University of Pisa

Department of Information Engineering

Hardware and Embedded Security

Nicolò Mariano Fragale

March 2025

Contents

Information

These notes are intended for educational purposes only and cover essential concepts in the field of data systems and security. The aim is to provide a comprehensive understanding of topics such as system vulnerabilities, protection techniques, and defense strategies in cybersecurity.

This document includes topics related to access control, authentication mechanisms, database security, cryptographic methods, and advanced persistent threats, with a particular focus on practical applications in real-world scenarios.

1 MOSFET

Esistono 3 tipi di Silicio:

1. Silicio puro;
2. Silicio di tipo Positivo (p-type) (eccesso di cariche positive);
3. Silicio di tipo Negativo (n-type) (eccesso di cariche negative).

Il silicio puro è poco conduttivo, quindi viene drogato con impurità per renderlo più conduttivo ed essere usato per dispositivi elettronici.

Positivo e Negativo poi deteminano il verso della corrente elettrica.

Il silicio drogato viene utilizzato per realizzare i MOSFET (Metal Oxide Semiconductor Field Effect Transistor) che sono i componenti base dei circuiti integrati.

Struttura di un MOSFET

1. **VDD**: Tensione di alimentazione, rappresenta in logica digitale il valore 1;
2. **VSS, Ground**: Tensione di massa, rappresenta in logica digitale il valore 0;

Il Mosfet si divide in Mosfet di arricchimento e Mosfet di depauperamento.

Consideriamo solo il primo che è formato da 4 regioni: Sorgente, Drenaggio, Gate e Bulk.

MOSFET-N (NMOS):

- Sorgente e Drenaggio sono di tipo N;
- Gate è isolato e riceve il segnale di controllo;
- Bulk è collegato al GND ed è di tipo P.

A bassa tensione il transistor è spento, a tensione alta il transistor è acceso, quindi la corrente passa da Drain a Source quando viene applicata una tensione positiva **VDD** e tensione negativa **VSS** è 0V. Ma soprattutto quando esiste differenza di potenziale tra Drain e Source.

MOSFET-P (PMOS):

- Sorgente e Drenaggio sono di tipo P;
- Gate è isolato e riceve il segnale di controllo;
- Bulk è collegato a VDD ed è di tipo N.

A bassa tensione il transistor è acceso (VSS), a tensione alta il transistor è spento (VDD), quindi la corrente passa da Source a Drain.

NMOS è preferito di gran lunga al PMOS, visto che è piu veloce, resistenza minore e consuma meno energia.

Un circuito **CMOS** è formato da una coppia di MOSFET:

- Un MOSFET di tipo P (PMOS) con la sorgente collegata al VDD;
- Un MOSFET di tipo N (NMOS) con la sorgente collegata al GND (VSS).

I due transistor sono complementari, ovvero quando uno è acceso, l'altro è spento.

Assorbe corrente solo quando cambia stato.

CMOS è usato per realizzare porte logiche (AND, OR, XOR, NAND, ecc.), microprocessori, memorie (SRAM, Flash) e sensori di immagine nelle fotocamere.

1.1 Propagation delay

Il tempo di propagazione è il tempo necessario affinché l'uscita cambi stato dopo una variazione dell'ingresso.

Il ritardo di propagazione misura il tempo tra il cambiamento dell'ingresso e la risposta dell'uscita. Si definiscono due ritardi:

- t_{PLH} (Propagation Delay Low-to-High) → Tempo impiegato dall'uscita per passare da LOW (0V) a HIGH (V_{DD}).
- t_{PHL} (Propagation Delay High-to-Low) → Tempo impiegato dall'uscita per passare da HIGH (V_{DD}) a LOW (0V).

Il ritardo medio si calcola come:

$$t_p = \frac{t_{PLH} + t_{PHL}}{2}$$

Propagation delay

Il ritardo di propagazione è influenzato da 3 fattori:

- Resistenza equivalente dei MOSFET R_{eq};
- Capacità di carico C_L;
- Corrente di commutazione I.

Il ritardo di propagazione è inversamente proporzionale alla corrente di commutazione e alla capacità di carico, e direttamente proporzionale alla resistenza equivalente dei MOSFET.

1.2 Power consumption

- Consumo Statico: corrente assorbita quando il circuito è in stato statico (non cambia stato);
- Consumo Dinamico: corrente assorbita quando il circuito cambia stato.
- Consumo da corto circuito: corrente assorbita quando i transistor sono in stato di corto circuito sempre durante il cambio di stato.

Consumo statico

Il consumo statico si riferisce alla corrente assorbita quando il circuito si trova in uno stato stabile, ovvero senza transizioni logiche. In teoria, i circuiti CMOS hanno un consumo statico molto basso, poiché idealmente un percorso diretto tra V_{DD} e GND non dovrebbe esistere quando il circuito è stabile.

Cause del consumo statico:

1. **Subthreshold Leakage Current:** è la corrente che scorre attraverso un MOSFET anche quando è spento e dipende dalla temperatura e dalle dimensioni dei transistor.

   È più significativa nei NMOS, perché hanno una soglia di tensione inferiore rispetto ai PMOS.

2. **Reverse Bias Junction Leakage:** è la corrente che scorre attraverso la giunzione PN anche quando il transistor è spento ed è maggiore negli NMOS rispetto ai PMOS a causa della più alta mobilità degli elettroni.

Consumo dinamico

Il consumo dinamico è la potenza dissipata quando il circuito commuta tra stati logici, ovvero quando l'uscita cambia da 0 a 1 o viceversa.

$$P_{dynamic} = \alpha \cdot C_L \cdot V_{DD}^2 \cdot f$$

Dove:

- α è il fattore di attività, ovvero la probabilità che il circuito commuti stato;
- C_L è la capacità di carico;
- V_{DD} è la tensione di alimentazione;
- f è la frequenza di commutazione (frequenza del clock).

Durante la transizione LOW $\rightarrow$ HIGH, il PMOS si accende e carica la capacità di carico C_L.

Durante la transizione HIGH $\rightarrow$ LOW, l'NMOS si accende e scarica C_L verso GND.

Maggiore è la capacità di carico, maggiore è l'energia richiesta per la commutazione.

Quindi per ridurre il consumo dinamico si può ridurre la tensione di alimentazione V_{DD}, rifurre la capacità di carico C_L usando transistor più piccoli e ridurre l' attività di commutazione f.

Consumo da corto circuito

Il consumo da corto circuito avviene quando entrambi i transistor NMOS e PMOS sono temporaneamente accesi durante la transizione di stato, creando un percorso diretto tra V e GND.

$$P_{short-circuit} = I_{sc} \cdot V_{DD} \cdot t_{sc}$$

Dove:

- I_{sc} è la corrente di corto circuito;
- V_{DD} è la tensione di alimentazione;
- t_{sc} è il tempo durante il quale entrambi i transistor sono accesi;

1.3 Componenti riciclati e Invecchiamento Fenomeni dell'invecchiamento:

- Negative Bias Temperature Instability (NBTI);
- Positive Bias Temperature Instability (PBTI);
- Hot Carrier Injection (HCI);
- Time-Dependent Dielectric Breakdown (TDDB).

Negative Bias Temperature Instability (NBTI):

NBTI è un fenomeno che degrada le prestazioni dei transistor PMOS quando sono sottoposti a bias negativo prolungato (V_GS ¡ 0, quindi gate negativo rispetto a source). Questo porta a un aumento della tensione di soglia (V_T), riducendo la corrente che il transistor può fornire e rallentando il circuito. In questo modo il transistor chiederà una tensione maggiore per accendersi, maggiore resistenza interna e velocità di commutazione ridotta, infine il circuito risulterà rallentato.

Positive Bias Temperature Instability (PBTI):

Simile a NBTI, ma colpisce i transistor NMOS sotto bias positivo (V_GS ¿ 0) e le cariche tendono a rimanere intrappolate nei difetti dell'ossido di gate. A causa di ciò NMOS si accende più lentamente e gli elettroni si muovo pià lentamente.

Hot Carrier Injection (HCI):

HCI è un effetto di degrado che si verifica quando gli elettroni o i fori acquisiscono energia sufficiente per superare la barriera dell'ossido di gate e rimanere intrappolati all'interno, modificando le caratteristiche del transistor. Questo problema è più critico negli NMOS rispetto ai PMOS, perché gli elettroni hanno una maggiore mobilità rispetto ai fori. In questo caso si riduce la corrente di drain e la velocità di commutazione, facendo si che il transistor diventi meno efficiente.

Time-Dependent Dielectric Breakdown (TDDB):

TDDB è un meccanismo di guasto progressivo che porta alla rottura dell'ossido di gate nel tempo, a causa di uno stress elettrico prolungato. Una volta che il breakdown avviene, il transistor smette di funzionare correttamente. Quindi si perde la capacità isolante dell'ossido, aumentando la corrente di perdita e eventualmente il guasto irreversibile del dispositivo.

1.4 Process Variation Inter-Die Variations: due chip identici prodotti nello stesso o diversi wafer(disco di silicio) possono avere caratteristiche elettriche leggermente diverse.

Alcuni esempi di variazioni sono:

- Differenze nella tensione di soglia;
- Differenze nella corrente di perdita;
- Differenze nel tempo di propagazione.

Intra-Die Variations: variazioni intra-die sono differenze tra transistor all'interno dello stesso chip. Queste possono essere dovute a imperfezioni locali nel processo di fabbricazione.

Alcuni esempi di variazioni sono:

- Variazioni nelle frequenze di funzionamento tra diverse parti del chip.
- Variazioni nella tensione di soglia tra diversi transistor.
- Variazioni nella corrente di perdita tra diversi transistor.
- Degrado delle prestazioni nei percorsi critici.

A fine slide il professore consiglia dei video per approfondire il discorso.

2 Testing

Viene effettuato per verificare le funzionalità dei circuiti integrati e per individuare eventuali difetti dopo che sono stati fabbricati.

Un chip senza difetti è un good chip, altrimenti un bad chip; vengono testati tutti i chip e non tutti sono facili da verificare.

Queste verifiche possono essere sfruttate anche per scovare i chip contraffatti, che di solito sono chip vecchi, infatti piu dell' 80% di chip contraffatti sono chip vecchi.

Si effettuano principalmente 2 electrical test:

1. Parametric test: testa le proprietà elettriche del chip sia in caso di corrente continua (comportamento statico) (DC) che in corrente alternata (comportamento dinamico) (AC):
    - Tensione di soglia dei transistor (DC);
    - Corrente di perdita (DC);
    - Tensione e Corrente di Alimentazione, quindi se il dispositivo opera con la tensione e corrente prevista (DC).
    - Frequenza di taglio, ovvero la banda che passa nel dispositivo (AC);
    - Propagation delay (AC);

- Tempo di salita e discesa del segnale (AC).
- Il livello di disturbo che può interferire con il segnale (AC).
- Vedi slide per altri esempi...

2. Functional test: testa le funzionalità del chip, quindi se svolge correttamente il suo lavoro.
   - Logica digitale: verifica se le porte logiche funzionano correttamente;
   - Memorie: verifica se le memorie funzionano correttamente;
   - Interfacce: verifica se le interfacce funzionano correttamente;
   - Il codice viene eseguito senza errori.

Un altro tipo di test è il Temperature test: verifica se il chip funziona correttamente a diverse temperature.

- Militare: -65°C a 175°C; nel caso di chip militari con l'obiettivo di resistere a temperature estreme;
- Industriale: -25°C a 85°C; per chip che devono resistere a temperature elevate;
- Commerciale: -10°C a 70°C.

Burn-In: testa il chip a temperature elevate per un lungo periodo di tempo per trovare infant mortality, ovvero i chip che si rompono subito dopo la fabbricazione.

Temperature Cycling: testa il chip a temperature alte e basse per verificare se il chip funziona correttamente a diverse temperature.

Per verificare i risultati questi vengono comparati con i risultati attesi, se sono uguali il chip è buono, altrimenti è difettoso.

3 Hardware Metering

Hardware metering (introdotto nel 2005) è un insieme di protocolli che permettono all'autore del prodotto di deterne i diritti successivamente alla produzione e distribuzione.

Questa tecnica quindi vale come un **Fingerprinting univoco** per identificare il prodotto.

Alcuni esempi di autenticazione hardware sono:

1. **On-Chip Aging Sensor:** misura l'invecchiamento del chip e quindi la sua vita utile;
2. **Physically Unclonable Functions (PUFs):** sono funzioni che generano un fingeprint univoco per ogni chip, quindi non clonabile;

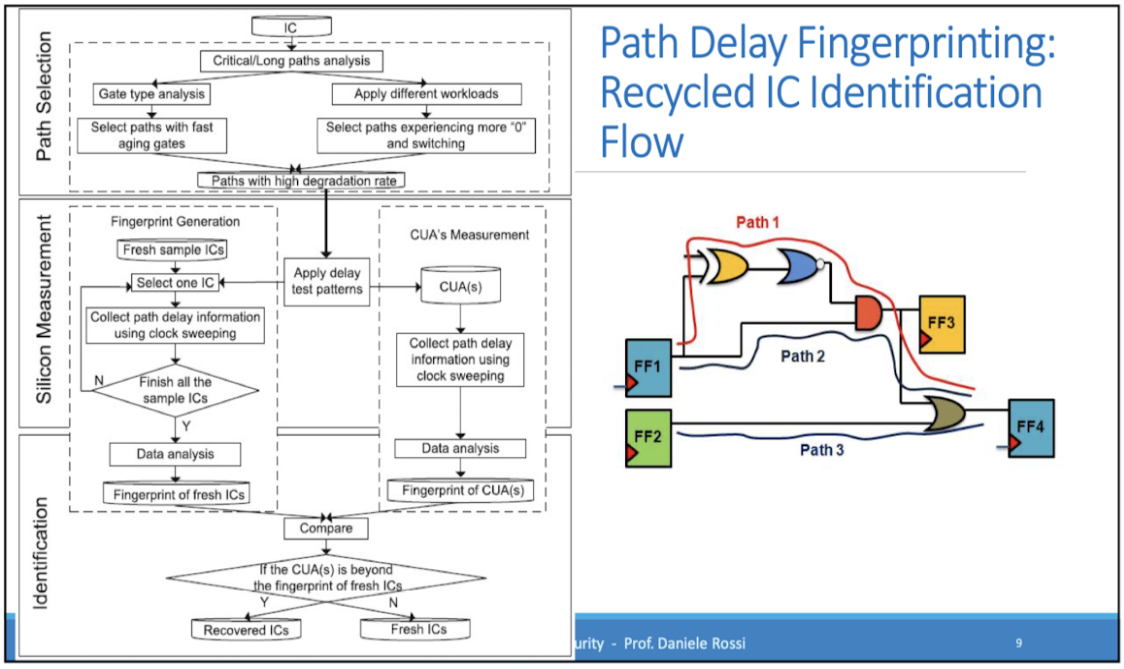
3.1 Path Delay Analysis Path Delay Fingerprinting è una tecnica che permette di identificare un chip tramite il degrado delle prestazioni senza usufruire di ulteriore hardware dedicato alla registrazione di aging.

Con degrado delle prestazioni ci si riferisce al tempo che i chip impiegano per eseguire operazioni funzionali → Ritardi più grandi ↔ Uso prolungato.

(In generale) I test affermano che la più alta percentuale di degradazione si misura durante il primo anno (0%-10%), al termine del 4& anno si arriva circa al 17%.

Il grafico in slide mostra che le porte logiche XOR e NOR si degradano piu velocemente delle porte NAND, e gli inverter di taglia piu piccola si degradano piu velocemente di quelli di taglia piu grande.

L'immagine mostra il processo di identificazione dei circuiti integrati (IC) riciclati tramite Path Delay Fingerprinting.



Teniamo conto di 3 fattori principali:

1. **Proprietà delle porte logiche**
   (a) Analisi delle porte logiche, XOR degradando più velocemente;
   (b) Selezione dei percorsi che contengono maggior numero di pMOS (si degradano piu velocemente degli nMOS);
   (c) Selezione dei percorsi che contengono più switchings (cambi di stato).
   (d) Selezione dei percorsi che contengono più inverter di taglia più piccola (si degradano più velocemente di qulli di taglia più piccola).
2. **Misurazione di silicio e Clock Sweeping**

(a) Si usa il clock del circuito per testare i percorsi, in questo modo si possono analizzare chip gia sul mercato senza che serva un design specifico.

(b) Vengono comparate le misurazioni svolte durante il testing primario con quelle attuali (∀ paths), le misurazioni devono essere effettuate con le stesse condizioni (i.e Temperatura).

(c) Il clock viene fatto variare a diverse frequenze per determinare la frequenza limite alla quale il percorso smette di funzionare.

3. **Identificazione del path**

   Una volta misurati tutti i path si capisce se il chip è nuovo o riciclato.

   L'Analisi Statistica considera:

   (a) Simple Outlier Analysis (SOA): verifica se i ritardi misurati sono anomali rispetto al campione di riferimento (media di IC originali).

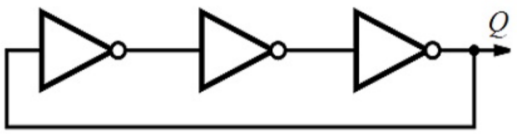
   (b) Principal Component Analysis (PCA): i dati raccolti vengono considerati come un set di variabili in uno spazio multidimensionale che generano dei componenti principali e cose varie, è complesso e articolato e il prof non l ha spiegato.

Se il ritardo del CUA è fuori dal range dei chip nuovi, è probabile che sia un IC riciclato.

3.2 Clock Sweeping Il clock sweeping è una tecnica che permette di identificare IC (circuiti integrati) riciclati basandosi sui risultati che forniscono quando vengono testati. Infatti gli IC devono rispondere in maniera specifica in base alla frequenza del clock e durante questo test gli IC vengono sottoposti a diverse frequenze di clock. Il punto è trovare la frequenza limite alla quale il percorso smette di funzionare, quindi non propaga alcun segnale. In questo modo possiamo misurare il **delay del path**.

I vari modi in cui possiamo performare questa tecnica dipende quindi dal controllo che abbiamo sul manovrare la frequenza di clock.

3.3 Ring Oscillator RO è una tecnica utilizzata per identificare IC contraffatti, in cosa consiste?
Un RO è un circuito costituito da un anello di porte logiche collegate in cascata (di numero disparo), il cui output è collegato all'input successivo.

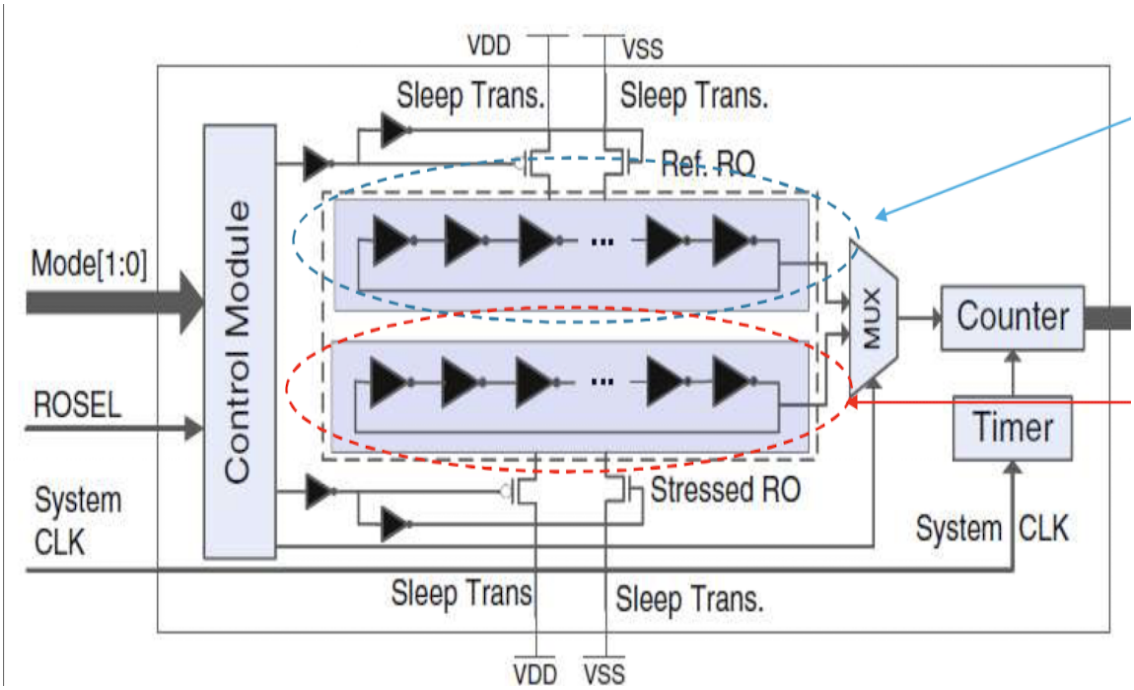


RO evaluate the intrinsic speed of a CMOS.
La frequenza del RO è inversamente proporzionale al numero di stage e alla velocità di commutazione delle porte logiche (propagation delay).

$$f_{osc} = \frac{1}{N(\tau_{LH} + \tau_{HL})} = \frac{1}{2N\tau_p}, \qquad \tau_p = \frac{\tau_{LH} + \tau_{HL}}{2}$$

La frequenza in un RO riciclato è inferiore rispetto a quella di un IC nuovo.
Temperatura e frequenza sono inversamente proporzionale, inoltre sostenere temperature elevate può causare un degrado delle prestazioni del chip.
Il suo comportamento deve essere tale che invecchi a un ritmo simile a quello del circuito integrato (IC) che sta monitorando.

3.4 RO as Sensor



1. Il primo RO (quello sopra) è chiamato il *Reference RO*, è un RO programmato per invecchiare molto lentamente;
2. Il secondo RO (quello sotto) è chiamato il *Sensor RO*, è un RO programmato

per invecchiare al ritmo del chip che si sta monitorando.

3. **N.B:** Da chiedere prima al prof: numero inverter correlati a frequenza
4. **Counter:** conta il numero di cicli del RO, quindi il numero di oscillazioni in a given time.
5. **System clock:** serve a compensare gli effetti negativi del degrado del circuito, mantenendo la precisione temporale costante nel tempo.
6. **MUX:** seleziona il RO da monitorare, è controllato dal **ROSEL** signal.
7. I transistor (pmos e nmos) collegati ai RO servono per attivarli e disattivarli. (se pmos=1 allora RO attivo perche controlla VDD, nmos controlla VSS)

Three mode of operation control:

1. **Manufacturing test mode:** both RO are disconnected to experience no aging;
2. **Functional mode:** Ref RO is disconnected (age a bit bit) and Stressed RO is connected to experience aging; (riflette lo stato di invecchimaneto del chip)
3. **Authentication mode:** both are connected to authenticate the chip.

These mode ensures:

1. The frequency beetwen the two RO will be larger over time;
2. Is impossible to modify RO based sensor during the recycling process.

4 Hardware Trojans

A malicious addition or modification to the circuit used the change its beahavior, reduce the reliability, or leak sensitive information.

- **IP**: intellectual property, most little form of the three;
- **IC**: integrated circuit (i.e CPU);
- **SoC**: system on chip → set(IC) → CPU, GPU, RAM, ROM, etc.;

Hardware Trojans can be inserted in any of these three levels, but the most common is the IC level.
Detection of Hardware Trojans is difficult because they are often small and can be hidden in the design of the circuit and there is no known golden model to compare with. (Those who insert HT are the same who provide golden model (comparing both taint circuits)).
Detection is extremely difficult, both in hardware and software.
HT can be injected during each phase of the design process.

4.1 Implementation

HT consists of:

1. Trojan Trigger: a condition that must be satisfied to activate the HT;

2. Trojan Payload: the action that the HT performs when activated;

Type of Trojan:

- **Functional Trojan:** takes as input nets of the main circuit and restitches (ricuce) with some other nets of the main circuit modifying its functionality;
- **Combinational Trojan:** trigger is activated when is performed (from the main circuit) a specific set of conditions (rare to append);
- **Sequential Trojan:** trigger is activated when a specific sequence of conditions/state is performed/transitions (more common);

4.2 HT Taxonomy:

- Physical characteristics:
  - Distribution: Defines how the Trojan is spread across the IC, whether localized or widely distributed. (Depend by the dead space available on the layout)
    - Tight: when components are topologically close in the layout;
    - Loose: when components are dispersed across the layout of a chip;
  - Structure:
    - Layout Change: The Trojan introduces modifications to the physical layout of the circuit.
    - Layout Same: The Trojan does not alter the overall physical layout but instead exploits existing structures.
  - Size: The relative footprint of the Trojan, which can range from a few gates to a significant portion of the circuit.
  - Type:
    - Functional: HTs realized through the addition or deletion of transistor or gates;
    - Parametric: HTs realized through the modification of existing gates;
- Activation characteristics:
  - Internally Triggered: The Trojan is activated by internal circuit conditions.
    - Always On: The Trojan continuously operates without requiring a specific trigger, is always activate and can disrupt the chip's function at any time.
    - Conditional: The Trojan activates only when specific internal conditions are met.
      - Logic: Triggered by a specific logic state or sequence.

· Sensor: Activated by environmental conditions (e.g., temperature, voltage).

- Externally Triggered: The Trojan is activated by external signals or influences as Antenna or sensor...
  - ∗ Antenna: Triggered via wireless signals (e.g., RF, electromagnetic waves).
  - ∗ Sensor: Activated by detecting external environmental changes (e.g., light, temperature).

- Action characteristics:
  - Transmit Information: The Trojan leaks sensitive data to an external entity.
  - Modify Specification: Alters the design parameters, such as timing or power characteristics such as delay.
  - Modify Functionality: Changes the intended behavior of the circuit.
    - ∗ Change: The Trojan modifies the circuit's normal operation.
    - ∗ Disable: The Trojan disrupts or completely disables certain functionalities.

- Moles: MOLES is a type of hardware Trojan that leaks sensitive information by exploiting side-channels such as:
  - Power consumption;
  - electromagnetic radiation;
  - Path delay.

  MOLES circuits are designed to subtly modify power consumption in a way that depends on secret data (e.g., cryptographic keys). This creates a side-channel that attackers can measure externally to reconstruct sensitive information.

  A critical feature of MOLES is the signal-to-noise ratio (SNR), defined as the power level of side-channel leakage to that of the host IC. An effective MOLES requires a low SNR to evade evaluators' detection, but a high enough SNR for the attacker to extract the secret key bits

Side-Channel Attack exploit the implementations of cryptographic algorithms or software: When performing a side-channel attack, some observable behaviour of the additional information that allows the attacker to decode some cipher text, calculate the cryptographic keys or obtain details of the executed instructions and data within the system

(altri esempi con immagini le trovi nella slide 4)

5 Hardware Trust

Design for hardware trust aims to:

1. **Prevent** Trojan insertion;
2. **Detect** Trojan.

ICs must be designed in such a way that: undetected changes to a circuit are near impossible; and IC will carry out only our desired function and nothing more.

Trojan detection approaches:

- Destructive: expensive and time consuming, require reverse enginnering or golden model IC
- Non Destructive
    1. Run-time monitoring: it uses ML algorithm which are difficult to create, despite this it is a usual approach.
    2. Test-time: throughout test duration
        - Logic Test: consolidated methodology which try to activate in purpose the trojan, unluckily some of them are triggered under rare conditions or externally. Logic test are splitted in Functional test and structural test.
        - Side-channel analysis: include power analysis and time analysis. **Golden chip required!**
            (a) Delay
            (b) Quiescent current
            (c) Transient current
            (d) Radiation: Electromagnetic radiation due to switching activity
            (e) Multiple-parameter

These tests are runned during the Manufacturing or right after.

Functional test Si scrivono testbench e si simulano scenari d'uso reali per vedere se il circuito si comporta come previsto cercando di coprire il maggior numero possibile di stati logici per provocare una possibile attivazione del trojan.
Ha il vantaggio di rappresentare il comportamento reale del sistema.
Ha lo svantaggio di non riuscire ad attivare trojan rari e non considera strutture interne del circuito che non vengono attivate dai test funzionali.

Structural test Il Structural Test lavora a un livello più basso rispetto al Functional Test. Analizza la struttura fisica/logica del circuito, mirando a stimolare ogni nodo logico, ogni flip-flop, ogni porta.
Può lavorare anche senza sapere il comportamento funzionale dal momento che si concentra sulla struttura del circuito.
Ha Più probabilità di attivare trojan ben nascosti. (vantaggio)
Ha lo svantaggio di essere più complesso da implementare.

Se prima venivano runnati piu test funzionali che strutturali, Negli ultimi anni, vista la complessità delle nuove architetture, sono implementati entrambi.

Summary HW Trojans

- are designed to avoid detection, triggering only under rare conditions.
- silent most of their lifetimes and have a very small size.
- connect to nets with low controllability and/or observability
- **Controllability:** ability to induce any signal on line (activation of the signal). Require the capability to control the signal in logic state → Controlling a node at level n, requires controlling the nodes at levels 0...n-1.
- **Osservability:** propagation's effect of the signal to output pins. Observing a node at level i usually requires controlling some nodes at levels i .. n-1.
- Trojans typically change a design's parametric characteristics.

Power analysis Power-based side-channel signals provide visibility of the internal structure and activities within the IC, enabling detection of Trojans without fully activating them (partial activation). HT do not need to be activated.

- **IDDQ:** Extra gates will consume leakage power
- **IDDT:** Extra switching activities will consume more dynamic power

Current measurement consists of:

1. Main circuit current:
2. Measurement noise:
3. Process variations
4. Trojan

Do more test and If any measurment is noticible different from the golden chip raise a Warning.
The power consumed from the trojan depends by its activated cells. (again partial activation) Detection sensitivity depends by:

- Trojan size: direttamente proporzionali

- Circuit size: inversamente proporzionali

Hard to detect using power analysis are:

- Distributed Trojans
- Hard-to-activate Trojans

Time analysis Timing-based side channels can detect a Trojan's presence if the chip is tested using efficient delay tests that are sensitive to small changes in the circuit delay along the affected paths and that can effectively differentiate Trojans from process variations.
HT do not need to be activated.
Additional gates, wiring introduce extra capacitance that will increase path delay.
A change in physical dimension of the wires and transistors can also change path delay.
Delay depends by:

1. Conductivity of the circuit, ? proporzionale
2. Supply voltage, ? proporzionale
3. Capacitators, ? proporzionale

Possible delay-based technique:

- **Path delay fingerprinting** (similar to the technique seenfor counterfeit detection)
- Shadow Register

Shadow Register

Shadow-register provides a possible solution for measuring internal path delay.
Shadow register clock's negative skew is gradually increased until the shadow register stores a different (incorrect) value than its main register → we can obtain delay of comb. path as a function of skew amount.

- Lo *skew negativo* significa che il clock dello shadow register arriva **prima** rispetto al clock normale.
- Aumentarlo gradualmente vuol dire farlo arrivare sempre un po' prima alla volta.
- A un certo punto, lo shadow register legge il dato **troppo presto** → il valore dalla logica combinatoria non è ancora pronto → legge un valore errato.
- Quel momento preciso indica: *"Ecco, il dato ci mette troppo, arriva dopo questo skew"*.
- Quindi possiamo **calcolare il ritardo effettivo** della logica combinatoria.

Se il ritardo misurato è **più alto del previsto**, questo può indicare la presenza di un **Hardware Trojan** all'interno del cammino logico.

Its limitations are:

- Process Variation
- Overhead
- Shadow-Clock, Richiede una clock tree separata, con controllo preciso dello skew.

6 Saponara

6.1 The role of HW in cybersecurity: HW security and trust and HW-based security. Slide 1

The main roles of hardware (HW) when dealing with security is to guarantee that the hardware used is secure and can be trusted.
Hardware Security and Trust are different concepts but are strictly correlated and are needed together: you can have a secure Hardware but if it is not identified, validated and certified you may be not trust it.
If the hardware is corrupted, all the mechanisms introduced to make the software secure (at any level) are useless. A trusted and secure Hardware is needed to protect other system components.
If your Hardware is not trusted then also the software and data communication on it are not trustable.

HW Security 3 steps to achieve HW security:

1. Analyze HW vulnerabilities;
2. Analyze possible ways of hardware attacks → develop tecnique aimed to prevent and mitigate attacks;
3. Implement protection solutions;
4. OTA example in slide

Hardware Security issues can be faced:

1. During the design and production phase (optimal);
2. When is already operating in the field (not optimal);

HW trust A trusted component, operation, or process is one whose behavior is predictable under almost any operating condition and which is highly resistant to subversion by application software, virus, and a given level or physical interference.
An entity can be trusted if it always behaves in the expected manner for the intended purpose.
An asset treated or owned by an Information System component must come from an entity that is able to prove, beyond any reasonable doubt, its originality and genuiness.
Hardware trust mainly concerns Authenticity, that can be verified statically checking for counterfeiting and dynamically using intrusion detection system, IDS, to verify that during the life cycle a virus/malware has not taken the control of an hardware.
A counterfeited HW is not a safe and trust HW.

HW-based security Refers to all those solutions aimed at resorting to hardware devices to protect the whole system from attacks that exploit vulnerabilities of other components of the system itself.
To offer security features to upper layers, hardware itself must be secure at first. From this point of view, Hardware Security play the role of a key enabler for Hardware-based Security.
It does provide a "chain of trust" rooted in silicon that makes the device and extended network more trustworthy and secure.
Hardware-based Implementations can be clustered as:

1. System level solutions. There are 2 standards:
   - Trusted platform module (TPM)
   - Trusted execution enviroment (TEE)
2. Architectural level solutions, improve the security of the CPUs and of the involved memories (example in slide)
3. Security-oriented components, is a Set of custom, special purpose components used for performing specific security-oriented operations, including: Smart cards, secure storage, random number generator, secure boot process ...
4. proprietary solutions VS open security platforms, Intel, AMD ... VS Open HW which include HW accelerator, anti tamper and secure boot process (i.e SEcube, USB Armory, OpenTitan)

6.2 Counterfeiting in electronics: types, sources, consequences and threats. Slide 1 e Slide 2

Types Tipi di contraffazione:

- Recycled, maoyority of counterfeit incidents (more then 80%).
- Remarked, maoyority of counterfeit incidents.
- Overproduced;
- Defective;
- Cloned;
- Forged documentation (i.e fake certifications);
- Tampered (i.e backdoor).

Sources and consequences Complexity of the electronic systems significantly increased over the past few decades and To reduce production cost, they are mostly fabricated and assembled globally. However This globalization has led to an illicit market willing to undercut the competition with counterfeit and fake parts.
Financially, counterfeiting is costing the semiconductor companies billion of dollars and is putting thousands of jobs at risk.
The consequences of an insecure IC supply chain are not only limited to major financial losses, they also pose national security threats.

6.3 HW-based security: discuss TPM and TEE. Slide 1 e 4a

Trusted Platform Module - TPM Describe the guideline for developing chips with strong cybersecurity features.

Trustwothiness of TPM is based on different *Root of trust* components and how interact each other.

Trusted execution enviroment - TEE TEEs are secure areas of a System-on-Chip (SoC) that guarantee code and data protection.
TEE consists of a set od hw and sw components providing facilities necessary to support application.
TEE needs a hw isolation mechanism and a OS runnning on top of isolation mechanism.

1. Software
2. OS, Hypervisor
3. Hardware

TEE not include only a secure hardware but also secure software components for the low-leves SW (i.e for OS, drivers ...).
Instead, TPM refers only to the fact the hw has some built-in security features.
N.B: A TPM may be a component of a TEE, but an alone TPM dont guarantee a TEE.

6.4 Discuss chip design & fabrication flow and sources of threats in the flow. Slide 2

6.5 How to detect chip counterfeiting and its limits.

1. is out of specifications
2. not produced and consequentelly not conform to OCM (Original component manufacturer)
3. incorrect, false, multiple markings or documentations
4. Devices which have been refurbished, but represented as new product
5. misrepresentation of an IC

The difference beetween overpdroduction and cloned is that the second do not have the authorized IP. Limitations:

1. A chip may fail at one particular structural test pattern;
2. Defect will appear in normal operation
3. It will fail at some early point
4. Fail to perferom at the design specifications

6.6 Make examples of security issues due to unsecure hardware in defence, energy, medical and automotive domains. slide 3, 4a e 4b

Defence

- **Xerox 914** Xerox 914 was a copy machine used in soviet embassies all over the world. The machine was so complex that the CIA used a tiny camera designed by Zoppoth to capture documents copied on the machine by the soviets and retrieved them using a Xerox repairman right under the eyes of soviet security.
- **Pentagon**

The Pentagon is worried that "backdoors" in computer processors might leave the American military vulnerable to an instant electronic shut-down. Those fears only grew, after an Israeli strike on an alleged nuclear facility in Syria. Many speculated that Syrian air defenses had been sabotaged by chips with a built-in 'kill switch" — commercial off-the-shelf microprocessors in the Syrian radar might have been purposely fabricated with a hidden "backdoor" inside. By sending a preprogrammed code to those chips, an unknown antagonist had disrupted the chips' function and temporarily blocked the radar."

This all had a very familiar ring to it. Those with long memories may also recall exactly the same scenario before: air defenses knocked out by the secret activation of code smuggled though in commercial hardware.

This was back in 1991 and the first Iraq War, when the knockout blow was administered by a virus carried by a printer : One printer, one virus, one disabled Iraqi air defence.

Fake Cisco routers risk "IT subversion"

- An internal Federal Bureau of Investigation presentation states that counterfeit Cisco routers imported from China may cause unexpected failures in American networks. The equipment could also leave secure systems open to attack through hidden backdoors.
- $76 million **fake Cisco routers**

Energy Theft Going From Bad to Worse

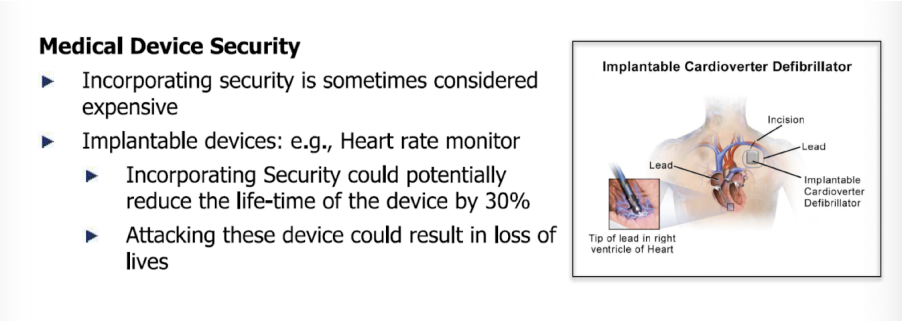
- Tampering with "smart" meters
  - Oil, electricity, gas, ...
- $1B loss in CT because of electricity theft

Energy

Medical

Automotive

- In 2021 R. P. Weinmann and B. Schmotzle (Comsecuris) using a drone and a wi-fi card attacked remotely a Tesla Model 3 by targeting with an attack



the Vcsec unit of Tesla (connected to the wi-fi car gateway, but also to the internal CAN). They were able to disable the car firewall and by using the car gateway and then using the CAN bus they were able to send messages internally in the car thus opening the car door.

- Man in the Middle attack to exploit the V2X communication between a RSU (Road Side Unit) and the 0BU (0n-Board Unit) to move the car to another position vs. the desired one and e.g. stole the car.

6.7 HW solutions to implement secure update of the SW/FW. slide 4a

6.8 Discuss the TPM specifications for HW security. Slide 1 ma penso sia 4a

6.9 Discuss the SHE specifications for HW security. slide 4a

The SHE (Secure Hardware Extensions) specification set the foundation, introducing the concept of a configurable (automotive) security subsystem.

1. The Secure Hardware Extension (SHE) is an on-chip extension to any given microcontroller.
2. SHE is intended to move the control over cryptographic keys from the software domain into the hardware domain and therefore protect those keys from software attacks.
3. SHE it is not meant to replace highly secure solutions like TPM chips (see later slide) or smart cards, i.e. no tamper resistance is required by the specification. The main goals for the SHE design are:
4. Protect cryptographic keys from software attacks
5. Provide an authentic software environment
6. Let the security depend on the strength of the underlying algorithm and the confidentiality of the keys

SHE can be connected to the CPU in several ways, e.g. through a dedicated interface or an internal peripheral bus. The interconnection must be implemented in a way that no other peripheral or an external entity can modify the data transferred between the CPU and SHE.

(Vedi immagini in slide 27 di 4a)

All cryptographic operations of SHE are processed by an AES-128.
The latency of the AES must remain ¡2 us per encryption/decryption of a single block, including the key schedule.
Minimum ECB and CBC modes of AES supported.
CMAC supported for Message Authentication Code (verification/generation).

The performance of the AES must be high enough to allow for a secure boot of 5% of the flash memory, but 32kByte at minimum and 128kByte at maximum, of the microcontroller in ¡10ms.
SHE needs memory to store secure keys and MACs.
A non-volatile memory is required to store information that needs to be available after power cycles and resets of the microcontroller.
A volatile memory is required to store temporary information. The volatile memory may loose its contents on reset or power cycles. The memory of SHE should only be accessible by the SHE control logic.
At least 100 successful write-cycles to the non-volatile memory must be guaranteed per memory slot by the implementation, more write cycles must be possible.

Key policy in the Secure Hardware Extension (SHE)

The **MASTER_ECU_KEY** is for the “owner" of the component using SHE and it can be used to reset SHE or change any of the other keys. The MASTER_ECU_KEY is only used for updating other memory slots inside of SHE

The **BOOT_MAC_KEY** is used by the secure booting mechanism to verify the authenticity of the software

The **BOOT_MAC** is used to store the MAC of the Bootloader of the secure booting mechanism and may only be accessible to the booting mechanism of SHE.

KEY_ n can be used for arbitrary functions. n is a number 3..10, i.e. SHE must at least implement three and at maximum ten keys for arbitrary use.

PRNG_SEED is used to store the seed for pseudo random number generator as described. in worst-case scenarios it is written on every power cycle/reset.

The **RAM_KEY** can be used for arbitrary operations. The PRNG_KEY and PRNG_state are used by the pseudo random number generator

R0M slots may be writable during production but not after leaving the fabrication.

Key policy in Secure Hardware Extension (SHE)

R0M slots may be writable during production but not after leaving the fabrication.

SHE must contain a unique secret key SECRET_KEY that shall not only be derived from the serial number or any other publicly available information.

The SECRET_KEY has to be inserted during chip fabrication by the semiconductor manufacturer and should not be stored outside of SHE. It can be generated by a certified physical random number generator, e.g. a Hardware Security Module (HSM).
The SECRET_KEY may only be used to import/export keys.

The UID is specified to 120 bit because it is always used in conjunction with two key ids or the status register to form a 128 bit block.
If the identification item is smaller than 120 bits it has to be padded with zero bits on the MSB side before feeding it into SHE.
The UID has to be inserted during chip fabrication by the semiconductor manufacturer

6.10 Discuss the Evita (small, medium, full) specifications for HW security. slide 4a, 5

6.11 Discuss the typical HW security units that can be found in the HSM of a chip. slide 4a e 4b

6.12 Discuss limits of current cryptographic technologies (AES, SHA, ECC, RSA) in the new post-quantum era. slide 4b

6.13 Discuss the main secure-architectural solutions adopted in the European Processor Initiative. slide 4b

6.14 What is the Panic mechanism and when is it useful? slide 4b

6.15 Side channel attacks: what are they? Ways of implementing side channel attacks? slide 4b

6.16 Discuss the HW for SIM (Subscriber Identity Module). slide 4b

6.17 Briefly review what IEC 62443 is. slide 5

6.18 Hard IP macro and soft IP macro: definition, differences in database organization, licensing costs, in protection of the IP value.

6.18.1 IP Macro

IP macro is a block of logic that can be used in a IC or SOC. It could be a CPU, cryptographic unit, USB o HDMI interface ....

6.18.2 Hard IP macro definition

Hard IP macro is a physical block implemented in our SoC already programmed, having a functional behavior defined and immutable.
Hard IP macro is optimized for a specific technology node and is not reconfigurable. Ready to be integreted in a specific chip and technology. It is a block ready to use like plug and play
Hard IP macro could be a processor or a bluetooth module ...
Hard IP macro are added directly in the layout of the chip.

- Pros: Efficient in prestazioni e consumo, testato e sicuro e sai perfettamente come funziona.
- Cons: Flessibilità pressocche assente e se cambi technology could not work anymore.

6.18.3 Soft IP macro definition

Soft IP Macro is a description od the block, structural e beahavioral description written in HDL.
In this point IP is customizable e portable on many techonologies or FPGA. This code is translated in a netlist of logic elements (logic port, flip-flp, ...) and later on phisically mapped in the layout. cryptographic modulus are Soft IP examples.

1. Write HDL;
2. Logic sysntesis → flip-flop ...
3. Place and Route → phisically mapping
4. Now it is physical circuit like Hard IP.

6.18.4 Differences in database organization

6.18.5 Differences in licensing costs

- Hard IP macro:
    1. Higher costs;
    2. Licensing payment;
    3. Licensing available for a single project or defined number of usage;
    4. Need Non Disclosure Agreement (NDA) to use it;

5. From decine to hundreds of thousands of euros;
6. High commercial value assets;
7. Without source code;

- Soft IP macro:
  1. Lower costs;
  2. The license is often perpetual or site-wide, less binding, more "open".
  3. Vendors also sell soft IP with accessible source code, so it can be easily modified and integrated.

6.18.6 Differences in protection of the IP value.

1. **Technical Value**
   (a) **Soft IP**: Flexible, portable across different devices and technologies. Quality depends on the RTL code. Moderate technical value, but highly reusable and adaptable.
   (b) **Hard IP**: Fully optimized for power, performance, and area. Not modifiable. Used for complex functions (e.g., SerDes, DDR PHY, PCIe). Very high technical value, but context-specific.
2. **Economic Value**
   (a) **Soft IP**: Lower cost, often free or included with EDA tools. Value increases with documentation and reusability. Open-source IPs may have no commercial value but are still technically useful.
   (b) **Hard IP**: Very expensive to develop or license. Can cost tens or hundreds of thousands of euros. High economic value because it reduces project risks and accelerates time-to-market.
3. **Strategic Value**
   (a) **Soft IP**: Useful for prototyping, testing, or educational purposes. Strategic if it can be modified to create proprietary derivatives.
   (b) **Hard IP**: Critical asset in commercial ASICs. Often key to a company's competitive advantage. Very high strategic value in domains like telecom, automotive, AI, etc.

6.19 Discuss if a pure synchronous design is a good solution vs side channel attacks and the possible countermeasures in HW.

6.20 Difference between a True RNG and a CSPRNG in terms of security and throughput.

6.21 What are the trusted zones in a multi-processor system on chip?

6.22 Security mechanisms for memories (MPU, HW protection in SD memory card).

6.23 Difference between a chip for security and an IP macrocell for security.

6.24 Difference between ASICs and FPGAs for security.

6.25 Discuss the acronyms COTS, SoC, MPSoC, FPGA, FPSoC and ASICs.

6.26 Difference between front-end and back-end in the chip design flow.

6.27 What's a Fabless company? Make some examples.

6.28 HW solutions to implement secure boot.

6.29 HW solutions to uniquely identify the HW.

6.30 Discuss correlation among safety and security and needs for confidentiality, integrity, authenticity, traceability (non-repudiation), availability, reliability.

6.31 Why multiple AES modes are typically available in HW?

6.32 Briefly review what UN R155, ISO 21434 and NIS2 directive are.

6.33 What is an anomaly/intrusion detection system and the difference between HW fingerprinting and rule-based IDS.

6.34 Trash, but useful The European Processor Initiative (EPI) is one of the most ambitious projects in the EU to design sovereign, high-performance, and secure processors, especially targeting HPC (High Performance Computing) and automotive markets.

Solutions:

1. **Hardware Root of Trust (HRoT)** Pensa all HRot come il blocco di bedrock di Minecraft.

   Infatti è quel componente hardware sicuro e fidato da cui vengono avviate tutte le operazioni di sicurezza (dovrebbe essere non modificabile da nessuno).

   Se compromesso, allora tutto ciò che ne consegue è a rischio.

   Spesso è un chip dedicato con le funzionalità di verificare crittograficamente il firmware UEFI prima dell'esecuzione, gestione delle chiavi sicure, integrità del sistema e protezione da malware.

2. **Secure Boot e Measured Boot**

   All'accensione del computer, l'Hardware Root of Trust (HRoT) esegue il firmware UEFI. Questo, a sua volta:

   (a) Inizializza l'hardware di base (CPU, RAM, dispositivi I/O . . .)

   (b) Avvia il bootloader, che poi carica il sistema operativo in RAM

   Prima di avviare il bootloader, entra in gioco la funzionalità di **Secure Boot**, che ha il compito di **verificare la firma digitale** dei componenti che stanno per essere eseguiti (es. il bootloader stesso). Se la firma non corrisponde a una delle chiavi fidate presenti nel firmware, l'esecuzione viene bloccata.

   Secure Boot può fallire o essere inefficace se:

   - La chiave privata è stata compromessa
   - Secure Boot è stato disattivato o modificato dall'utente
   - Il firmware UEFI stesso è stato compromesso → viene meno la Root of Trust

   In parallelo al Secure Boot viene avviato anche il **Measured Boot**, che **calcola e registra gli hash** dei componenti caricati durante la fase di avvio. Questi hash vengono salvati nel TPM (Trusted Platform Module) per poter essere **analizzati successivamente** (es. in ambiente aziendale, per rilevare modifiche sospette).

   Measured Boot non blocca l'avvio, ma fornisce una traccia affidabile dello stato del sistema al momento dell'accensione.

   HRoT → avvia firmware UEFI → Secure Boot and Measured Boot

3. **Memory Protection and Encryption**

L'architettura del sistema prevede l'utilizzo di **Memory Management Unit (MMU)** avanzate, che implementano controlli di accesso granulari per isolare i diversi processi e livelli di privilegio.

In aggiunta, è supportata la **crittografia della RAM in tempo reale**, una misura fondamentale in scenari dove è plausibile un attacco con accesso fisico al dispositivo (es. dispositivi edge, sistemi automotive, ambienti IoT).

Questa protezione contribuisce a garantire sia la **confidenzialità** che l'**integrità** dei dati in uso, contrastando attacchi come cold boot, DMA attacks, o dump della memoria.

La **Memory Management Unit (MMU)** è un componente dell'architettura della CPU che gestisce:

- la traduzione degli indirizzi virtuali in indirizzi fisici (*paging*);
- i controlli di accesso alla memoria (lettura, scrittura, esecuzione).
- impedisce a un processo in *user mode* di accedere alla memoria del *kernel*;
- impedisce a un processo di accedere alla memoria di altri processi.

Durante l'esecuzione, i dati sensibili (es. password, chiavi crittografiche) risiedono in chiaro nella RAM. Chi ha accesso fisico al dispositivo può eseguire attacchi come:

- **Cold Boot Attack**: congelamento e lettura fisica della RAM;
- **DMA Attack**: accesso diretto alla memoria tramite interfacce ad alta velocità (es. Thunderbolt, PCIe).

Per contrastare questi attacchi, alcune architetture prevedono la **crittografia automatica della RAM** in tempo reale. In questo schema:

- Il controller di memoria cifra/decripta i dati in ingresso e uscita dalla RAM;
- La chiave crittografica viene generata all'avvio tramite un *True Random Number Generator* (TRNG);
- La chiave non è mai esposta al sistema operativo né all'utente;
- Se il sistema si spegne o riavvia, la chiave va persa, rendendo i dati in RAM illeggibili.

Implementazioni reali di questa tecnologia includono:

- **AMD SME (Secure Memory Encryption)**: crittografia trasparente dell'intera RAM con una chiave hardware;
- **Intel TME (Total Memory Encryption)**: approccio analogo con chiave crittografica gestita internamente dalla CPU.

4. **Hardware Isolation & Secure Execution Environments**

5. **Secure Interconnects and I/O Protection**
6. **Support for Post-Quantum Cryptography (PQC)**
7. **Side-Channel Attack Mitigation**
8. **Secure Debug and Update Mechanisms**

7 Nannipieri

7.1 Cos' è VLSI? VLSI (*Very Large Scale Integration*) è una tecnologia che consente di integrare milioni (e oggi anche miliardi) di componenti elettronici, principalmente transistor, all'interno di un singolo chip di silicio. Questa tecnologia è alla base della progettazione dei circuiti integrati (IC) moderni, come microprocessori, memorie e ASIC[1].

Il flusso di progettazione VLSI (design flow) è un processo ingegneristico ben definito e sequenziale che trasforma una specifica funzionale in un layout fisico fabbricabile. Questo flusso è suddiviso in tre fasi principali:

1. RTL Design
2. Logic Design
3. Physical Design

7.2 1. RTL Design L'acronimo RTL sta per *Register Transfer Level*, ovvero un livello di astrazione che descrive il comportamento di un circuito digitale in termini di trasferimenti di dati tra registri e delle operazioni logiche o aritmetiche che avvengono su tali dati.

In questa fase, l'ingegnere descrive l'architettura funzionale del sistema utilizzando linguaggi di descrizione hardware (HDL), come Verilog o VHDL. L'output di questa fase è un modello RTL del circuito, che può essere simulato per verificarne il comportamento logico.

Le attività principali dell'RTL Design includono:

- Scrittura del codice HDL secondo le specifiche funzionali del progetto.
- Simulazione funzionale per verificare la correttezza del comportamento logico.
- Verifica formale, che può includere il controllo delle proprietà del sistema (model checking).
- Sintesi RTL, che traduce il codice HDL in una rete logica costituita da porte logiche standard.

A questo livello non si ha ancora alcuna informazione sulla temporizzazione reale del circuito, né sulla sua implementazione fisica: tutto è ancora astratto.

Prima di procedere con la descrizione RTL, però, vengono effettuati ulteriori step preliminari fondamentali, che gettano le basi strutturali e funzionali del progetto:

1. **Architectural Design**: questa fase consiste nella definizione dell'architettura generale del sistema. Viene stabilita la struttura ad alto livello, compresi i blocchi funzionali principali (ALU, registri, controller,

[1] Application-Specific Integrated Circuit: circuito integrato progettato per un'applicazione specifica, a differenza di un circuito generico o programmabile come un FPGA.

interfacce, memorie, ecc.), la loro organizzazione e le modalità di interazione. Si decide anche se utilizzare un'architettura pipelined, Harvard, Von Neumann, ecc. In altre parole, si costruisce lo *scheletro* logico del chip. Il risultato è un diagramma architetturale e una specifica tecnica.

2. **Functional Design**: in questa fase si passa dalla descrizione architetturale alla definizione dettagliata delle funzionalità di ciascun modulo. Si stabiliscono i comportamenti desiderati a livello funzionale per ogni componente, identificando ingressi, uscite e modalità di funzionamento. Questo è spesso rappresentato tramite diagrammi a stati finiti (FSM[2]) o pseudocodice. Lo scopo è garantire che tutti i requisiti funzionali siano stati interpretati correttamente prima di procedere alla codifica HDL.

7.2.1 RTL Simulation

La **simulazione RTL** è il primo test concreto del design, effettuato a partire dal codice HDL scritto durante la fase RTL. Viene utilizzata per verificare che il comportamento funzionale del circuito rispecchi esattamente quanto definito nelle specifiche funzionali e nella progettazione architetturale.

Gli strumenti di simulazione (come ModelSim, QuestaSim o VCS) eseguono il codice HDL e permettono di osservare il comportamento dei segnali in ingresso e in uscita nel tempo.

Le attività principali includono:

- Scrittura di *testbench*, ovvero moduli HDL che generano segnali di test per stimolare il circuito.
- Analisi delle waveform risultanti per identificare eventuali discrepanze con il comportamento atteso.
- Debug del codice RTL in caso di errori logici o malfunzionamenti.

Questa fase è cruciale per identificare e correggere bug logici prima di passare alla sintesi, poiché successivi errori saranno più costosi da correggere.

7.3 2. Logic Design La fase di Logic Design, o *Logica Gate-level*, rappresenta il passaggio intermedio tra la descrizione astratta (RTL) e l'implementazione fisica.

Dopo la sintesi RTL, si ottiene una netlist[3] composta da primitive logiche appartenenti a una specifica libreria di celle standard (*standard cell library*).

Le attività fondamentali in questa fase includono:

- Ottimizzazione logica per area, potenza e temporizzazione (delay).

[2]Finite State Machine: modello computazionale che descrive un comportamento mediante un numero finito di stati e transizioni.

[3]Una *netlist* è una rappresentazione del circuito sotto forma di lista di componenti logici (porte AND, OR, flip-flop, ecc.) e delle loro interconnessioni. È un'astrazione più vicina all'hardware reale rispetto al codice RTL.

- Timing Analysis statica (STA)[4].
- Equivalence Checking per garantire che la netlist risultante sia funzionalmente equivalente al design RTL originale.

In questa fase si comincia a considerare il concetto di *clock cycle* e i vincoli di temporizzazione, per preparare il circuito all'implementazione fisica.

Il circuito viene descritto in termini di:

1. Porte logiche (AND, OR, NOT, XOR, flip-flop, ecc.)
2. Interconnessioni tra le varie porte logiche, in base alla netlist generata

Input: file HDL contenenti la descrizione RTL e la libreria tecnologica del processo produttivo.
Output: una *netlist*, ovvero una descrizione del circuito in termini di porte logiche e connessioni fisiche.

Questa traduzione dalla descrizione RTL alla netlist è detta **sintesi logica** (*logic synthesis*).

I parametri principali che influenzano la sintesi logica sono:

- **Temporizzazione** (Timing): il tempo impiegato dai segnali per propagarsi attraverso i componenti.
- **Area**: la superficie fisica del chip occupata dal circuito.
- **Potenza** (Power): il consumo energetico complessivo, sia statico che dinamico.

Alcune considerazioni fisiche importanti:

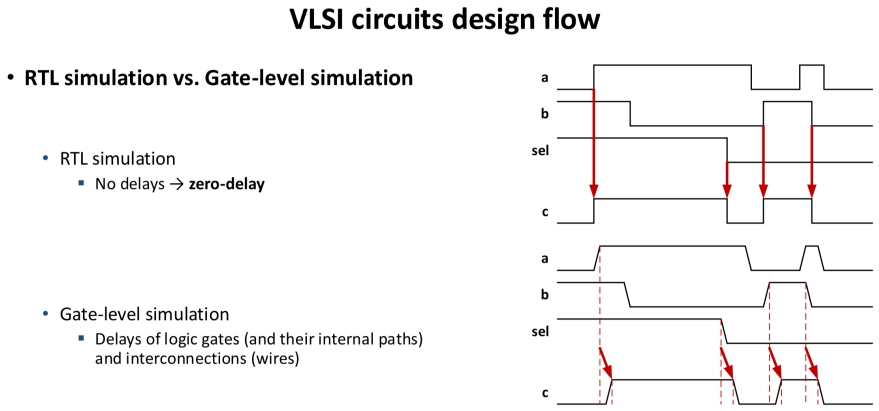
- Allargando i transistor, aumenta la corrente condotta, si riduce il ritardo e cresce la frequenza operativa.
- Tuttavia, transistor più larghi consumano più energia, portando a un compromesso tra prestazioni e efficienza energetica.

Durante questa fase vengono eseguite simulazioni di tipo **gate-level**, le quali:

- Verificano la *chiusura temporale* (*timing closure*), assicurando che i percorsi critici rispettino i vincoli di clock.
- Stimano il consumo di potenza totale e per porzione di circuito.
- Possono riutilizzare i *testbench* impiegati nella simulazione RTL.
- Includono ritardi realistici (a differenza delle simulazioni RTL, che assumono ritardi nulli).

[4]La Static Timing Analysis è una tecnica per valutare la temporizzazione di un circuito senza effettuare simulazioni dinamiche, individuando i percorsi critici e verificando che i segnali rispettino le soglie di tempo definite.

- Permettono di modificare la netlist in caso di problemi (es. frequenza troppo bassa).
- Possono, se necessario, riportare il progetto alla fase RTL per una revisione più strutturale.



7.4 3. Physical Design Il Physical Design è la fase in cui il circuito digitale viene tradotto in una rappresentazione geometrica fisica, pronta per la fabbricazione su silicio. È qui che la teoria incontra il mondo fisico.

Le attività principali includono:

- **Floorplanning**: suddivisione dell'area del chip in regioni funzionali, assegnazione delle macro e dei blocchi di memoria.
- **Placement**: posizionamento delle celle logiche standard all'interno del floorplan, ottimizzando area e temporizzazione.
- **Clock Tree Synthesis (CTS)**: progettazione della rete di distribuzione del clock in modo da minimizzare skew[5] e ritardi.
- **Routing**: connessione fisica tra le celle secondo la netlist. Avviene tramite strati metallici sovrapposti.
- **DRC/LVS**: verifica del layout rispetto alle regole di fabbricazione (*Design Rule Checking*) e confronto con la netlist (*Layout Versus Schematic*).
- **Extraction e Post-layout Simulation**: calcolo delle effettive capacità e resistenze presenti nel layout per una simulazione più realistica delle prestazioni.

Al termine del Physical Design, si genera un file chiamato GDSII o OASIS[6], che rappresenta il layout finale del circuito e viene inviato alla fonderia per la fabbricazione.

Input: netlist generata dalla fase di logic design, accompagnata dalla libreria tecnologica.
Output: layout fisico, cioè una descrizione geometrica del circuito (forma, posizione, orientamento dei transistor e metallizzazioni).

Questo processo di traduzione dalla netlist al layout è detto **sintesi fisica** (*physical synthesis*).

[5]Lo *skew* è la differenza di tempo con cui il segnale di clock raggiunge componenti differenti del circuito. Un eccessivo skew può compromettere la correttezza del funzionamento sincrono.

[6]GDSII (Graphic Data System II) e OASIS sono formati standard di output utilizzati per fornire il layout fisico alle fonderie per la produzione dei chip.

Le fasi principali della sintesi fisica comprendono:

- **Place**: posizionamento e orientamento ottimale delle celle logiche per migliorare area, temporizzazione e potenza.
- **Route**: creazione delle connessioni fisiche tra le celle tramite strati metallici (routing multilivello).
- **Clock Tree Synthesis (CTS)**: creazione di una rete di distribuzione del segnale di clock con ritardi controllati e minimizzazione dello *skew*.

Le simulazioni svolte in questa fase sono dette **a livello transistor** (*transistor-level simulations*), e servono a modellare in modo accurato:

- La temporizzazione reale (inclusi i parassiti RC)
- Il consumo di potenza dinamico e statico
- Le prestazioni del circuito post-layout

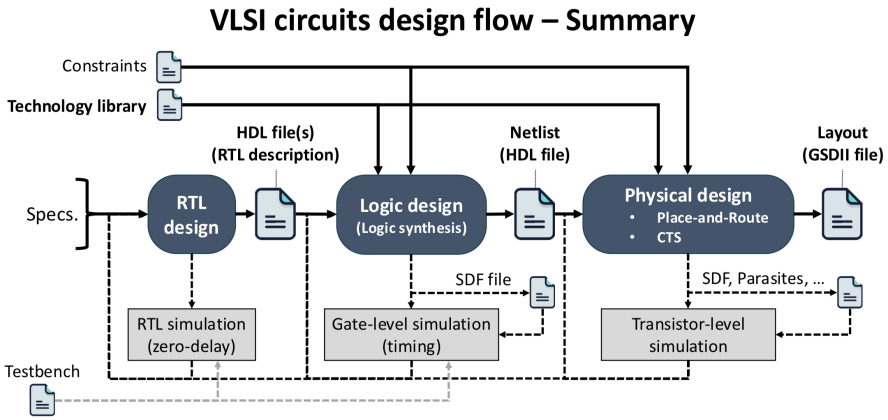
Anche in questo stadio è possibile:

- Aggiustare il layout se emergono problemi (es. frequenza non soddisfacente).
- Tornare alla fase di logic design o persino alla RTL se serve una modifica più profonda.

Questa fase precede le attività finali del ciclo di vita del chip:

1. **Fabbricazione** (Fabrication): invio del file GDSII/OASIS alla fonderia per la produzione fisica del chip.
2. **Packaging e Testing**: incapsulamento del chip e collaudo con strumenti ATE[7].

[7]Automatic Test Equipment: macchinari automatici per il test elettrico dei chip.



Infine, ricordiamo la distinzione tra:

- **Front-end**: tutte le fasi che precedono la sintesi logica; sono indipendenti dalla tecnologia di fabbricazione.
- **Back-end**: tutte le fasi successive, che dipendono da parametri fisici e tecnologici specifici della fonderia.

7.5 FPGA Un **FPGA** (Field Programmable Gate Array) è un circuito integrato digitale programmabile dall'utente dopo la produzione. A differenza degli ASIC, gli FPGA non sono progettati per una funzione specifica, ma contengono una rete di risorse logiche riconfigurabili (come LUT[8], flip-flop, blocchi di memoria, moltiplicatori, ecc.) e un'infrastruttura di interconnessione programmabile.

Grazie a questa flessibilità, gli FPGA possono implementare una vasta gamma di circuiti digitali, da semplici combinatori a sistemi embedded complessi.

I principali vantaggi degli FPGA includono:

- Riconfigurabilità: possono essere riprogrammati più volte, anche dopo la distribuzione.
- Tempo di sviluppo ridotto: adatti a prototipazione rapida e testing.
- Basso costo iniziale: non richiedono fasi di fabbricazione personalizzate.
- Ampio supporto tool: strumenti come Vivado, Quartus e ModelSim permettono la progettazione, simulazione e programmazione.

Tuttavia, gli FPGA presentano anche alcune limitazioni:

- Prestazioni inferiori rispetto agli ASIC, soprattutto in frequenza massima e area occupata.
- Consumo energetico relativamente più elevato.
- Costi unitari più alti se il volume di produzione è elevato.

Gli FPGA trovano applicazione in:

[8]Look-Up Table: unità logica programmabile che implementa funzioni booleane

- Prototipazione di circuiti ASIC.
- Accelerazione hardware (ad es. reti neurali, codifica video, crittografia).
- Controllo industriale e automazione.
- Telecomunicazioni e SDR (Software Defined Radio).
- Sistemi embedded critici (es. settore aerospaziale, automotive).

7.5.1 FPGA Architecture

L'architettura di un FPGA è organizzata in modo modulare e riconfigurabile. Un dispositivo FPGA è composto principalmente da tre elementi fondamentali:

1. **Blocchi logici configurabili (CLB - Configurable Logic Blocks)**: sono i blocchi base che realizzano le funzioni logiche. Ogni CLB contiene:
   - **LUT** (Look-Up Table)[9]
   - **Flip-Flop** o latch per la memorizzazione dei valori (logica sequenziale)
   - Multiplexer per la selezione delle uscite
2. **Matrice di interconnessione (Routing Matrix)**: rete di collegamenti programmabili che consente la connessione tra i CLB, tra i CLB e le I/O, o verso altri blocchi specializzati. L'interconnessione è il cuore della flessibilità di un FPGA.
3. **Blocchi I/O (Input/Output)**: interfacce programmabili che permettono di collegare l'FPGA al mondo esterno (segnali digitali, bus di sistema, clock, ecc.). Supportano differenti standard logici (LVTTL, LVCMOS, ecc.).

Oltre ai blocchi fondamentali, gli FPGA moderni integrano risorse avanzate:

- **Blocchi DSP**: moltiplicatori e accumulatori ottimizzati per elaborazione numerica ad alte prestazioni.
- **Blocchi RAM**: memoria distribuita e blocchi RAM (es. BRAM) utilizzabili internamente dal circuito.
- **Clock management units**: PLL[10], MMCM, e reti di distribuzione del clock.
- **Controller embedded**: alcuni FPGA includono core ARM o RISC-V (System-on-Chip FPGA).
- **Interfacce ad alta velocità**: transceiver SERDES, PCIe, DDR controller.

L'architettura fisica è altamente ripetitiva e scalabile, rendendo possibile l'integrazione di migliaia di CLB in FPGA di grandi dimensioni.

[9] Una LUT è una piccola memoria che implementa una funzione booleana arbitraria precaricata.

[10] Phase-Locked Loop: circuito per generare clock interni sincronizzati a frequenze diverse.

7.5.2 FPGA Design Flow

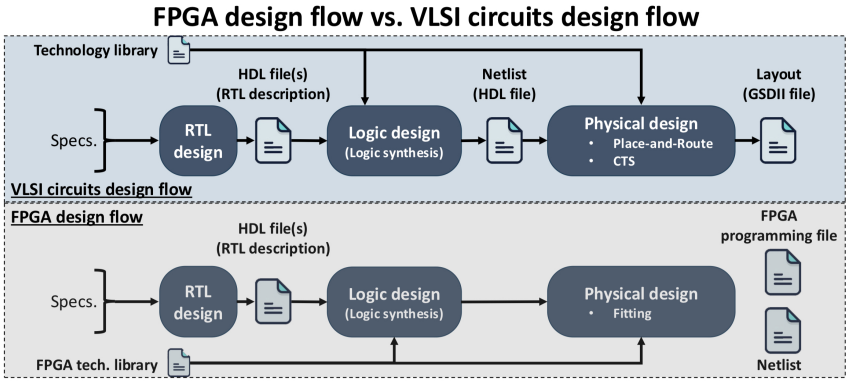
Il **flow di progettazione per FPGA** (Field Programmable Gate Array) è una sequenza di passi che permette di implementare un circuito digitale all'interno di un dispositivo riconfigurabile. A differenza del design flow VLSI, non si arriva alla fabbricazione di un chip, ma alla configurazione di un chip già prodotto.

I principali step del flow FPGA sono:

1. **Specifiche**: definizione del comportamento funzionale del sistema da implementare.
2. **RTL Design**: scrittura del codice HDL (tipicamente VHDL o Verilog).
3. **RTL Simulation**: verifica funzionale tramite testbench.
4. **Synthesis**: traduzione del codice RTL in una netlist di primitive logiche (porte, flip-flop, LUT, ecc.).
5. **Implementation (Place and Route)**: posizionamento delle celle logiche sulle risorse fisiche dell'FPGA e instradamento dei segnali.
6. **Timing Analysis**: verifica che il circuito rispetti i vincoli di temporizzazione.
7. **Bitstream Generation**: generazione del file binario che verrà caricato sull'FPGA.
8. **Programming**: caricamento del bitstream nel dispositivo FPGA per il test fisico.

7.5.3 Differenza tra FPGA design flow e VLSI design flow

- Il flow **FPGA termina con la programmazione di un dispositivo preesistente**, mentre il flow **VLSI termina con la fabbricazione fisica del chip** (ASIC).
- La **sintesi logica** nel flow FPGA produce una netlist ottimizzata per una specifica architettura FPGA (es. Xilinx, Intel), mentre nel VLSI si basa su una libreria di celle standard fisiche.
- Il flow FPGA non ha **sintesi fisica** vera e propria, perché il layout è già definito nel chip.
- Le **simulazioni post-layout** negli FPGA modellano solo il comportamento della configurazione logica, non dei transistor.
- I vincoli di potenza e area sono meno stringenti negli FPGA, ma la frequenza massima è generalmente inferiore rispetto agli ASIC.



7.6 ASICs Gli **ASIC** (Application-Specific Integrated Circuits) sono chip progettati su misura per una singola applicazione o funzione. Offrono massime prestazioni, efficienza energetica e densità, ma sono costosi da produrre e modificare.

Vantaggi degli ASIC:

- Maggiore velocità e performance rispetto agli FPGA.
- Minore consumo energetico grazie all'ottimizzazione fisica.
- Minore area occupata e maggiore densità logica.
- Protezione della proprietà intellettuale più robusta.

Svantaggi:

- Costi di NRE (Non Recurring Engineering) molto elevati.
- Tempi di sviluppo e test più lunghi.
- Una volta fabbricato, il circuito non è più modificabile.

7.7 Differenza tra FPGA e ASICs

- **Flessibilità**: FPGA è riconfigurabile, ASIC è fisso.
- **Prestazioni**: ASIC ha prestazioni superiori.
- **Consumo**: ASIC consuma meno.
- **Costo unitario**: FPGA costa meno per piccoli volumi, ASIC conviene per grandi volumi.
- **Tempo di sviluppo**: FPGA è molto più rapido da progettare e modificare.
- La fase di fabbricazione non è richiesta (il dispositivo/circuito è fabbricato... si tratta dell'FPGA).
- La fase di confezionamento e collaudo può essere ancora necessaria se si sta integrando l'FPGA su una scheda con altri componenti ...

	FPGA	ASIC
Time to Market	Fast	Slow
Unit cost (RE, Recurring Expense)	High	Low
Non-Recurring Expense (NRE)	Low	High
Design flow	Simple	Complex
Performance	Medium	High
Power Consumption	High	Low
Unit size (area)	Medium	Low

Quando FPGA conviene rispetto a ASICs? Un FPGA è preferibile quando:

- Il tempo di immissione sul mercato (*time-to-market*) è critico.
- Si è ancora in fase di sviluppo o prototipazione del prodotto.
- Il numero di unità da produrre è basso o medio.
- Sono richieste funzionalità aggiornabili o riconfigurabili nel tempo (es. aggiornamenti firmware).
- Si vogliono evitare i costi di produzione di un ASIC per applicazioni temporanee o di nicchia.

7.8 Design by yourself RTL design, Testbench possono essere scriti con linguaggi di alto livello come python.
RTL Simulation in Modelsim.
Logic e Physical design in Quartus prime.

7.8.1 HDL e SystemVerilog

Un **HDL** (Hardware Description Language) è un linguaggio di descrizione hardware utilizzato per modellare circuiti digitali in modo formale e simulabile. A differenza dei linguaggi di programmazione tradizionali (come C o Python), un HDL non descrive una sequenza di istruzioni, ma il *comportamento hardware* di un circuito — cioè cosa fa ogni parte del circuito e come le varie parti sono collegate.

I due linguaggi HDL più diffusi sono:

- **VHDL** (VHSIC Hardware Description Language)
- **Verilog** (e la sua evoluzione moderna: **SystemVerilog**)

A cosa serve un HDL? Un HDL permette di:

- Descrivere il comportamento di circuiti digitali, sia combinatori (senza clock) che sequenziali (sincronizzati con un clock).

- Simulare il circuito prima della sintesi, per verificarne il funzionamento.
- Effettuare sintesi automatica in netlist gate-level.
- Programmare FPGA o progettare ASIC.

SystemVerilog: cos'è e perché si usa? **SystemVerilog** è un'estensione del linguaggio Verilog, progettata per migliorare le capacità di modellazione e verifica. Include funzionalità avanzate sia per la descrizione RTL sia per la *verification* (cioè la fase di test formale e simulativo).

SystemVerilog unisce tre mondi:

1. Descrizione hardware (come Verilog tradizionale)
2. Costrutti orientati agli oggetti per la verifica (OOP simile a C++)
3. Funzionalità avanzate per la definizione di interfacce e moduli complessi

Perché è importante? SystemVerilog è oggi lo standard de facto nell'industria per la progettazione di sistemi digitali complessi, specialmente per:

- la progettazione di microprocessori, DSP, controller, core IP
- l'uso con FPGA e ASIC
- l'automazione della verifica grazie a tool come UVM (Universal Verification Methodology)

Un file in SystemVerilog ha estensione .sv, mentre un file in VHDL ha estensione .vhd.
Un circuito descritto in SystemVerilog deve essere incapsulato in moduli, quindi definisci:

1. Nome del modulo
2. Input e output
3. Comportamento e funzione del circuito

Di seguito un semplice modulo combinatorio in SystemVerilog che implementa una porta AND:

```
verilog
module and_gate (
    input logic a,
    input logic b,
    output logic y
);

    assign y = a & b;

endmodule
```

Sintassi:

Elemento	Descrizione	Esempio
`<polarity>`	Direzione della porta: `input`, `output`, `inout`	`input`
`[<type>]`	Tipo del segnale (opzionale, consigliato: `logic`)	`logic`
`[<bit width>]`	Larghezza del bus (opzionale, in caso di vettori)	`[7:0]`
`<port name>`	Nome identificativo della porta	`data`, `clk`

Table 1: Sintassi generale per la dichiarazione di una porta in SystemVerilog

7.8.2 Modularità e gerarchia nei circuiti digitali

Un circuito digitale complesso può (e dovrebbe) essere suddiviso in sottounità più semplici, dette **moduli**. Questo approccio, detto *modular design*, è centrale nella progettazione HDL.

Vantaggi

- **Gestione semplificata** dei progetti complessi tramite il principio "divide et impera".
- **Riutilizzabilità** dei moduli in più progetti senza doverli riscrivere.
- **Verifica indipendente**: ogni modulo può essere simulato e testato da solo.
- **Scalabilità** del progetto grazie a una struttura gerarchica.

Progettazione gerarchica L'approccio modulare può essere applicato in modo ricorsivo:

- Un modulo può contenere al suo interno altri moduli (*istanziamento*).
- Si possono avere **più livelli gerarchici**.
- Ogni modulo può risiedere in un file separato (*best practice*) oppure essere scritto nello stesso file.

Il modulo di più alto livello, che:

- Istanzia tutti gli altri moduli;
- Definisce le porte di input/output verso l'esterno;
- Non viene mai istanziato da altri moduli;

è detto **Top-Level Module**. È il punto di partenza del progetto.

7.8.3 Testbench e verifica in SystemVerilog

I linguaggi HDL, incluso SystemVerilog, non solo permettono di descrivere hardware, ma offrono anche strumenti avanzati per il **testing** e la **verifica** del circuito. Questo avviene principalmente tramite l'uso di un **testbench**.

Concetto	Descrizione
`Modulo`	Blocco funzionale indipendente che implementa una parte del circuito
`Istanziamento`	Utilizzo di un modulo all'interno di un altro
`Gerarchia`	Struttura multilivello formata da moduli e sottoblocchi
`Top-Level`	Modulo principale che racchiude l'intero progetto

Table 2: Concetti chiave nella progettazione gerarchica di circuiti digitali

Cos'è un testbench? Un **testbench** è un modulo HDL speciale che:

- Non ha porte di input/output.
- Contiene un'istanza del circuito da testare (chiamato **DUT – Device Under Test**).
- Genera segnali di stimolo per testare il comportamento del DUT.
- Verifica che l'output del DUT sia corretto.

Struttura generale di un testbench

- Dichiarazione dei segnali da collegare al DUT.
- Istanza del DUT (tipicamente il top-level del progetto).
- Stimoli: comandi per cambiare i valori dei segnali nel tempo.
- Controlli/verifiche: confronti tra l'output del DUT e il risultato atteso.

Terminologia:

- **DUT (Device Under Test)**: modulo da testare.
- **Stimoli (stimuli)**: valori applicati agli ingressi del DUT.
- **Check/assertion**: verifica che l'uscita del DUT sia quella attesa.

Vantaggi dei testbench

- Permettono di testare circuiti complessi prima della sintesi fisica.
- Rendono il debug e la verifica automatizzabili.
- Possono essere riutilizzati per la regressione (test multipli su più versioni del design).

Esempio base di testbench

```
module and_tb;

    logic a, b;
    logic y;

    // Istanza del DUT
    and_gate dut (
        .a(a),
        .b(b),
        .y(y)
    );

    initial begin
        a = 0; b = 0;
        #10;
        a = 1; b = 0;
        #10;
        a = 0; b = 1;
        #10;
        a = 1; b = 1;
        #10;
    end

endmodule
```

Verifica e bug: separare i team Nelle aziende strutturate, solitamente il **team di design** è separato dal **team di verifica**, per evitare errori sistemici:

- Se lo stesso sviluppatore scrive sia il modulo che il testbench, potrebbe *ripetere lo stesso errore* in entrambi.
- Questo renderebbe il bug *invisibile* alla simulazione.

Approcci alla verifica Due tecniche principali vengono impiegate:

1. **Reference test vectors**: una serie di ingressi e uscite attesi viene usata per confrontare il comportamento del circuito.
2. **Reference model**: un modello software o funzionale del sistema fornisce l'output corretto, da confrontare con quello generato dal circuito.

Tabella riassuntiva dei concetti chiave

Concetto	**Descrizione**
`Testbench`	Modulo HDL che stimola e verifica il circuito
`DUT`	Device Under Test: modulo da testare
`Stimuli`	Segnali applicati agli ingressi del DUT
`Check`	Verifica che l'uscita del DUT sia corretta
`Reference model`	Modello teorico usato per confronto automatico
`Reference vectors`	Serie di ingressi e uscite attese per test diretti

Table 3: Componenti fondamentali nella verifica HDL