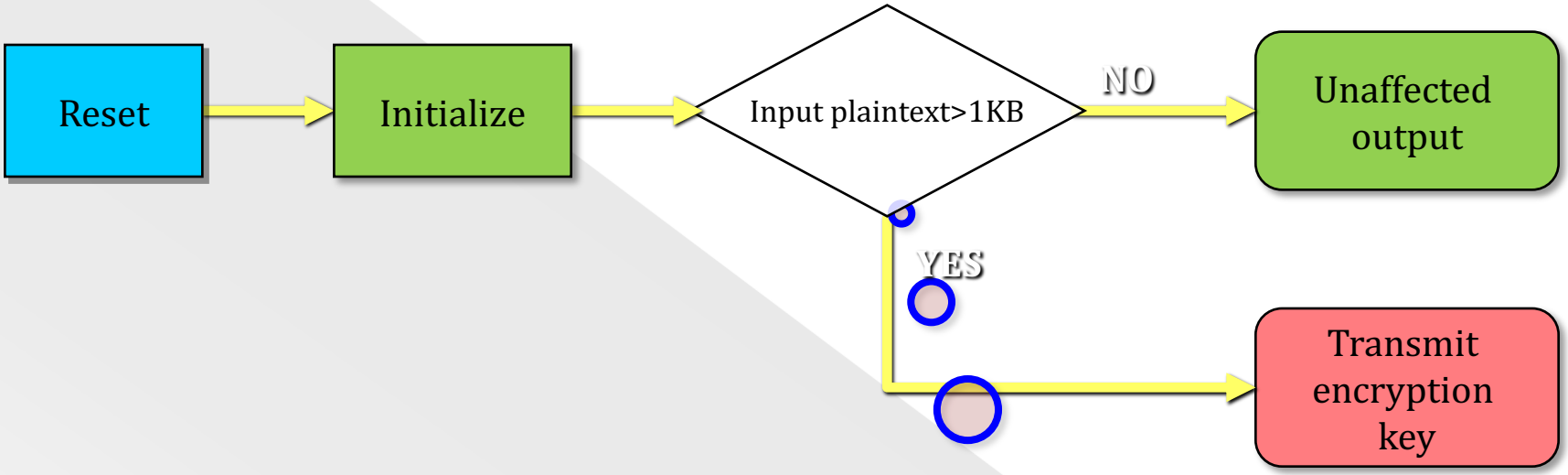
Buffer Overflow



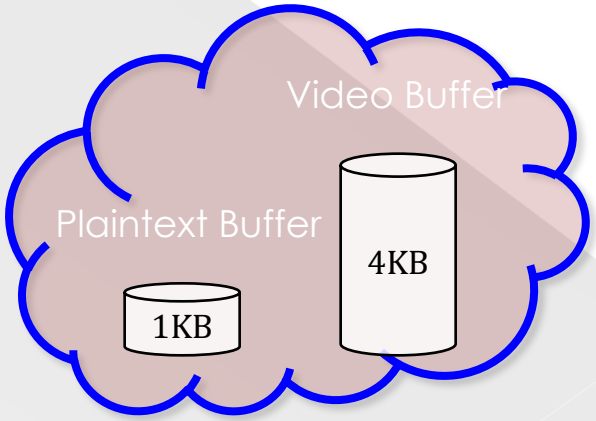
- Example of targeting design flaws
- Many design specifications do not consider potential trojans

Buffer Overflow

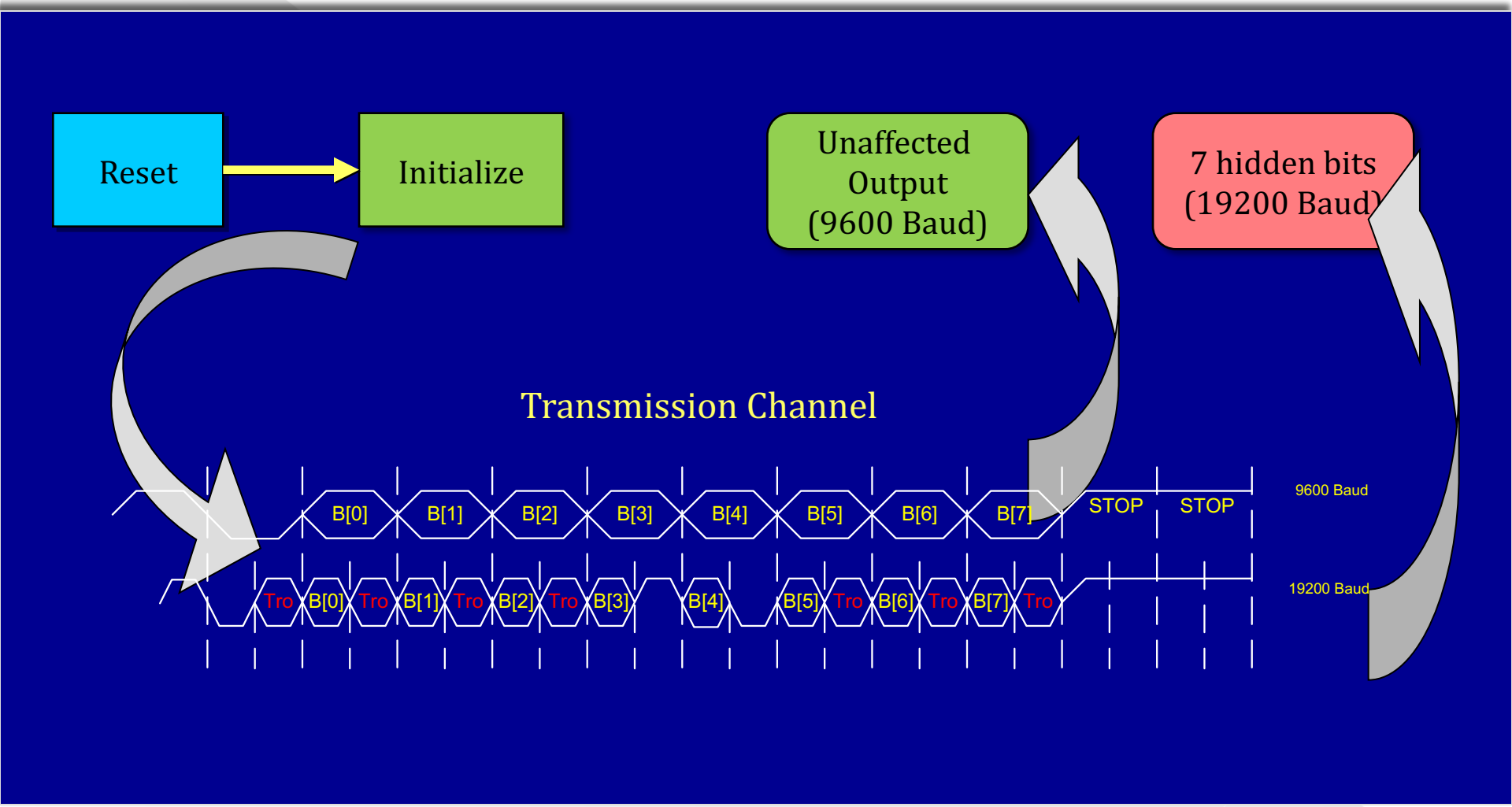
Trojan Type IV – System Flaw



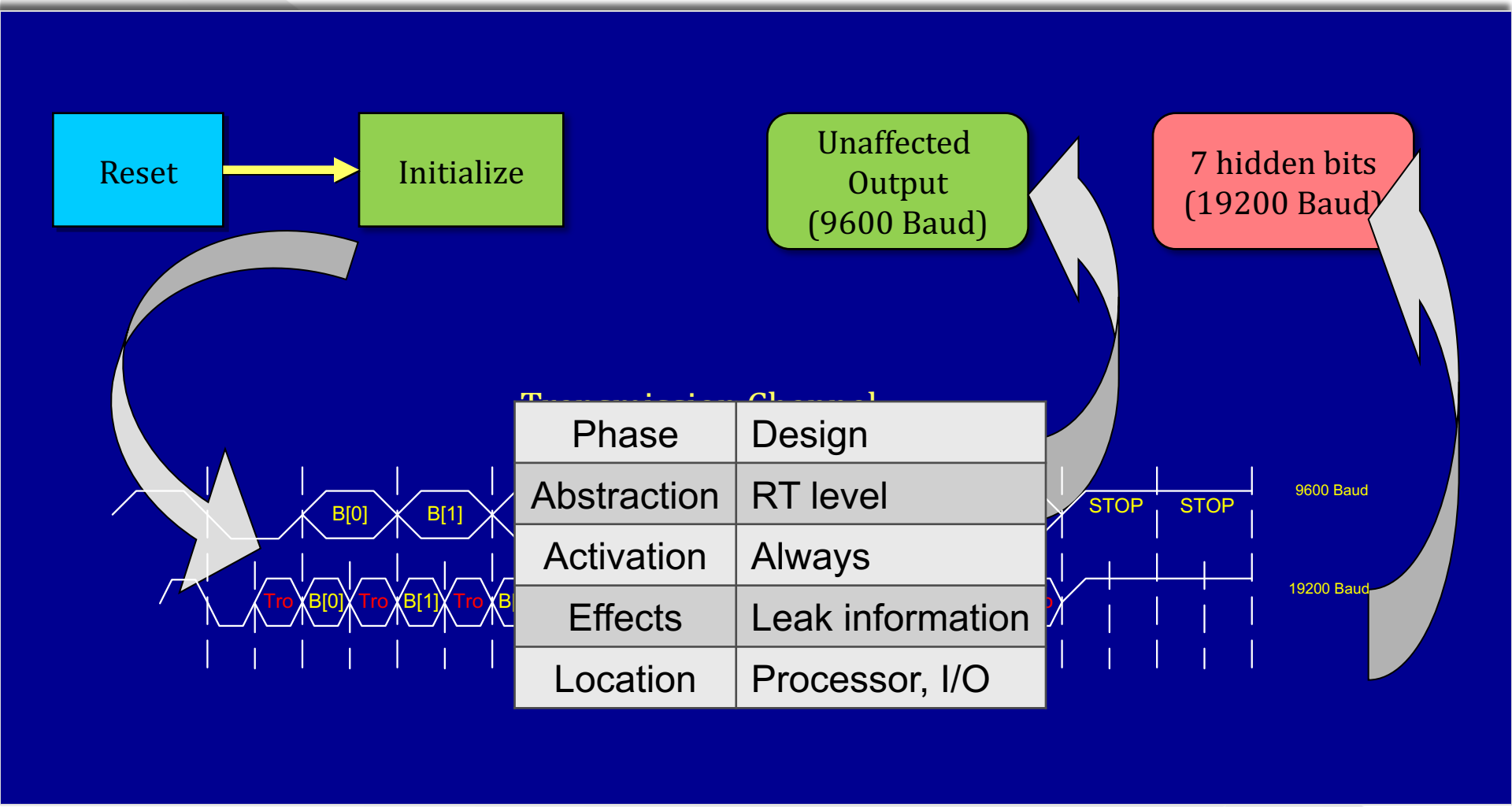
Phase	Design
Abstraction	RT level
Activation	User input
Effects	Leak information
Processor	Processor, I/O



Attacking the Transmission Protocol

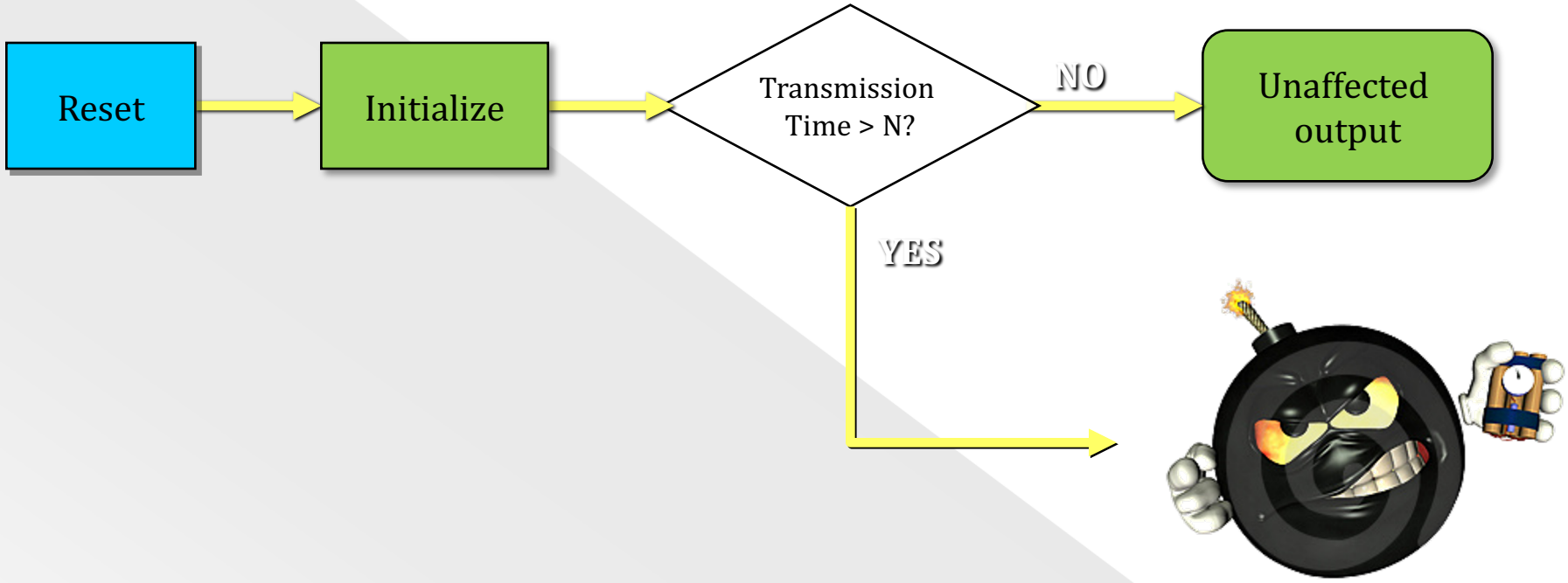


Attacking the Transmission Protocol



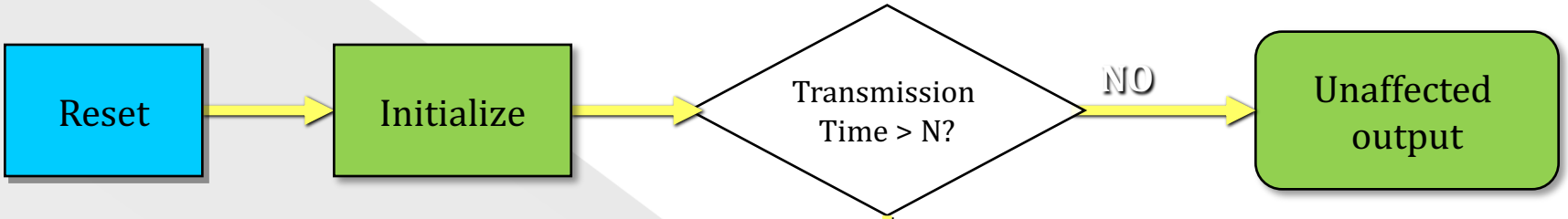
Phase	Design
Abstraction	RT level
Activation	Always
Effects	Leak information
Location	Processor, I/O

Time Bomb



- A large N can help trojan pass the functional tests
- Can be combined with any other payload

Time Bomb

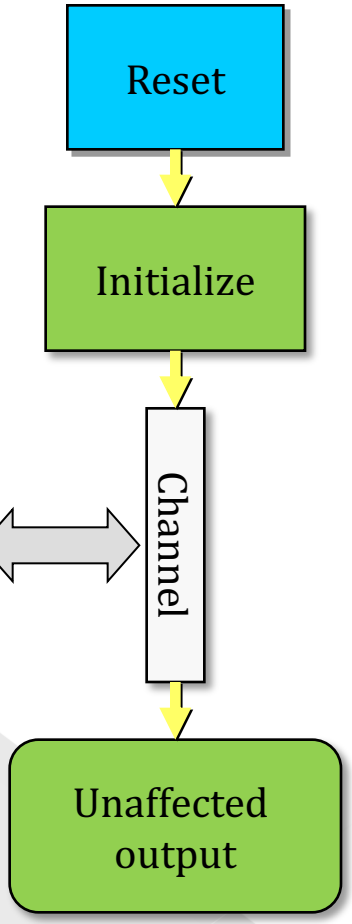


Phase	Design
Abstraction	RT level
Activation	Time triggered
Effects	Change in function
Location	Processor, I/O

- A large N can helpTrojan pass the functional tests
- Can be combined with any other payload

Controlling the device

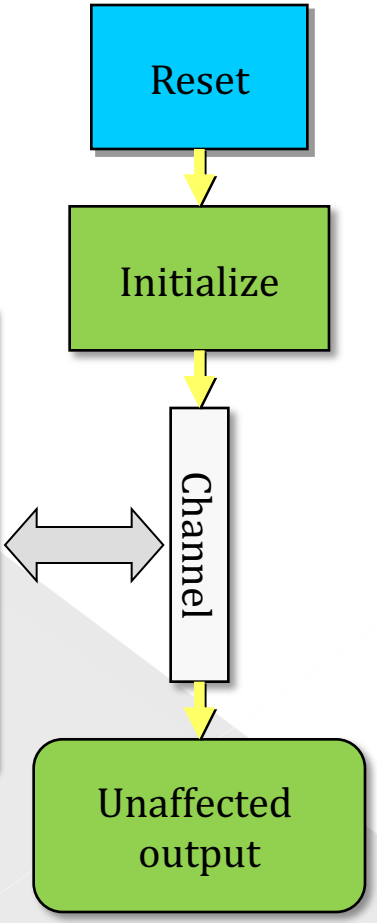
Trojan Commands	Effect
EFTRI	Reset the system
ABTRI	Encrypt the encryption key with the Trojan key
CDTRI	Transmit encrypted key on RS-232 TxD port



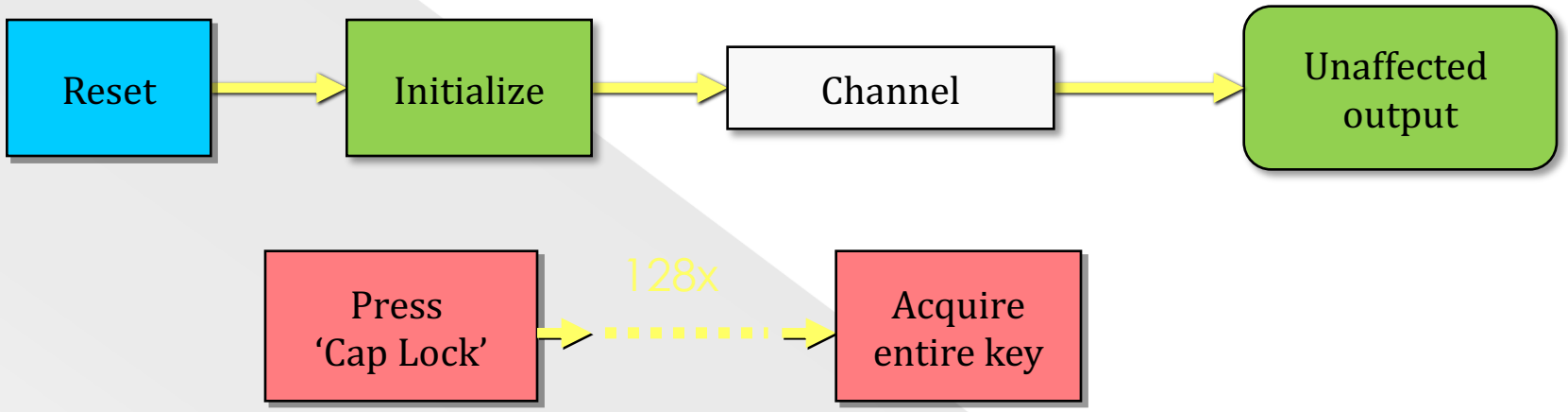
Controlling the device

Phase	Design
Abstraction	RT level
Activation	User input
Effects	Denial of service, leak information
Location	Processor, I/O

Trojan Commands	Effect
EFTRI	Reset the system
ABTRI	Encrypt the encryption key with the Trojan key
CDTRI	Transmit encrypted key on RS-232 TxD port

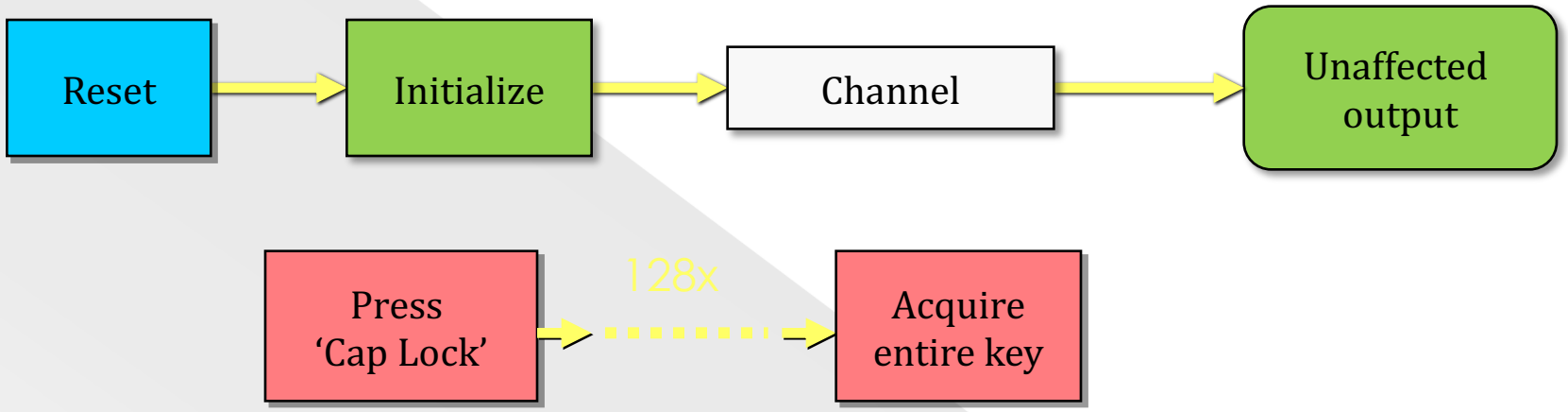


Stealing data from keyboard



- Peripheral devices present another attack path
- It is difficult to detect malfunction of peripheral devices

Stealing Data from Keyboard



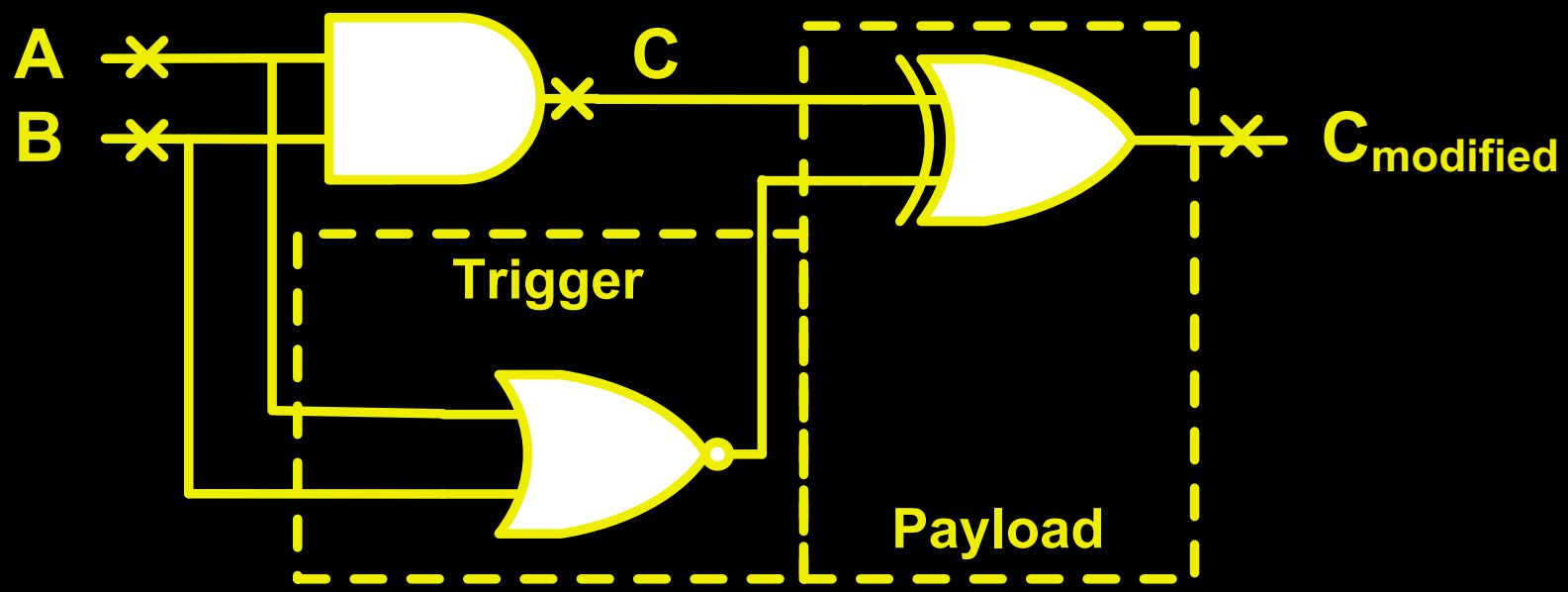
Phase	Design
Abstraction	RT level
Activation	User input
Effects	Leak information
Location	Processor

- Periphe a further attack path
- It is diffi ction of peripheral devices

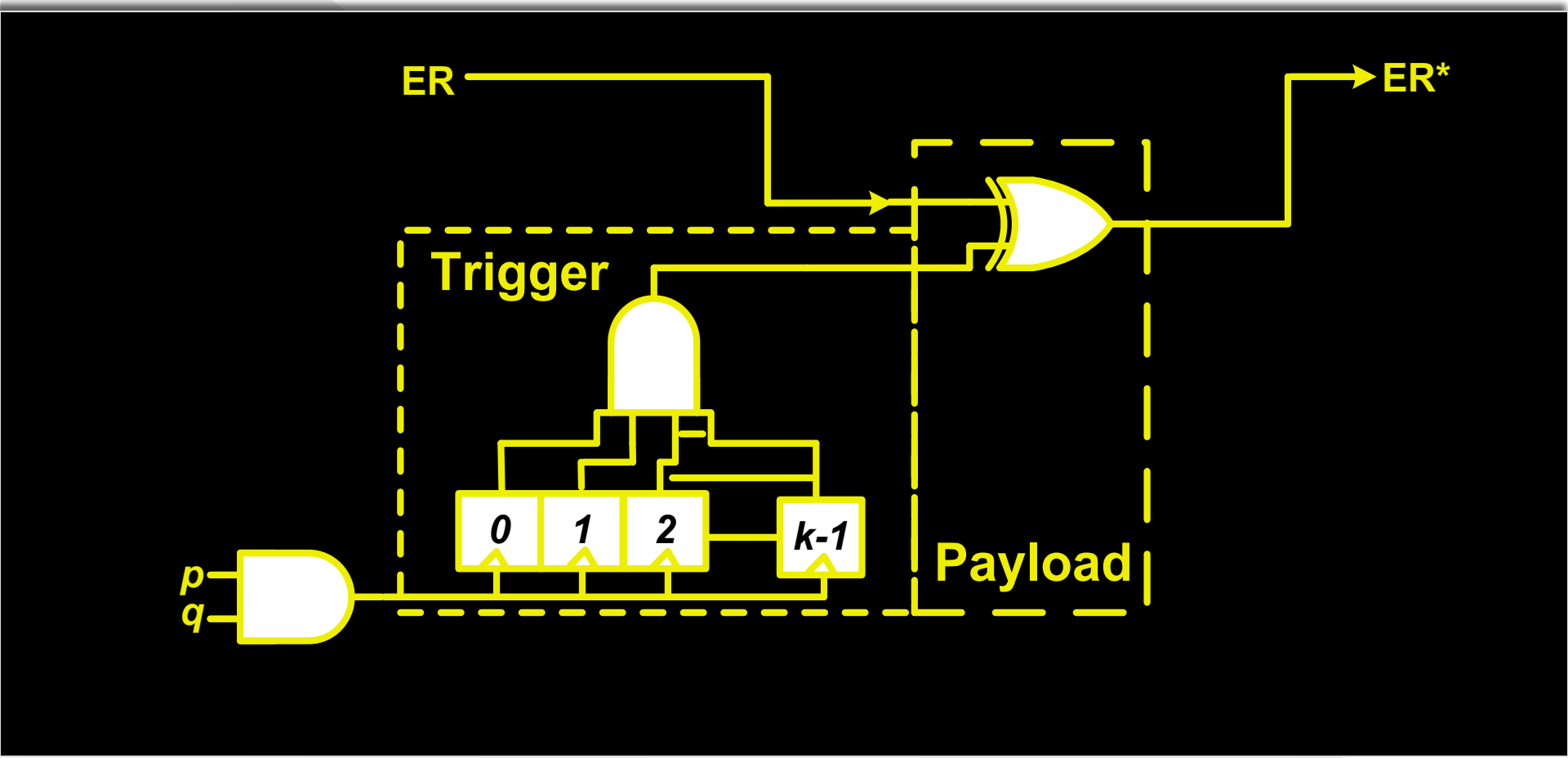
Summary

Trojan	At what phase?		At what level		How is it activated?					What does it do?							Where it is located?			
	Spec.	Design	Dev. env.	RTL	Triggered				Always on	Leak Info.				Change of func.	DOS		Processor	I/O units	Power	Clock
					Internally		External input			RF	Optical	Thermal	Peripheral		Temp.	Perm.				
					Time	Phys. parameter	User	Extl. Comp.												
A1[?]		✓		✓					✓				✓				✓			
A2[?]		✓		✓					✓				✓				✓			
A3[?]		✓		✓	✓										✓			✓		
A4[?]		✓		✓					✓			✓						✓		
A5[?]		✓		✓					✓	✓							✓			
A6[?]		✓		✓					✓	✓							✓			
A7[?]		✓		✓					✓		✓							✓		
B1[?]		✓		✓			✓						✓					✓		
B2[?]		✓		✓		✓	✓						✓					✓		
B3[?]	✓	✓		✓				✓					✓					✓		
C1[12]		✓		✓			✓						✓					✓		
C2[12]		✓		✓			✓						✓					✓		
C3[12]		✓		✓			✓						✓					✓		
D1[?]		✓		✓			✓						✓					✓		
D2[?]		✓		✓			✓									✓			✓	
D3[?]		✓		✓			✓							✓				✓		
D4[?]		✓		✓			✓						✓				✓			
D5[?]		✓		✓					✓				✓					✓		
D6[?]		✓		✓		✓										✓	✓			
D7[?]		✓		✓			✓						✓					✓		
D8[?]		✓		✓			✓				✓							✓		
E1[10]		✓		✓			✓								✓		✓			
E2[10]		✓		✓					✓				✓					✓		
F1[8]		✓		✓			✓									✓				✓
F2[8]		✓		✓			✓									✓		✓		
F3[8]		✓		✓			✓							✓			✓			
F4[8]		✓		✓					✓				✓					✓		
G1[9]		✓		✓					✓		✓							✓		
G2[9]		✓		✓			✓									✓		✓		
G3[9]		✓		✓			✓						✓					✓		
H1[15]		✓	✓	✓					✓	✓								✓		
I1[14]		✓		✓					✓				✓					✓		
I2[14]		✓		✓			✓						✓					✓		
I3[14]		✓		✓			✓									✓		✓		
I4[14]		✓		✓					✓		✓							✓		
J1[4]		✓		✓			✓						✓			✓		✓		
K1[6]		✓		✓			✓						✓							
K2[6]		✓		✓			✓									✓		✓		

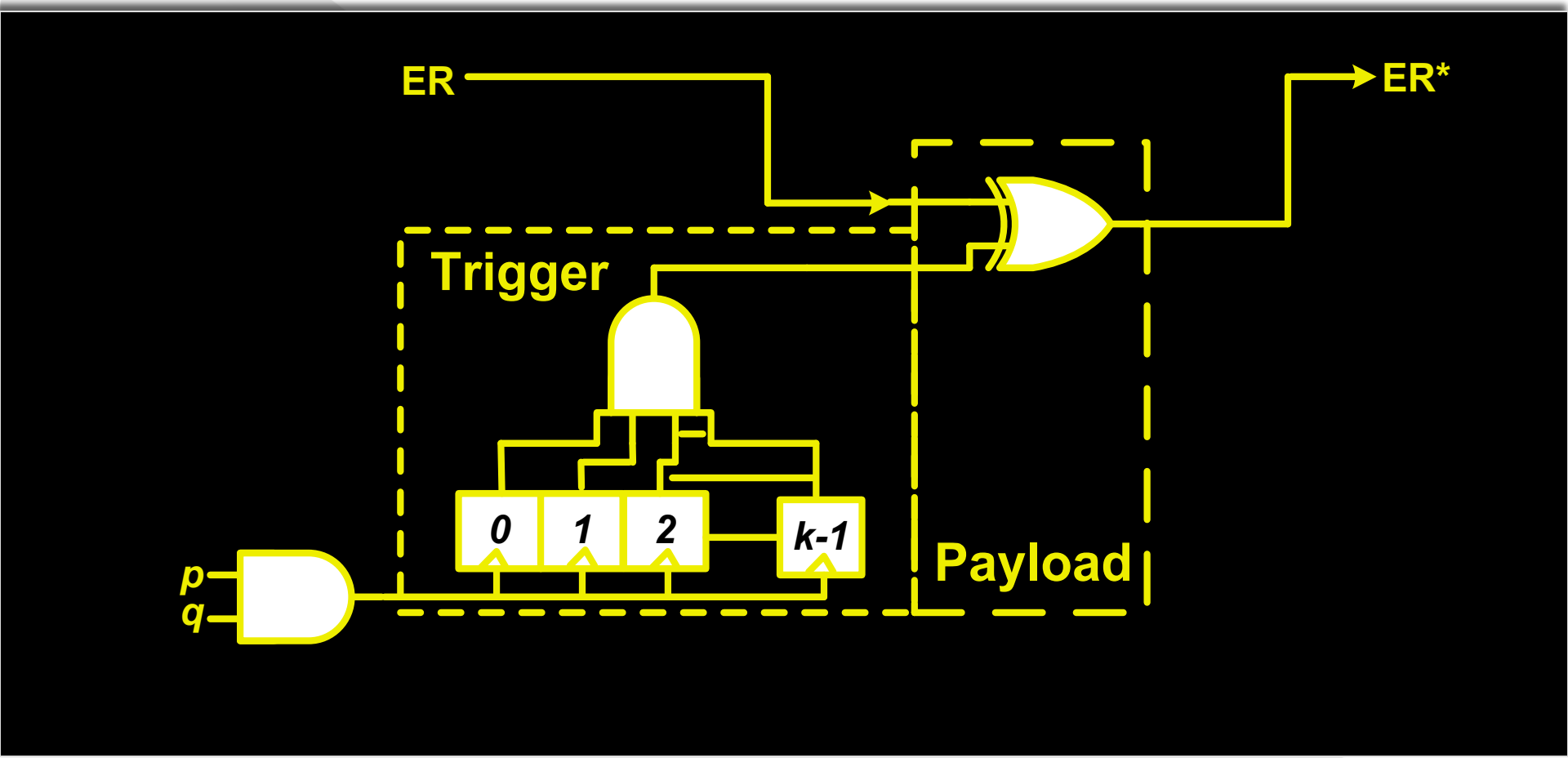
Combinational trojans



Sequential Trojans

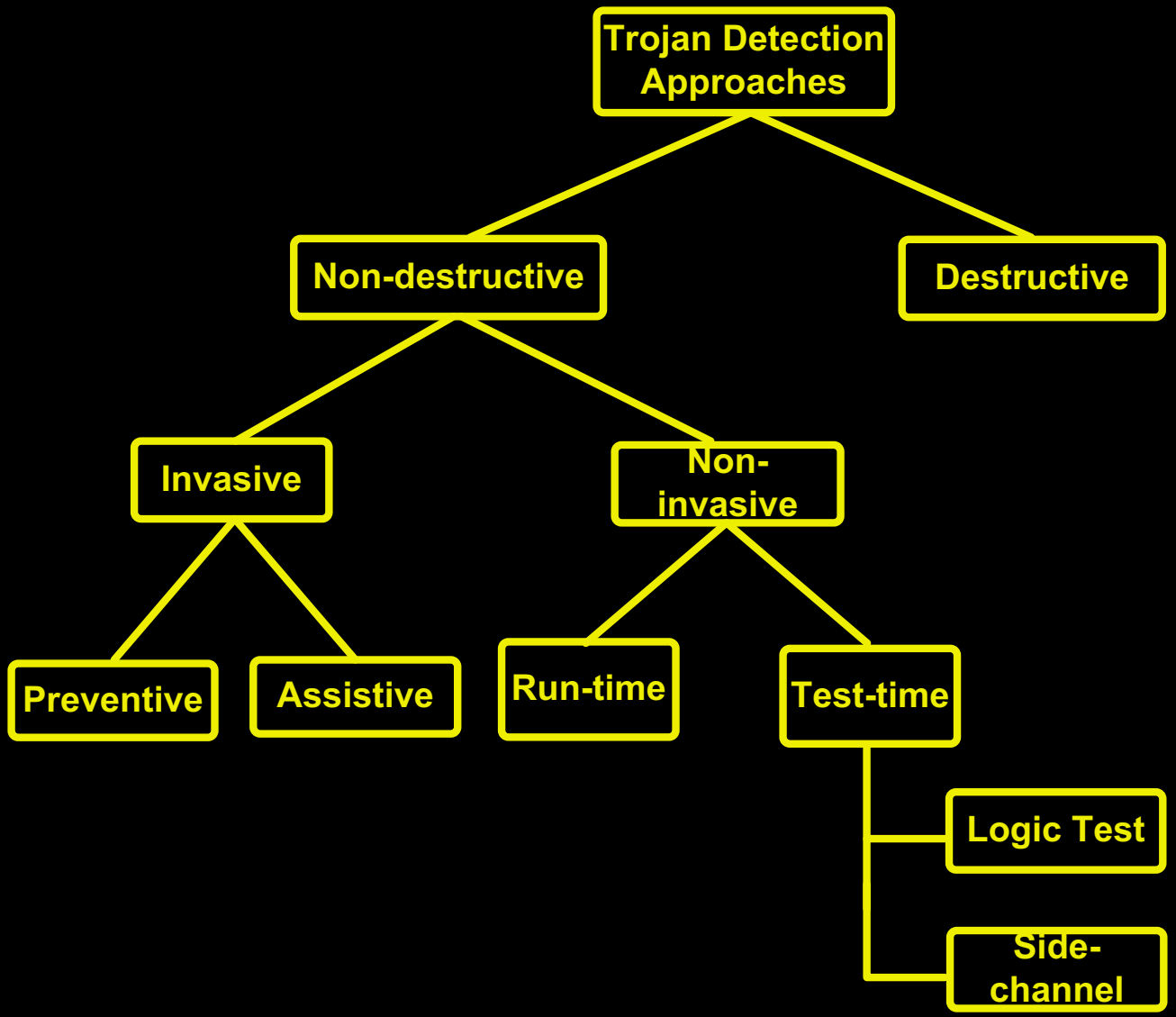


Sequential Trojans

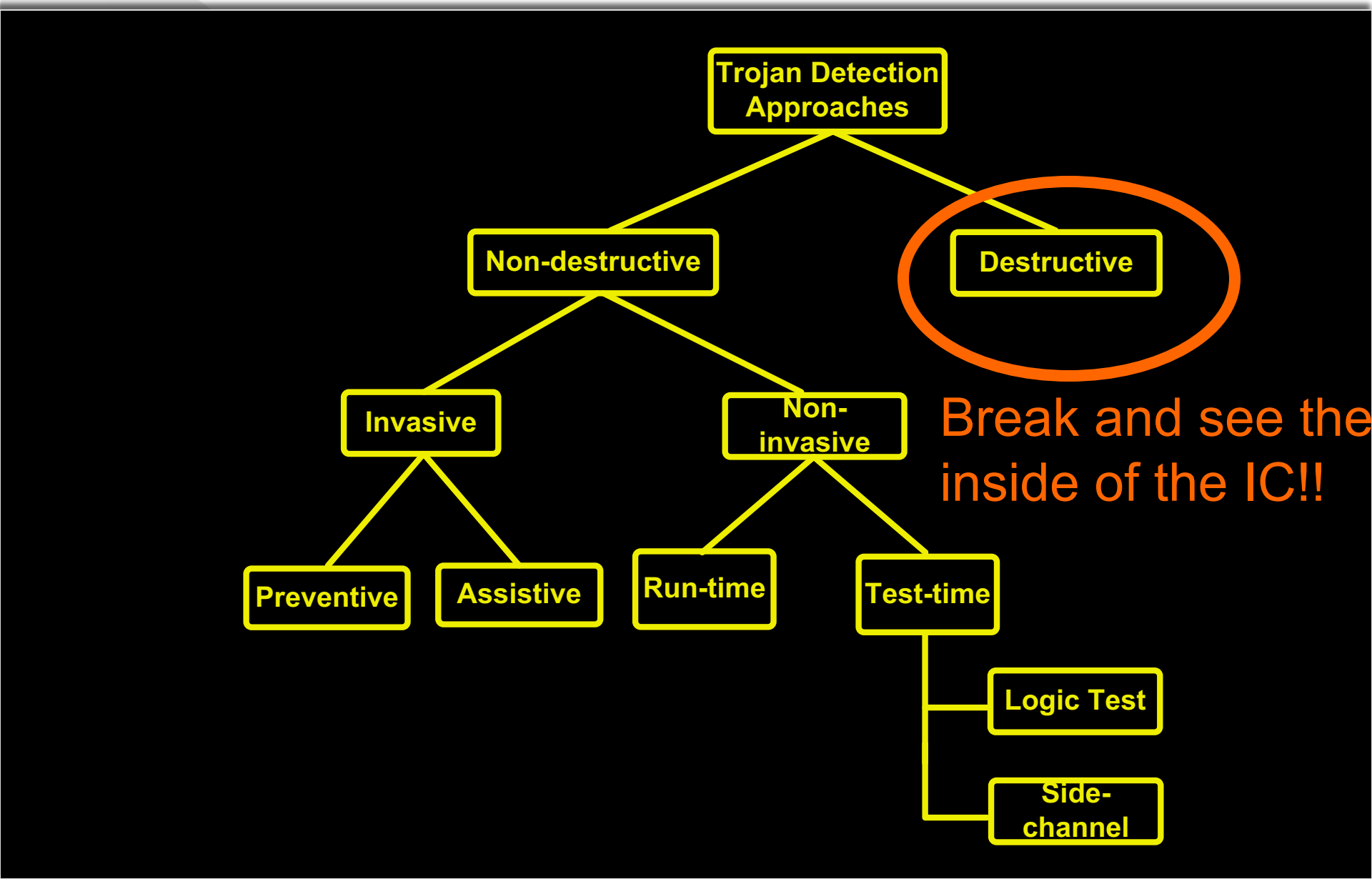


Trojan with K-counter in a N-input circuit needs 2^{N+K} test patterns

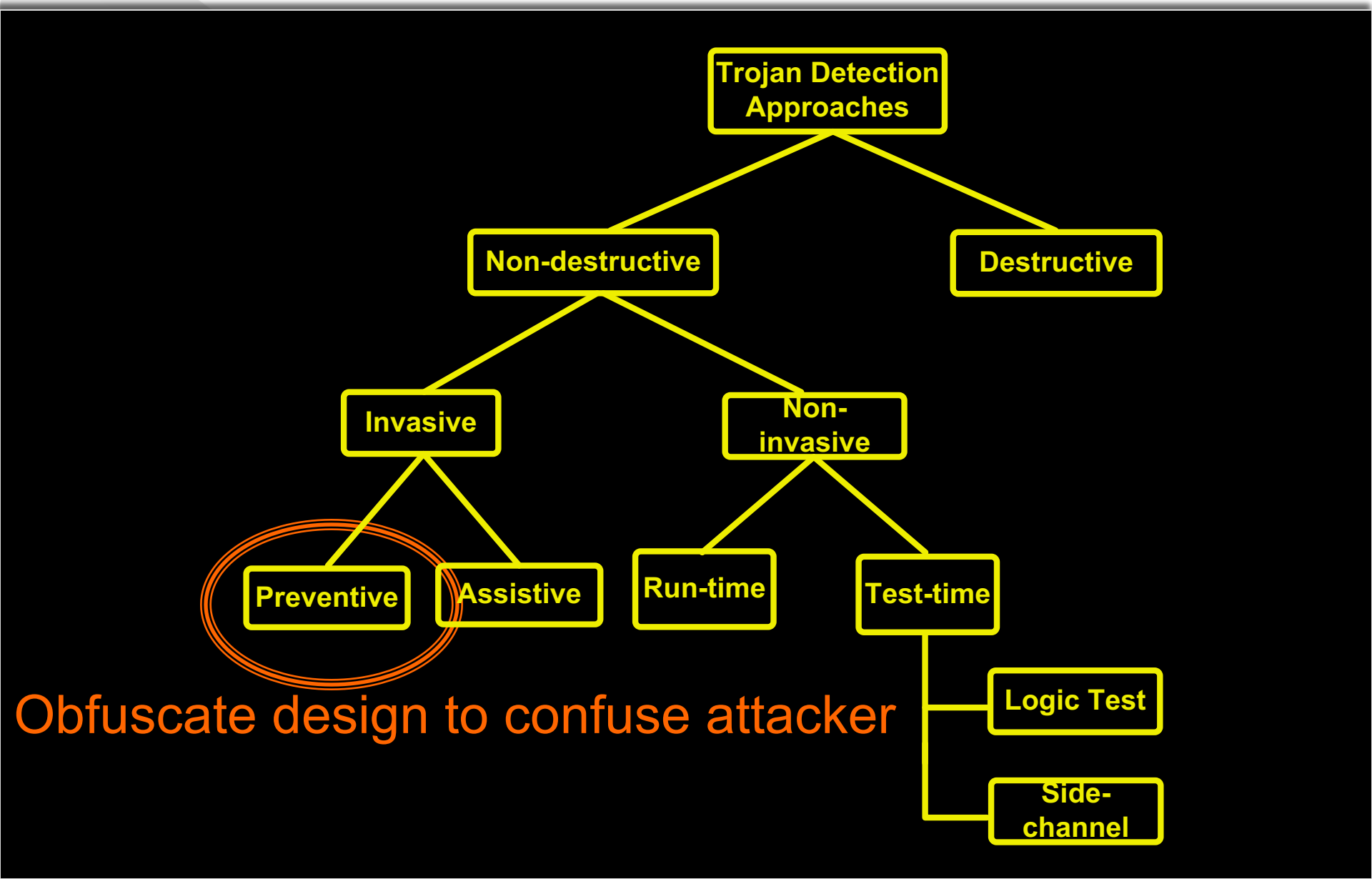
Trojan detection taxonomy



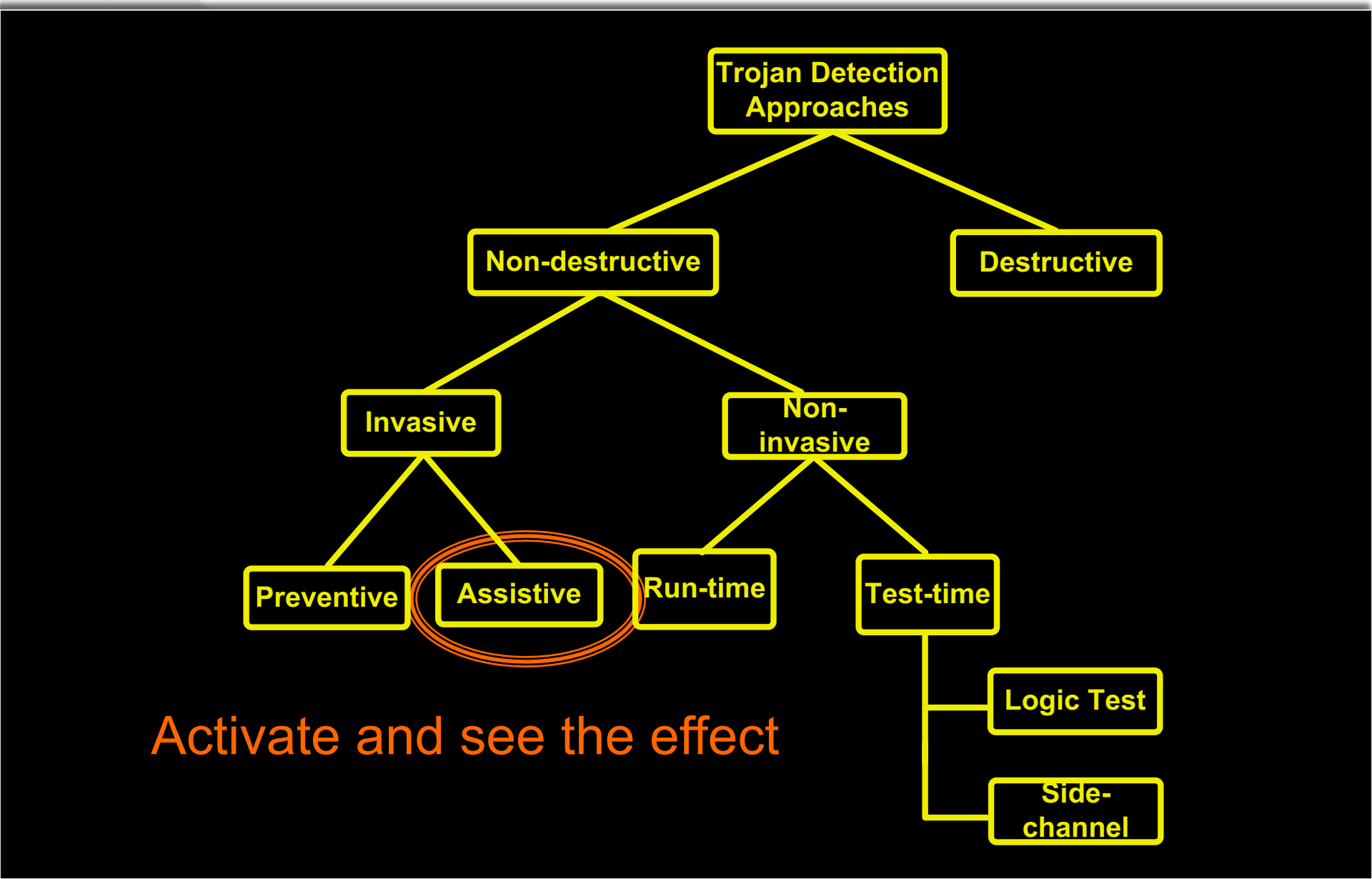
Trojan detection taxonomy



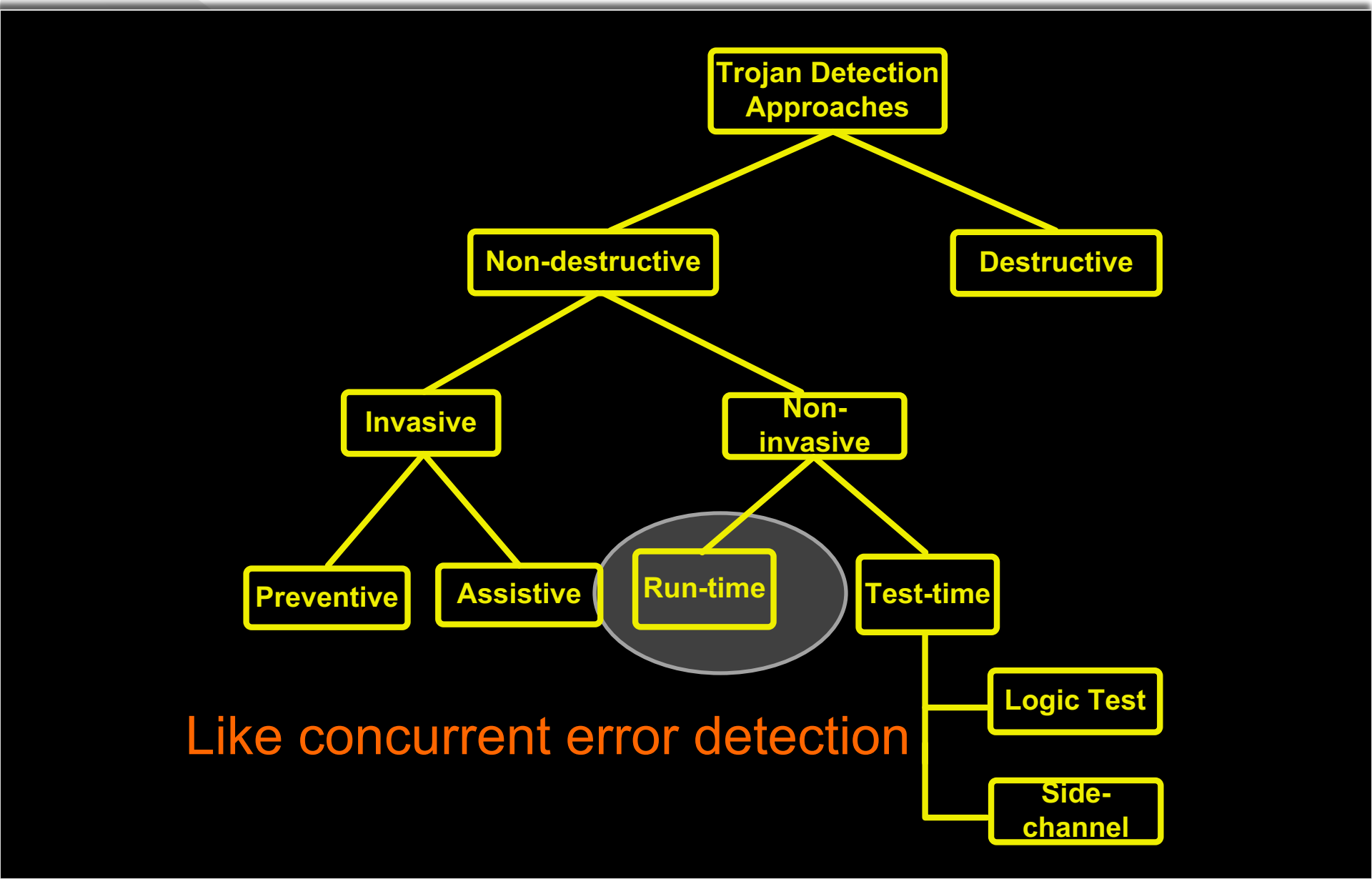
Trojan detection taxonomy



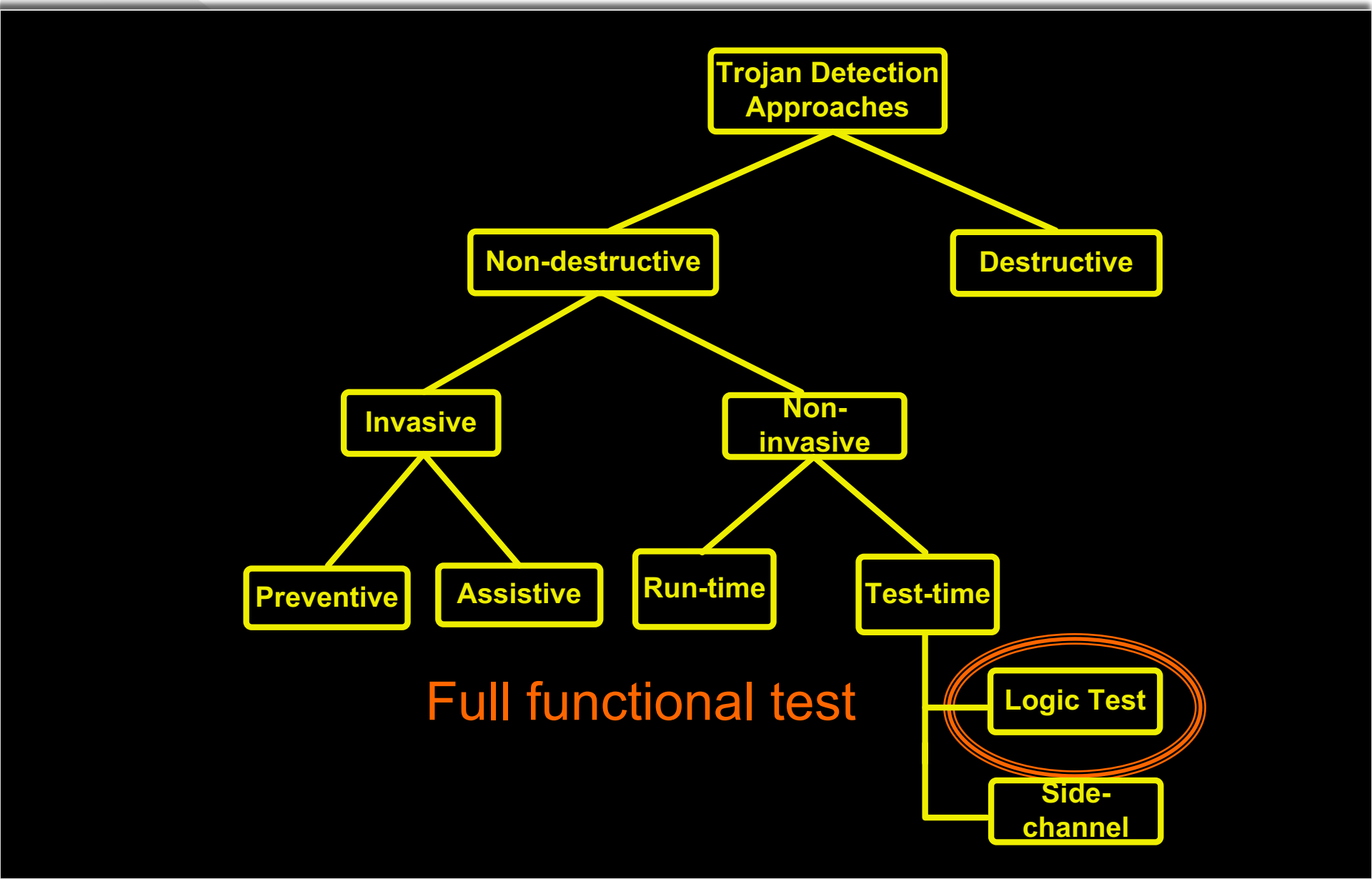
Trojan detection taxonomy



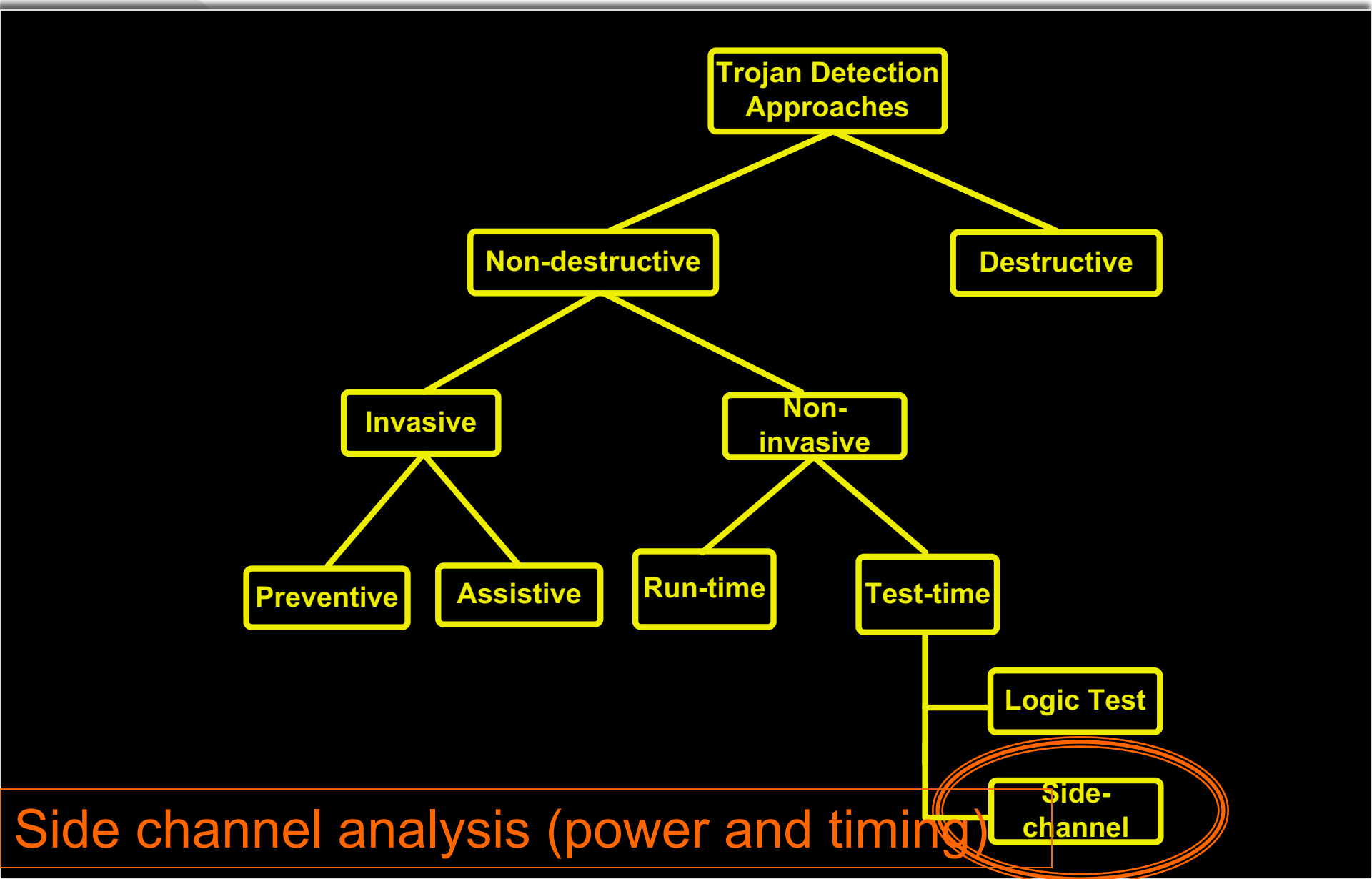
Trojan detection taxonomy



Trojan detection taxonomy



Trojan detection taxonomy

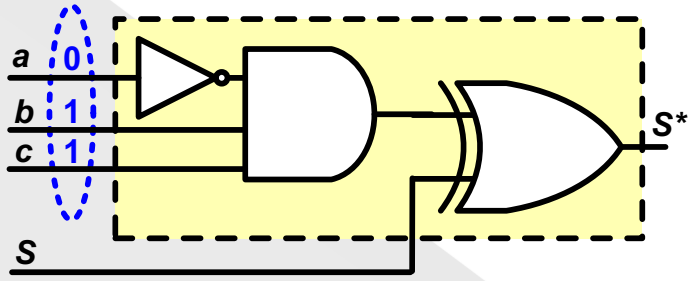


Statistical analysis (MERO-Bhunia et. al.)

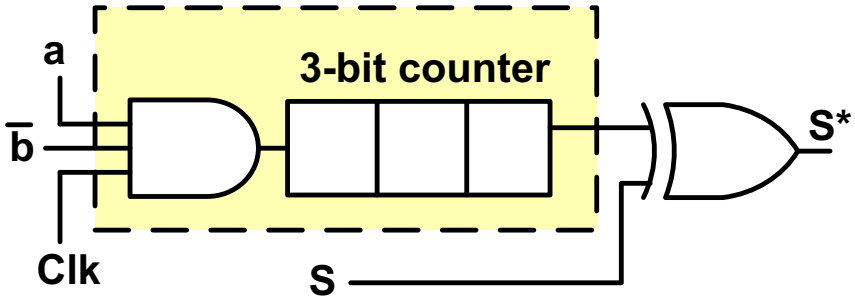
- Assumption: Trojans are inserted at nodes in the circuit which are less controllable and observable
- Solution: increase the activity of these nodes

http://www.iacr.org/archive/ches2009/57470397/57470397.pdf

Trojan: circuit examples



(i) Combinational Trojan

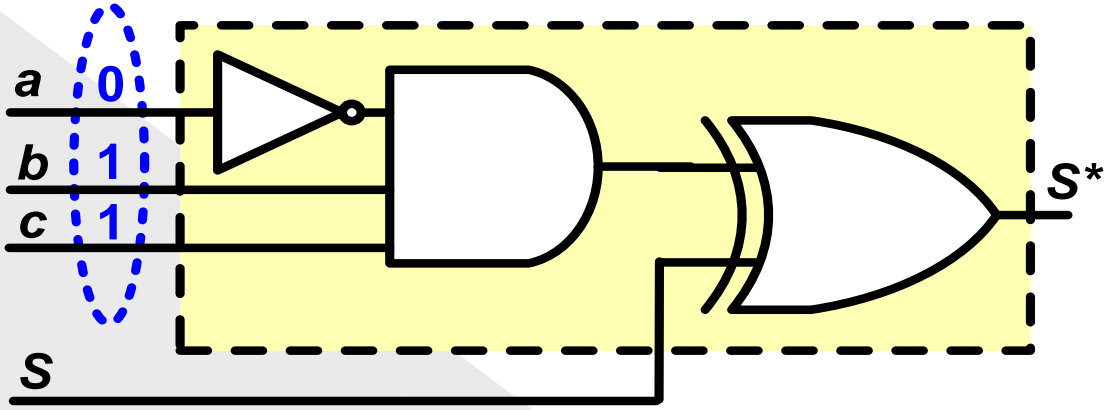


(ii) Sequential Trojan

- Trojan Trigger Condition:
- *(i) a=0, b=1, c=1* *(ii) a=1, b=0*
- Generate vectors to satisfy each of these conditions multiple (*N*) times
- Probability of Trojan activation increases with *N*
- The concept is similar to *N-Detect Tests**

** I. Pomeranz and S.M. Reddy, 2004.*

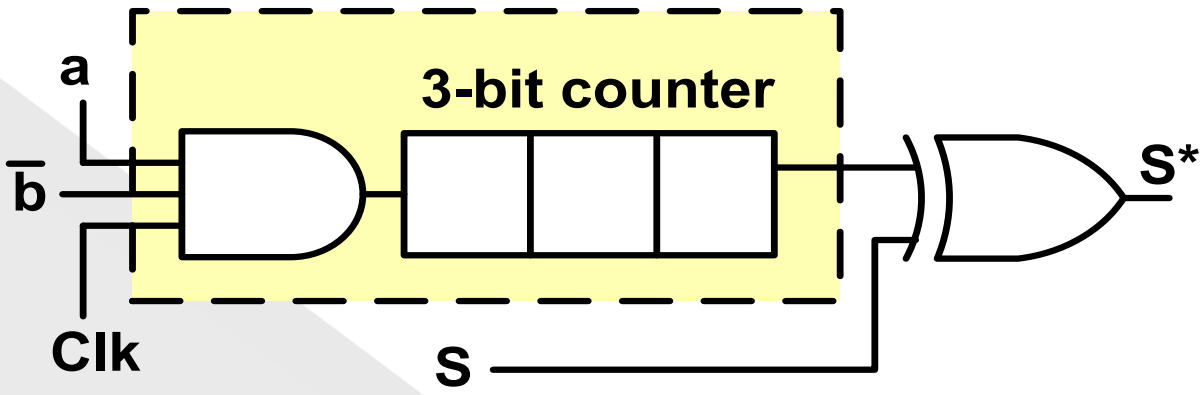
Combinational trojan: example



- Trojan Trigger Condition: *(i) a=0, b=1, c=1*
- Input pattern is 1 out 8 combinations

** I. Pomeranz and S.M. Reddy, 2004.*

Sequential trojan: example



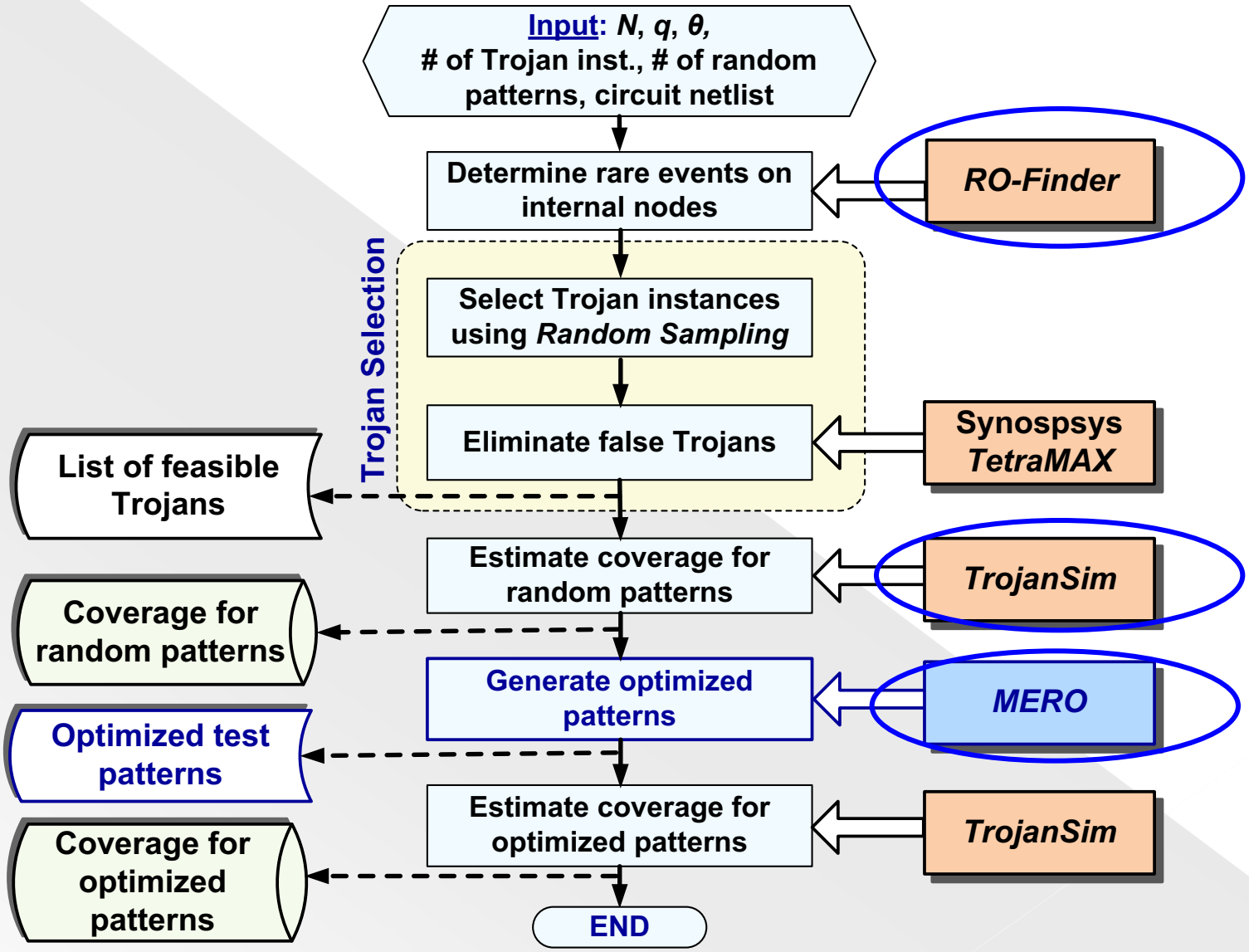
- Trojan Trigger Condition: *a=1, b=0*
- Generate vectors to satisfy each of these conditions multiple (*N*) times
- Probability of Trojan activation increases with *N*
- Concept is similar to *N-Detect Tests**

** I. Pomeranz and S.M. Reddy, 2004.*

MERO test generation steps

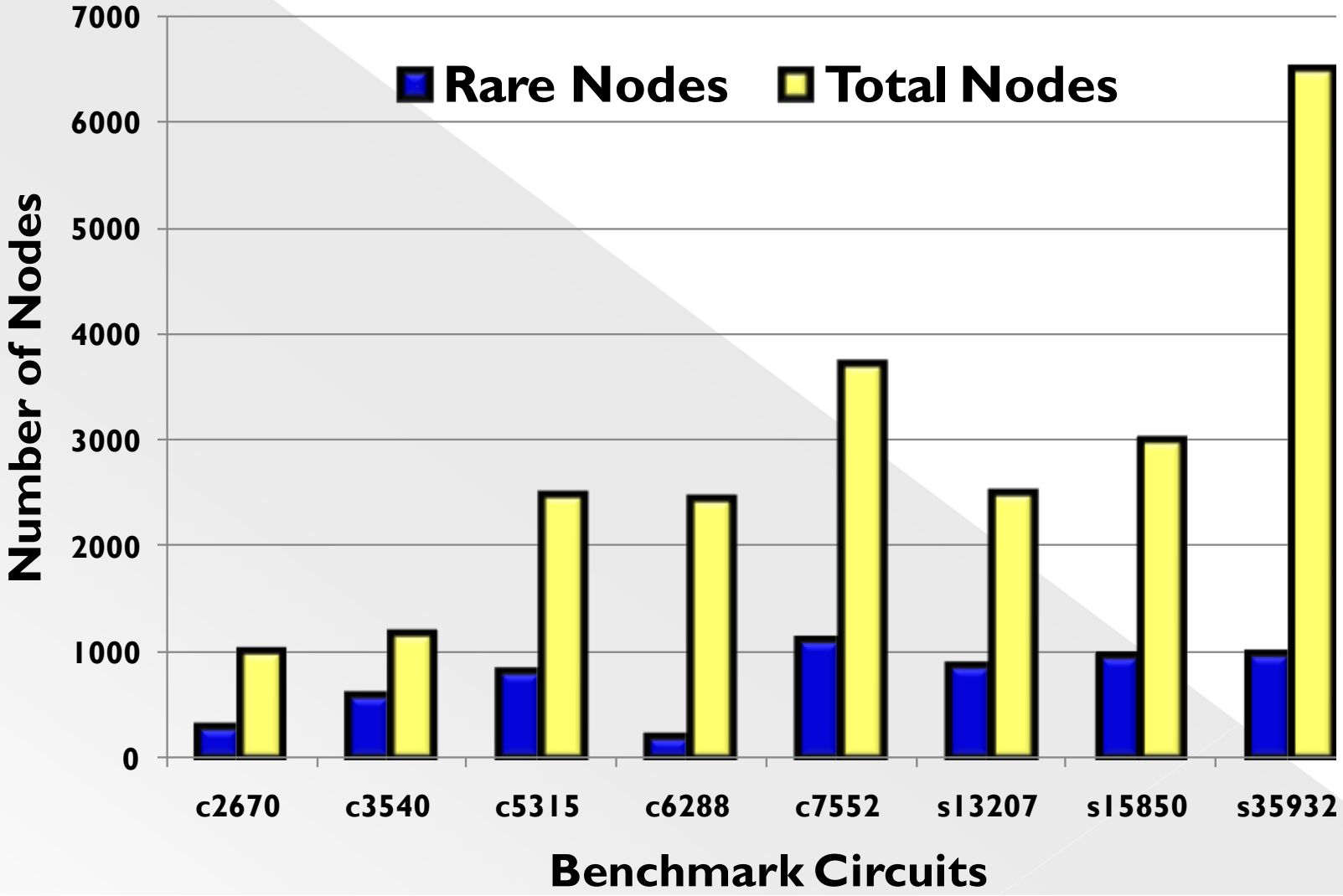
- Determine *rare nodes* and associated *rare values*
- Generate random vectors
- Rank vectors with decreasing rare node trigger probability (*r*)
- For each vector in the ranked list
  - Perturb one/two bit(s) at a time
  - Retain the perturbation if *r* improves
- Stop if all rare nodes are excited to rare values *N* times

MERO implementation and validation



Simulation Results – rare nodes

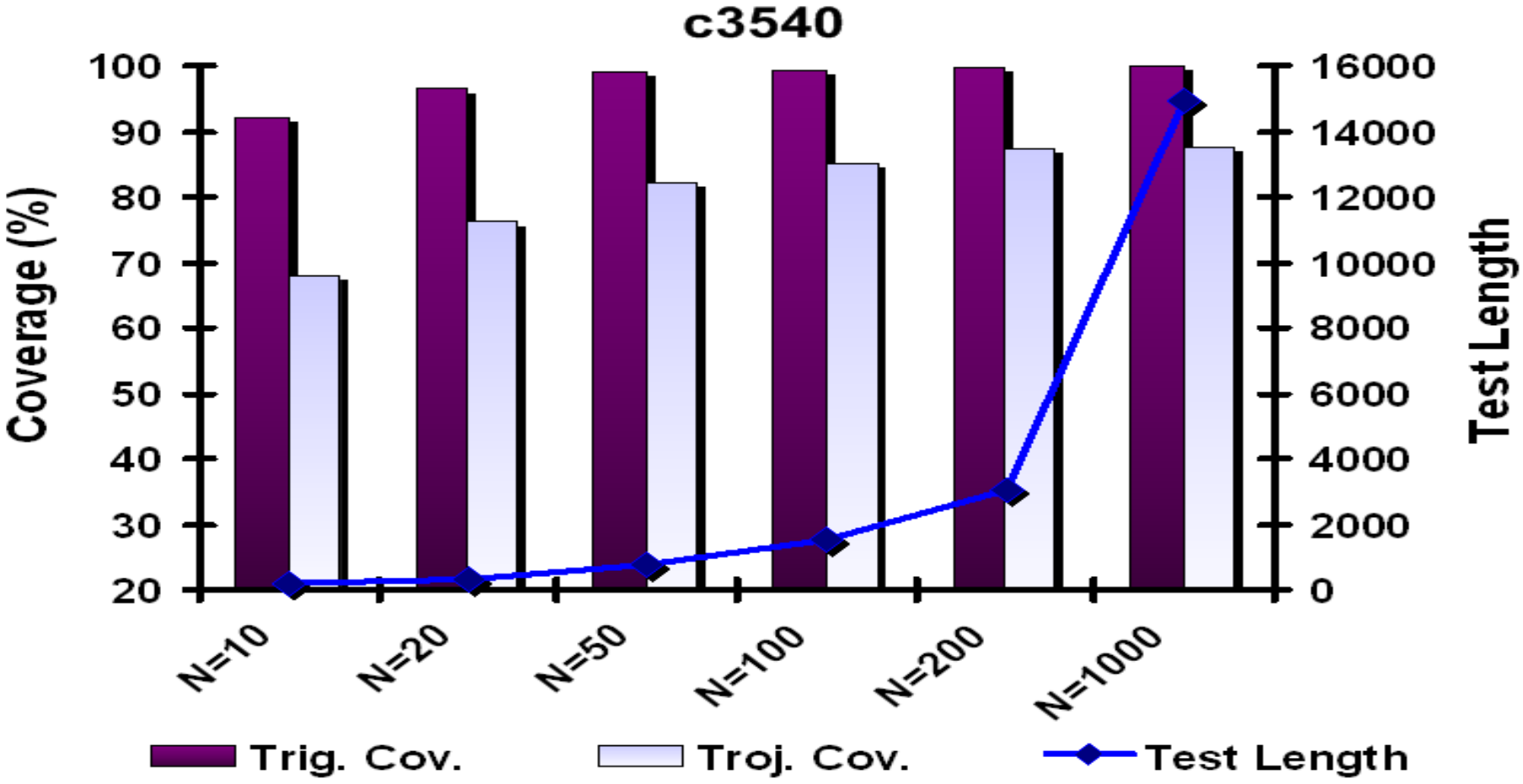
- ISCAS'85 and ISCAS'89 ckts:Comb/seq Trojans
  - # of trigger nodes (q) set to 2 or 4; Θ set to 0.2



Simulation results

Increase in N – more test patterns

Effect of *N* on coverage



N = 1000