Introduction à la cryptographie embarquée

Master Securité et Systèmes Embarqués (M2SE)
2022

Boly SECK
A partir du cours de Alexandre Venelli

Produits embarqués sécurisés

Types de produits embarqués sécurisés

- Cartes à puce
  - Contact
  - Sans contact
  - Dual interface
- Tag RFID
- NFC
- Tokens
  - USB
- Set top box
- Documents officiels:
  - Carte d'identité
  - Passeport
  - Permis de conduire

Historique de la carte à puce

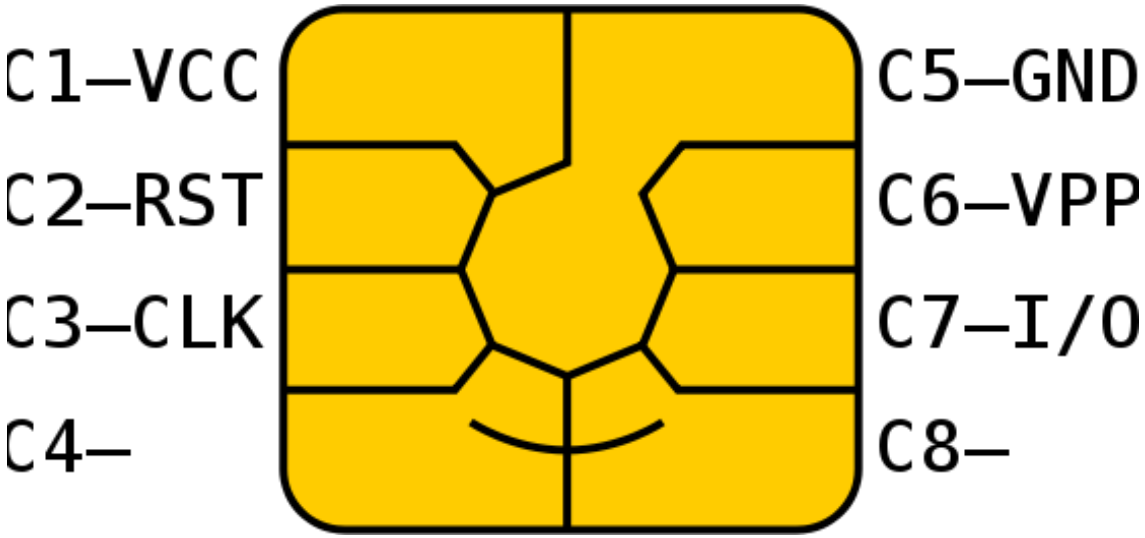
- 1974-1975: R. Moreno brevette une carte pouvant contenir beaucoup de données
  - Inventeur de la carte à puce
- 1977: M. Ugon ajoute
  - Une NVM
  - Un microprocesseur
- 1984: premières cartes bancaires
- 1984: premières cartes téléphoniques
- 1991: invention de la carte SIM
- 1998: cartes Vitale
- 1998-1999: boom de la carte SIM
  - Cartes les plus vendues encore aujourd'hui
- Dans plus en plus de produits depuis:
  - Cartes de fidélité, Pay-TV, …

Qu’est-ce qu’une carte à puce?

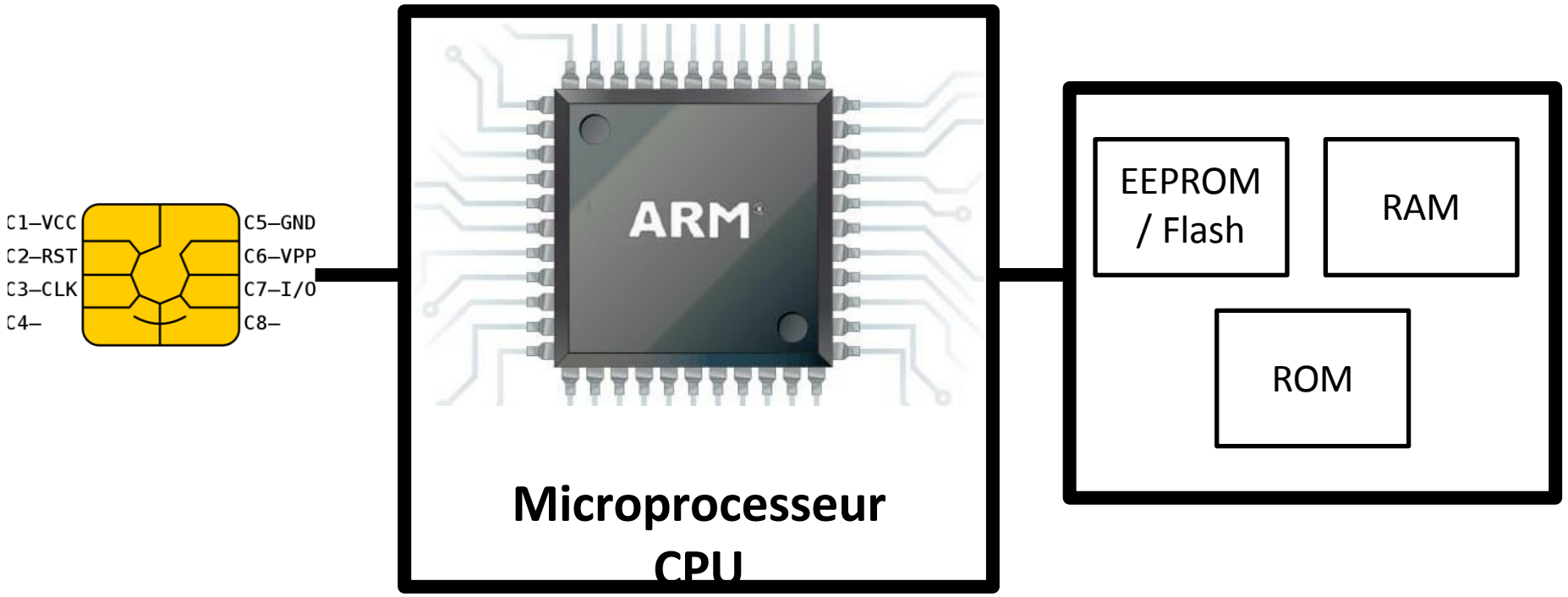
- Un morceaux de silicone sur une carte plastique
- Très bon moyen de stocker peu de données sensibles

Carte à puce contact

- VCC: tension d'alimentation de la carte
- RST: signal de remise à zéro
- CLK: signal d'horloge
- GND: masse électrique
- VPP: tension de programmation
- I/O: entrées/sorties des données

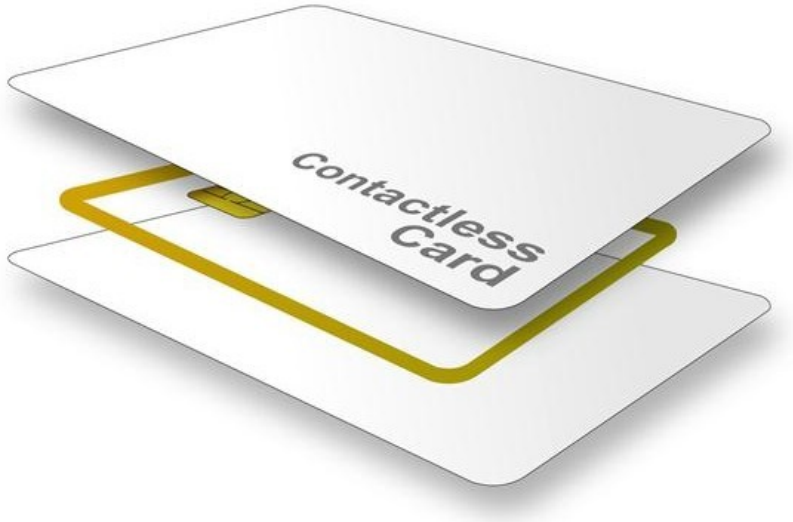


Carte à puce contact



Carte sans contact

- Caractéristiques proches d'une carte contact sans connexion électrique
- Energie fournie par le champ magnétique émis par le lecteur



Carte à puce duale

- Possède à la fois les technologies contact et sans contact
- Remplace peu à peu les cartes contact

Tags RFID

- Plus simple que les cartes à puces
- Peut ne pas avoir de microprocesseur
- Coût très réduit
- Exemples d'applications:
  - Magasinage, transport de colis, suivis dans les hôpitaux, etc.

NFC

- Near Field Communication
- Peut fonctionner en 3 modes
  - Lecteur
    - Agit comme un lecteur sans contact et peut parler avec une carte sans contact
  - Carte
    - Se comporte comme une carte sans contact et peut parler avec un lecteur
  - Peer-to-peer
    - Effectue un lien entre 2 participants (ex: téléphones).

Acteurs du marché

Cryptographie embarquée

Pourquoi ajouter de la sécurité?

- Vol de services
  - Attaques sur des fournisseurs de services (TV satellite, cartes d'accès, …)
- Accès à l'information
  - Récupération et extraction d'informations
  - Récupérer des secrets commerciaux
  - Vol d'identité
- Clonage et surproduction
  - Copier pour faire des profits sans coûts de R&D
  - Production en masse à faible coût par des sous-traitants

Qui a besoin de puces sécurisées?

- Demande de plus en plus importante
  - Industrie automobile, fournisseurs de services
  - Banques, applications militaires
- Puces sécurisées sont partout
  - Électronique grand consommation (authentification, protection contre la copie)
  - Contrôle du marché de l'après-vente (accessoires, pièces de rechange)
  - Contrôle d'accès (tags RF, cartes)
  - Contrôle de services (téléphones portables, TV satellite)
  - Protection de la propriété intellectuelle (IP) (sofware, algorithmes)
- Challenges
  - Comment concevoir une puce sécurisée? (ingénierie sécurité/crypto)
  - Comment évaluer sa résistance? (estimation du coût d'une faiblesse)
  - Comment trouver la meilleur solution? (compromis temps/argent)

Comment concevoir un système sécurisé?

- Pour quelles raisons attaquerait-on votre système?
  - Scénarios d'attaques et enjeux (vol, accès, etc)
- Qui l'attaquerait?
  - Classe de l'attaquant: extérieur, initié, organisation
- Quels outils utiliseraient-ils pour attaquer?
  - Catégories d'attaques: side-channel, faute, reverse engineering
- Comment se protéger de ces attaques?
  - Estimer le danger: comprendre les enjeux, le coût, la probabilité
  - Développer des protections adéquates aux point faibles
  - Faire une évaluation de sécurité
  - Choisir des composants sûrs pour construire le système → cryptographie forte

Cryptographie

- La cryptographie permet d'assurer
  - Intégrité
    - Un attaquant ne peut pas modifier un message
  - Confidentialité
    - Un attaquant ne doit pas pouvoir lire des messages
  - Authenticité
    - Un attaquant ne doit pas pouvoir se faire passer pour quelqu'un d'autre
  - Non-répudiation
    - Un attaquant ne doit pas pouvoir dénier avoir fait un échange

Principe de Kerkhoff

- Les algorithmes de cryptographie modernes sont basés sur le principe de Kerkhoff:
  - Un cryptosystème n'a pas à être secret et doit pouvoir tomber dans les mains d'un attaquant sans poser de problème
- La sécurité des cryptosystèmes est basée sur le fait que la clé soit gardée secrète!
- Quelques raisons pour ce principe:
  - Il est plus facile de garder une clé secrète qu'un algorithme
  - Les clés peuvent être changées plus facilement que les algorithmes
  - Communication avec d'autres plus facile
  - Les algorithmes publics subissent un examen public détaillé

Cryptographie – Primitives et cryptosystèmes

- Cryptosystèmes symétriques:
  - Chiffrements par blocs
    - DES, AES, …
  - Chiffrement par flot
    - RC4, A5/1, A5/2, …
- Cryptosystèmes asymétriques:
  - RSA, Diffie-Hellman, ECC, …
- Fonctions de hash
  - MD5, SHA-1, SHA-2, SHA-3, …

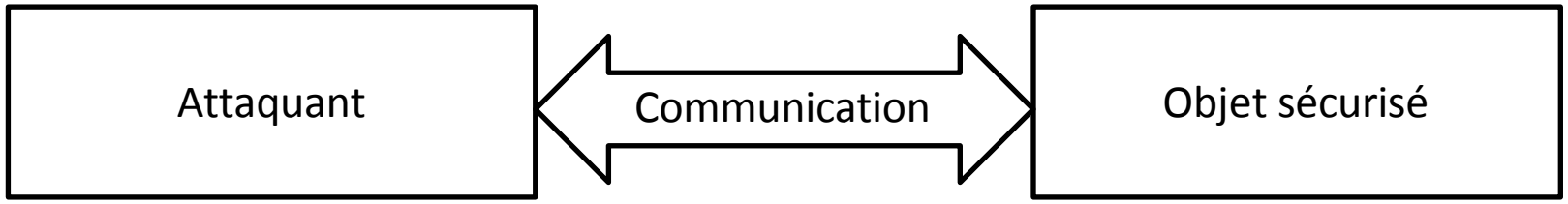
Attaques logiques et physiques

Scénarios classiques d'attaques

- Chiffré seul
  - L'attaquant n'a accès qu'au chiffré
- Message connu
  - L'attaquant a accès à un ensemble de chiffrés avec les messages correspondants
- Chiffré choisi
  - L'attaquant peut obtenir des messages pour un ensemble arbitraire de chiffrés choisis
- Chiffré choisi adaptatif
  - L'attaquant peut choisir des chiffrés en fonction des informations qu'il a obtenu des chiffrements précédents
- Attaque par clé apparentée
  - L'attaquant peut obtenir des chiffrés pour des messages choisis chiffrés avec deux clés différentes. Les clés sont inconnues mais l'attaquant connait la relation qui lie leurs valeurs.
- …

Attaquer un système embarqué sécurisé – attaque logique

- Les clés stockées et utilisées dans les objets sécurisés ne doivent jamais quitter l'objet
- Le but des attaques logiques est de révéler la clé en utilisant seulement l'interface de communication



Attaques logiques

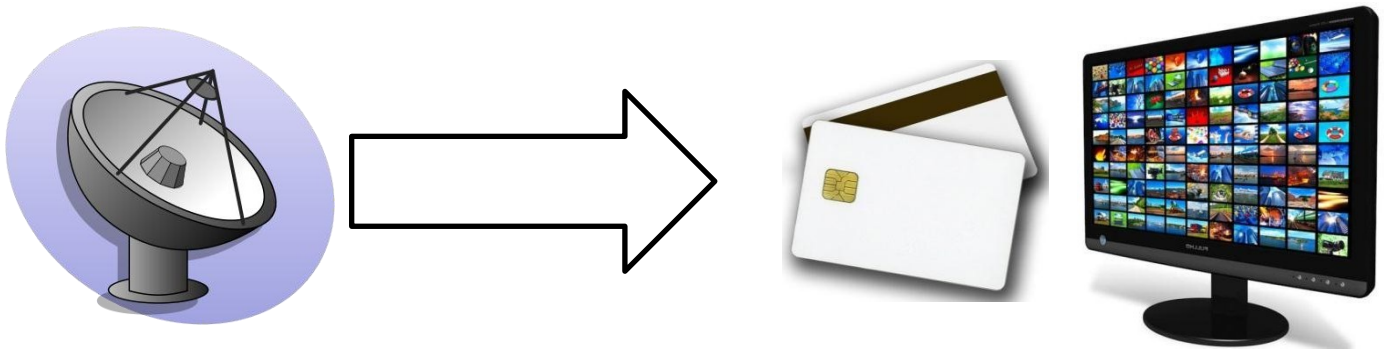
- Faiblesses basées sur des vulnérabilités software
- L'attaquant essaye de casser le système en envoyant des messages spéciaux
- Attaques typiques: buffer overflow, format string attack, ...

Autre type d'attaque?

- Les attaques logiques considèrent que l'attaquant a le contrôle complet sur l'interface de communication
- Il existe de nombreux scénarios où ces attaques sont les plus importantes (ex: communication via internet)
- Pour les objets embarqués, la situation est différente. L'attaquant a souvent accès à beaucoup plus que seulement l'interface de communication. Il possède physiquement l'objet en question.

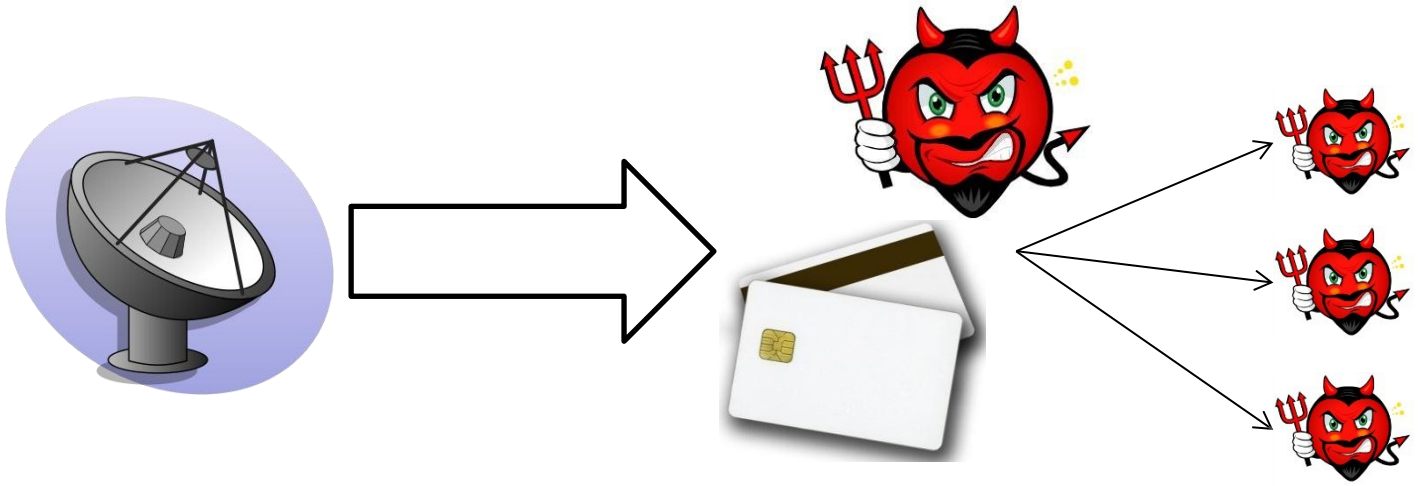
Exemple: Pay TV

- Les entreprises de pay TV donnent des cartes à puces à leurs clients qui payent pour voir un certain contenu.



Exemple: Pay TV

- La compagnie ne peut pas supposer que tous ses client sont honnêtes. Un client peut vouloir dupliquer sa carte pour ses amis

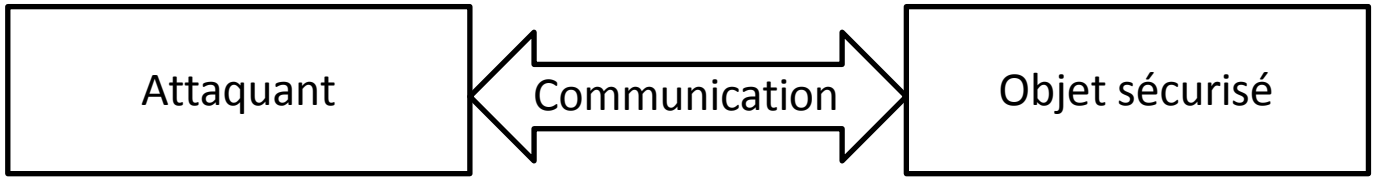


Autres exemples

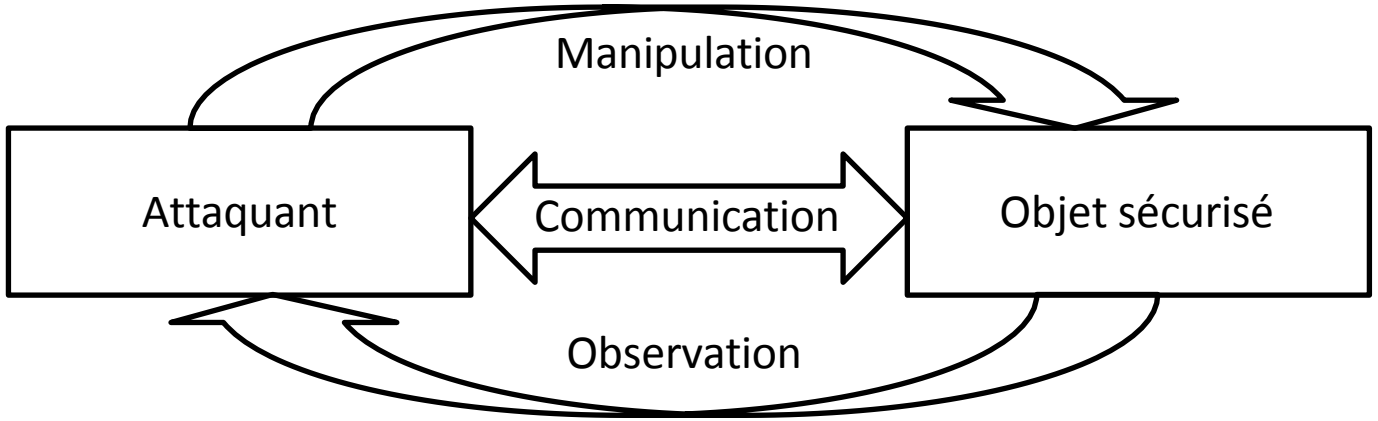
- Porte-monnaie électronique: le client peut vouloir s'ajouter de l'argent
- DRM: le client peut vouloir copier le contenu (film, musique, ...) pour ses amis
- Protection des marques: le client peut vouloir utiliser des encres, batteries moins chères
- …
- Résumé: beaucoup de scénarios où casser le système à un intérêt pour l'utilisateur
- → l'attaquant est en possession de l'objet attaqué et peut donc utiliser d'autres chemins d'attaques plus puissants

Attaques physiques sur systèmes embarqués

- Attaques logiques ne se basent que sur des informations récupérées lors de communications



- Attaques physiques observent et manipulent les propriétés du système ou de son environnement



Attaques physiques

- Dans une attaque physique, l'attaquant fait quasiment ce qu'il veut avec l'objet
- Avec assez de ressources, l'attaquant peut reuissir à casser le système en plus ou moins de temps
- En pratique, le but est de rendre les attaques tellement difficiles à réaliser qu'elles deviennent inutiles

Basiques du design hardware d’un circuit

Pourquoi s'intéresser au hardware?

- Certaines attaques spécifiques à l'embarqué utilisent l'implémentation physique du composant pour exploiter une faiblesse
- Il faut avoir une compréhension globale du fonctionnement d'un composant pour pouvoir se protéger

Caractéristiques physiques d'une carte à puce

- Micro-processeur de 8, 16 ou 32 bits
- De 2Ko à 32Ko de mémoire RAM
- De 32Ko à 512Ko de mémoire non volatile (Flash/ROM)
- Surface de quelques mm^2
- Modules de sécurité

Design synchrone

- La grande majorité des circuits sont synchrones
- Propriétés d'un circuit synchrone
  - Un état est stocké dans des éléments mémoires (registres, RAM, portes logiques, ...)
  - Des éléments de circuits combinatoires effectuent des calculs
  - Un signal d'horloge déclenche la mise à jour des élements mémoires
- Les circuits synchrones sont des *finite state machine* (automate fini) déclenché par un signal d'horloge

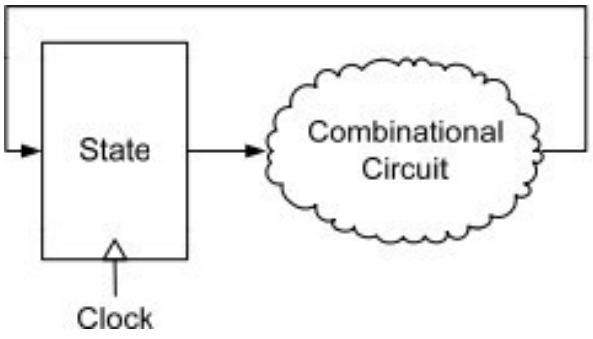
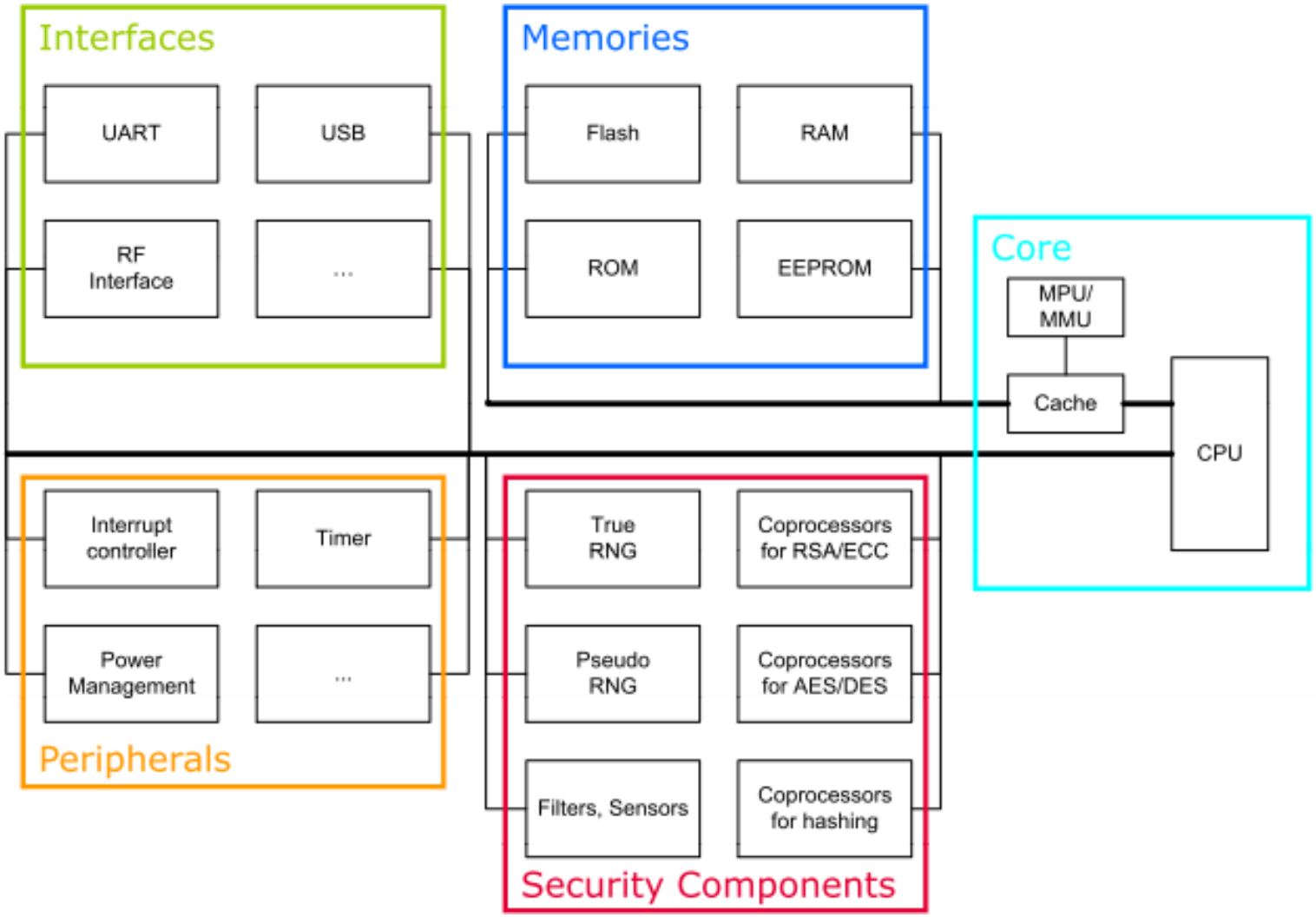
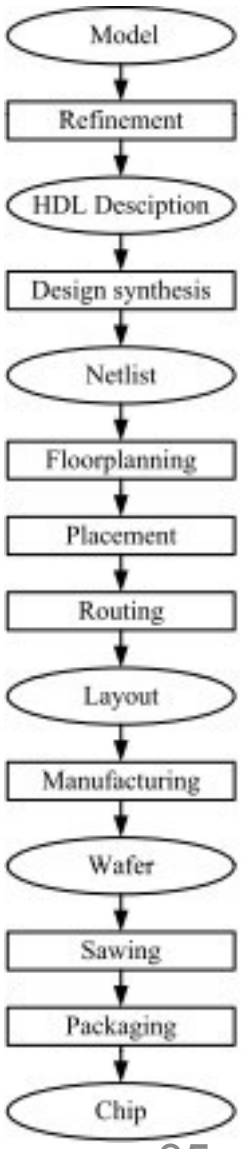


Diagramme de bloc typique d'un circuit



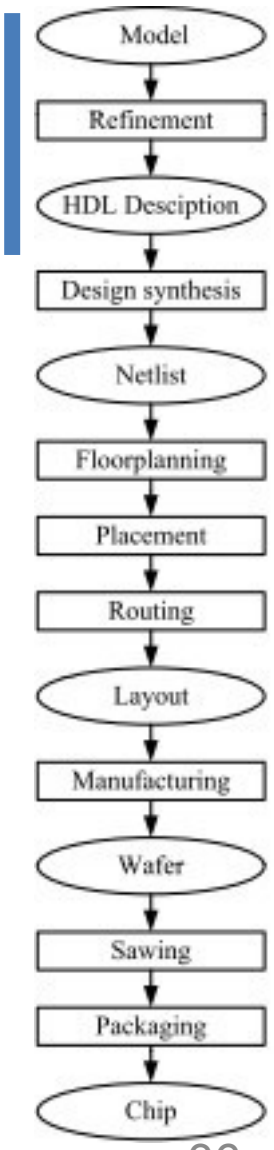
Processus de développement du hardware

- A un certain niveau, la conception du hardware est similaire à du software
- Comme en software, il existe des niveaux d'abstractions qui permettent de gérer la complexité



Modélisation et description HDL

- Le processus de design commence par une modélisation du hardware en software
- Un modèle de haut-niveau est constamment rafinée jusqu'à obtenir une description du design au *register-transfer-level* (RTL)
- Au niveau RTL, le hardware est décrit en utilisant un language de description (HDLs)
  - Ex: Verilog, VHDL
- A ce niveau, le hardware peut être vu comme une FSM qui met à jour son état à chaque coup d'horloge



Code HDL

and_gate.vhd

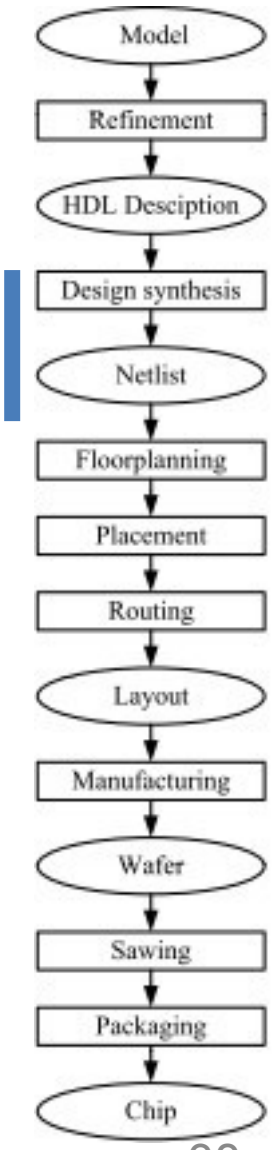
```
1   library IEEE;
2   use IEEE.STD_LOGIC_1164.ALL;
3   use IEEE.STD_LOGIC_ARITH.ALL;
4   use IEEE.STD_LOGIC_UNSIGNED.ALL;
5
6   --  Uncomment the following lines to use the declarations that are
7   --  provided for instantiating Xilinx primitive components.
8   --library UNISIM;
9   --use UNISIM.VComponents.all;
10
11  entity and_gate is
12      Port ( a : in std_logic;
13             b : in std_logic;
14             c : out std_logic);
15  end and_gate;
16
17  architecture Structural of and_gate is
18
19  begin
20       c <= a and b;
21  end Structural;
22
```

Implémentation d'une porte AND

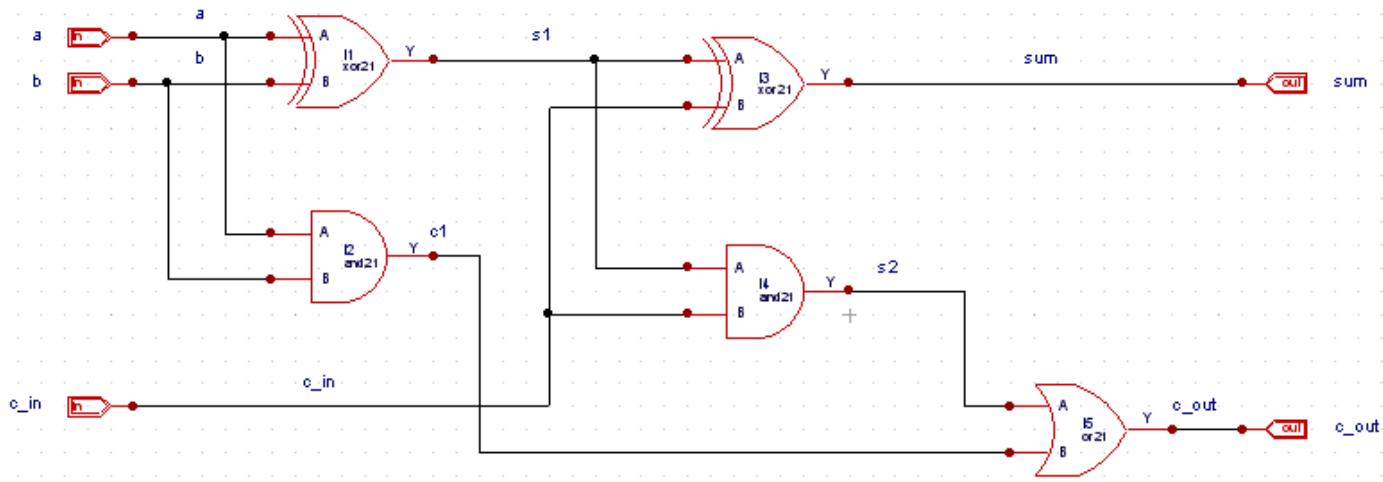
c = a AND b

Synthèse du design

- Pour chaque technologie de logique, il existe des librairies de portes logiques
  - Ex: NAND, OR, AND, …
- L'outil de synthèse transforme la description HDL en des portes logiques de la librairie
- La sortie de la synthèse est une *netlist*
  - Graphe où chaque nœud est une porte logique



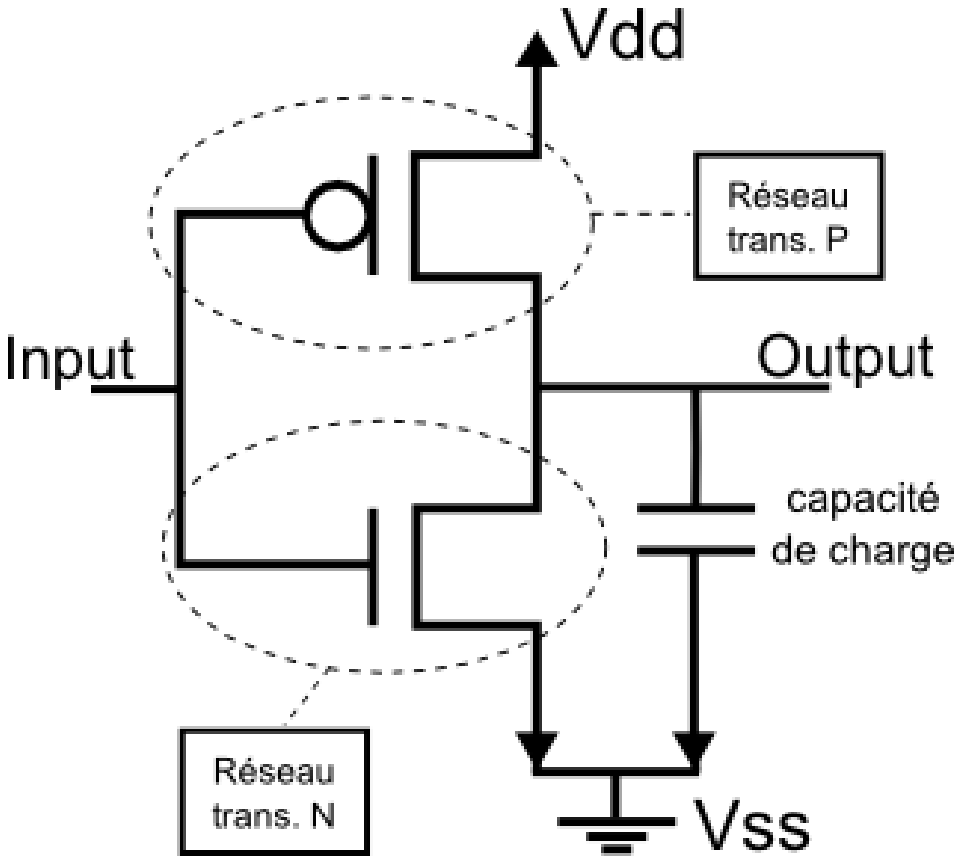
Netlist – niveau porte



Netlist – niveau transistor

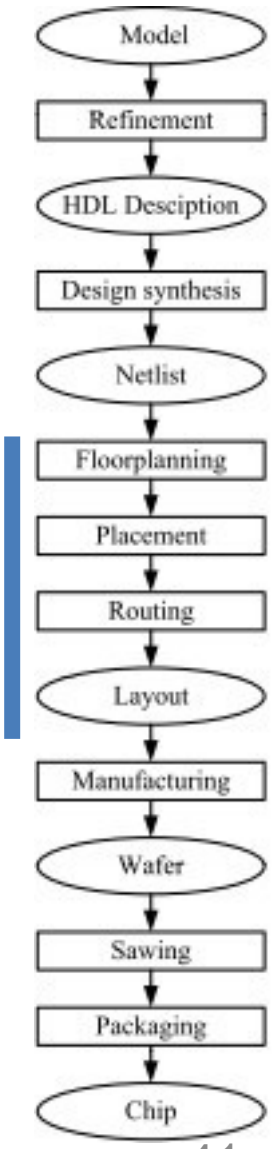
Implémentation d'un inverseur CMOS

Input=0 →Output=1
Input=1 →Output=0

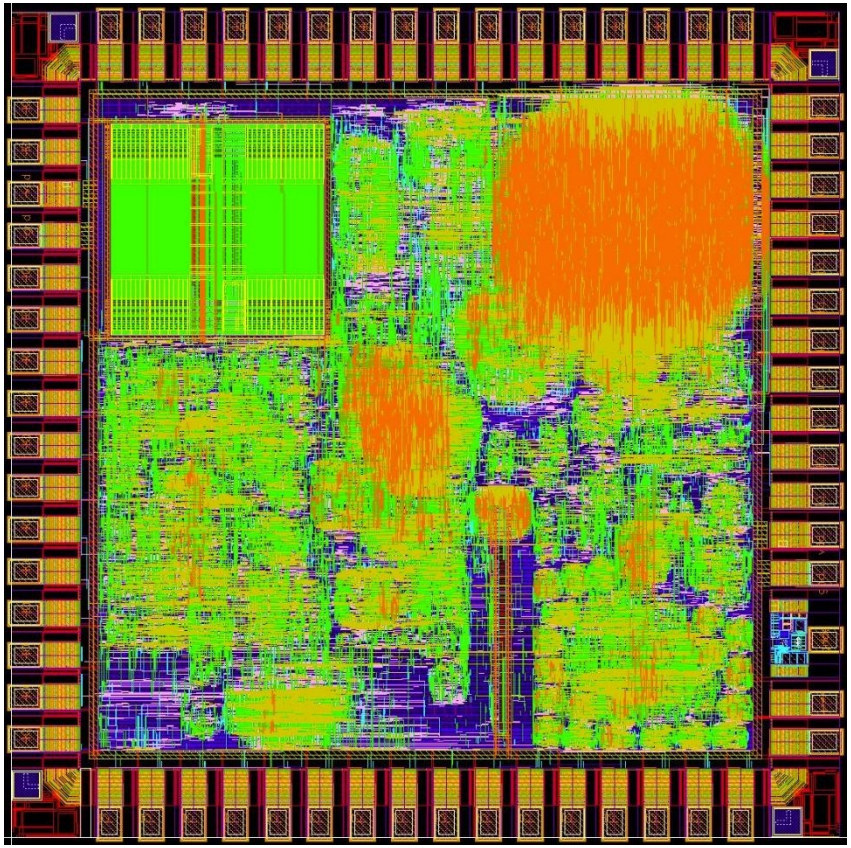


Layout

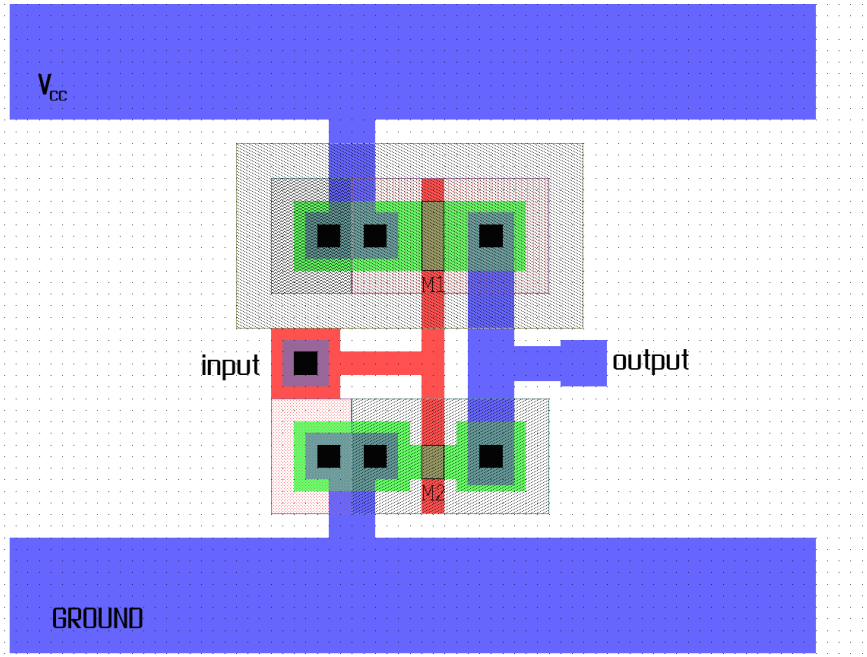
- Les portes sont placées et connectées
- Dans cette étape, de nombreuses contraintes physiques doivent être prises en compte
  - Placer proches les portes connectées
  - Minimiser le surcoût du placement
  - …
- Le *layout* est une description géométrique du design qui décrit à quel endroit doit être placé chaque élément



Exemples de layout



Layout d'un circuit complet



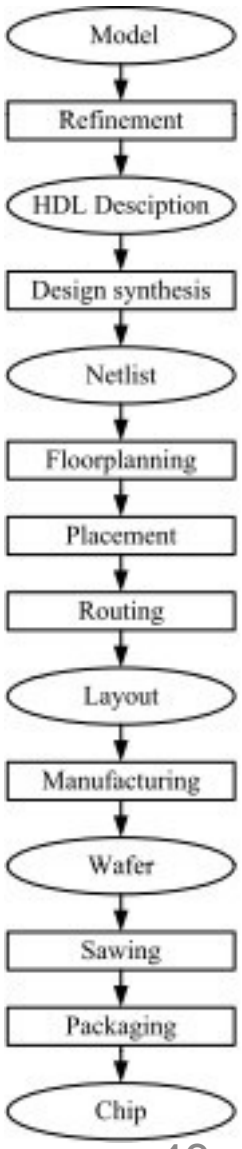
Layout d'un inverseur CMOS

Fabrication

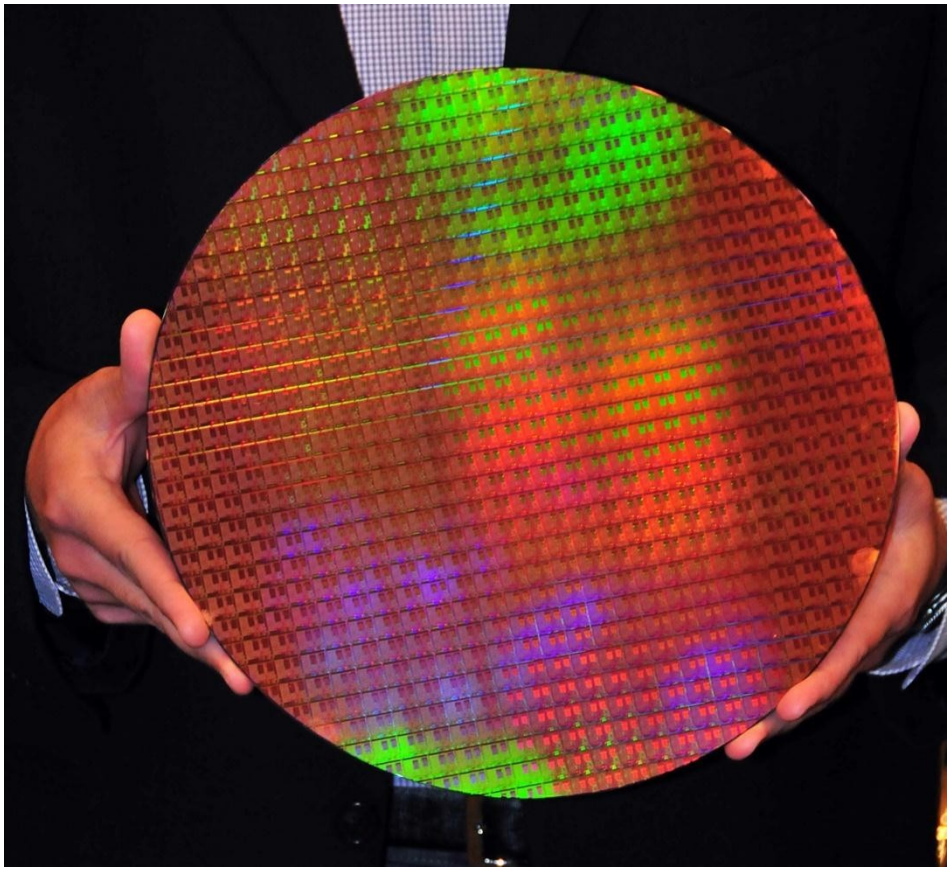
- Après la création du *layout*, il est envoyé à l'usine de fabrication

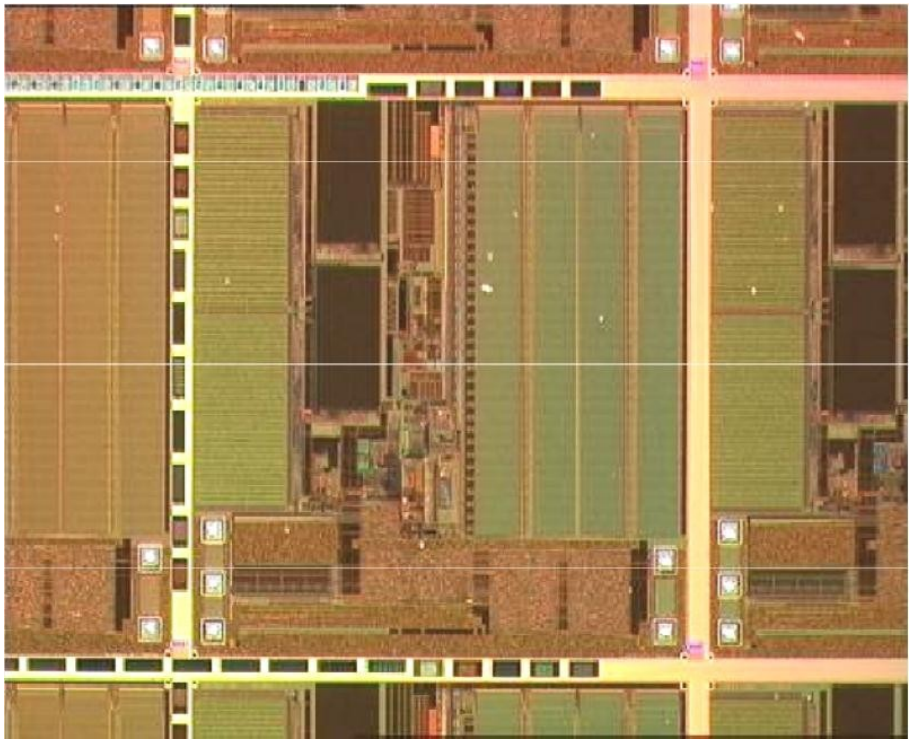
- Technologies courantes:
  - 120nm, 90nm

- Taille des wafers
  - 200mm, 300mm



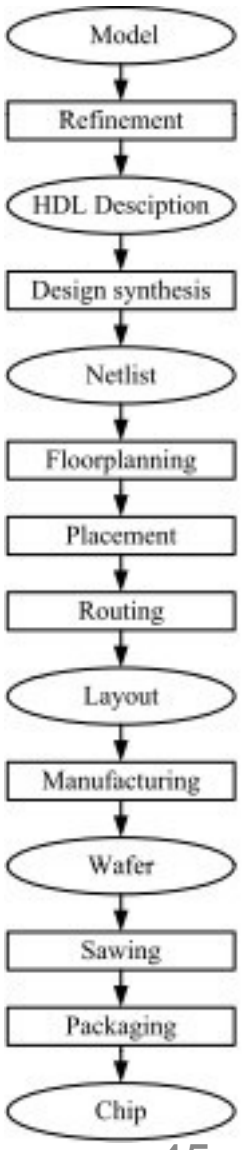
Wafer



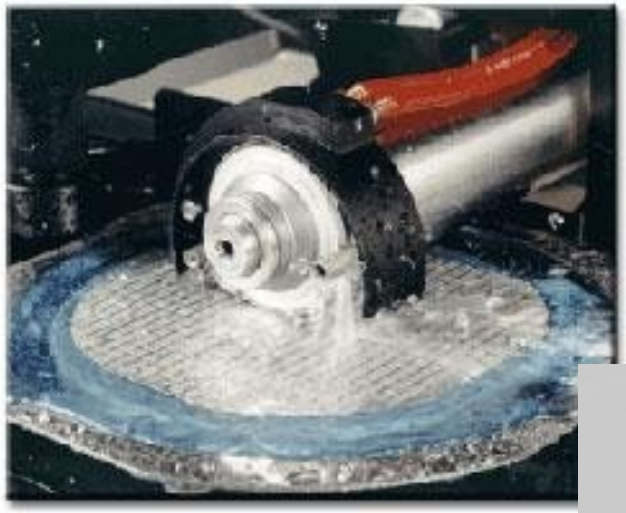


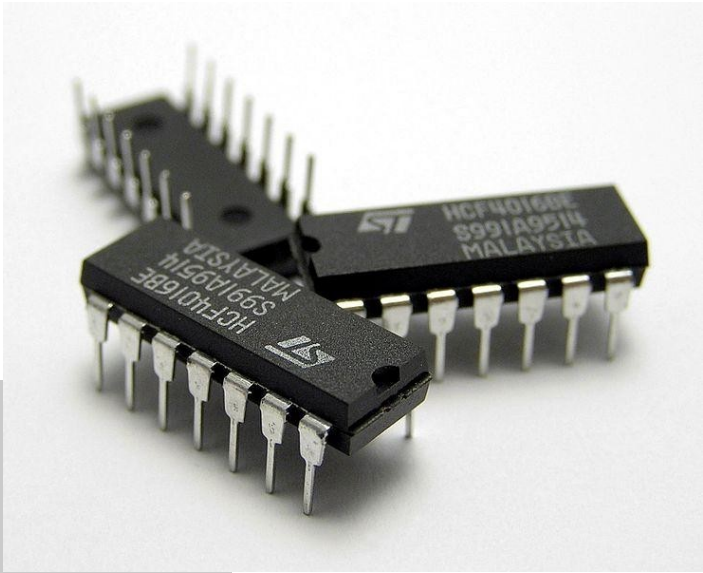
Découpe, Test et Packaging

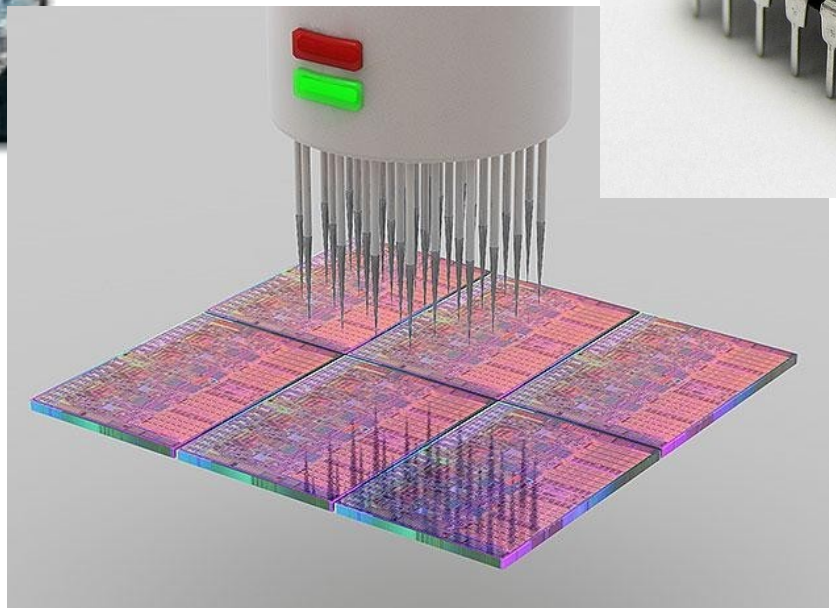
- Une fois le wafer produit, il reste à
  - Découper: séparer tous les circuits présents sur le wafer
  - Tester: le processus de fabrication n'est pas parfait, il faut donc tester tous les circuits produits
  - Packager: protège le circuit dans un petit bloc de matériel semi-conducteur pour le protéger de la corrosion et de dommages physiques



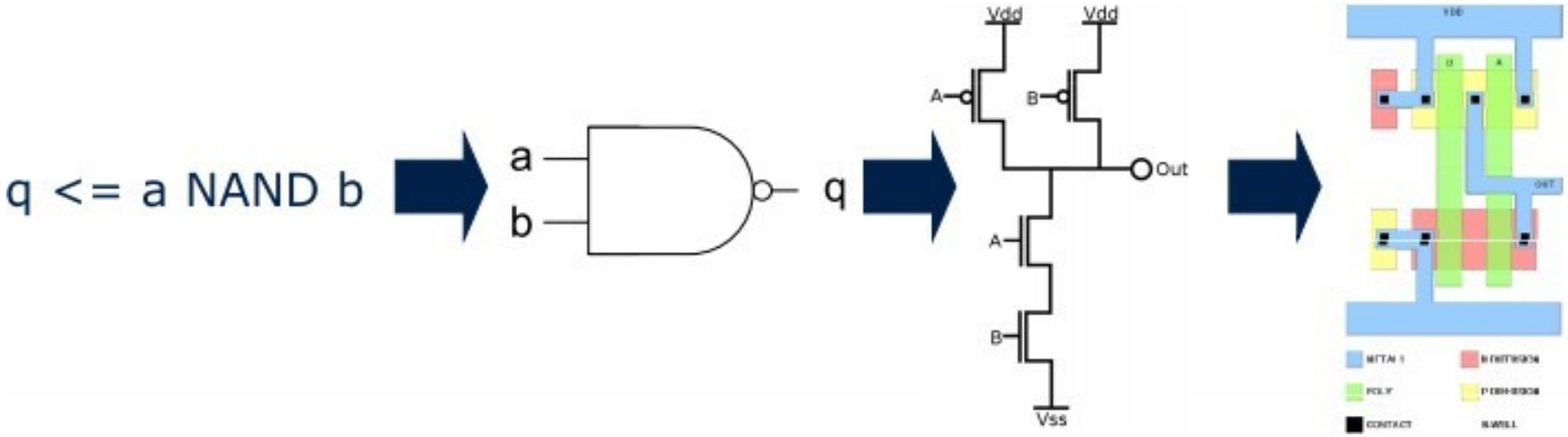
Découpe, Test, Packaging







Résumé d'un design hardware



- Le design hardware est un processus complexe avec beaucoup de niveaux d'abstraction
- Les étapes d'optimisations présentes à chaque niveau peuvent influencer la sécurité finale du composant

Classification des attaques physiques

Attaques physiques

- Deux critères principaux
- Comportement de l'attaquant
  - Actif: il agit, modifie le comportement du circuit
  - Passif: il observe certains propriétés physiques du circuit
- Degré d'implication de l'attaquant
  - *Invasive*: il n'a aucune limite (coûteux)
  - *Semi-invasive*: il peut enlever le packaging mais ne touche pas à la structure interne du circuit (abordable)
  - *Non-invasive*: observe et manipule le circuit sans modifications physiques (très peu coûteux)

Principe d'une attaque active

- Manipuler/Modifier le circuit pour exploiter des faiblesses
- Changer le comportement général du circuit
  - Activer le mode de test
  - Désactiver les contre-mesures, détecteurs
  - Changer le programme (sauter la vérification PIN, etc)
  - …
- Insérer des fautes dans les calculs crypto
  - L'attaquant faute l'algorithme pendant son exécution et récupère le résultat fauté
  - Il exploite des propriétés mathématiques de l'algo pour retrouver la clé

Principe d’une attaque passive

- Observer les propriétés physiques du circuit pour retrouver le secret
- Le secret est manipulé par le circuit (lors d’opérations crypto par ex)
- Des informations sur le secret peuvent être observée en surveillant
  - Le temps d’exécution
  - La consommation de courant
  - Les émanations électromagnétiques
  - …

Attaques physiques principales

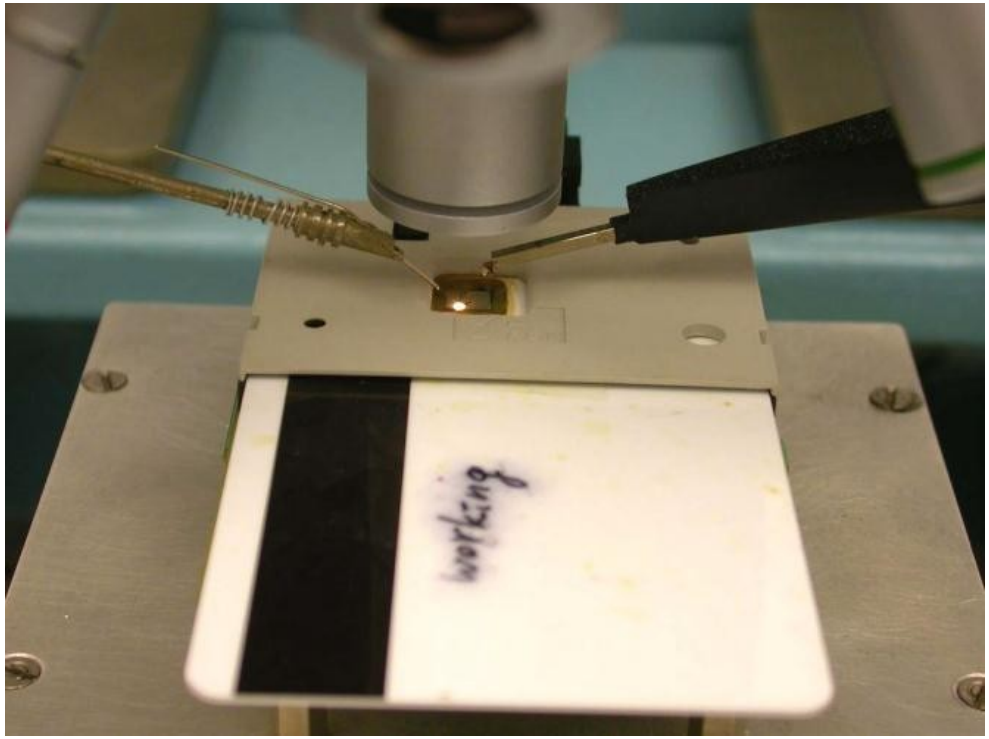
	Active	Passive
Non-invasive	Glitching, changement de température, faible voltage, …	Attaques par canaux cachés (*side-channel attack*)
Semi-invasive	Attaques par lumière, radiation, …	Attaques EM, inspection optique
Invasive	FIB	Probing

Active invasive attack

- Le composant est manipulé et modifié par l'attaquant
- Le circuit peut être modifié par un *focused ion beam* (FIB) (sonde ionique focalisée)
- Equipement **très** coûteux mais possibilités énormes pour l'attaquant

Active semi-invasive attack

- Injection de faute (*fault attack*)
  - Composant dépackagé et placé sous un microscope
  - Les transistors peuvent être changés d'état par une lumière forte et focalisées (laser)



Active non-invasive attack

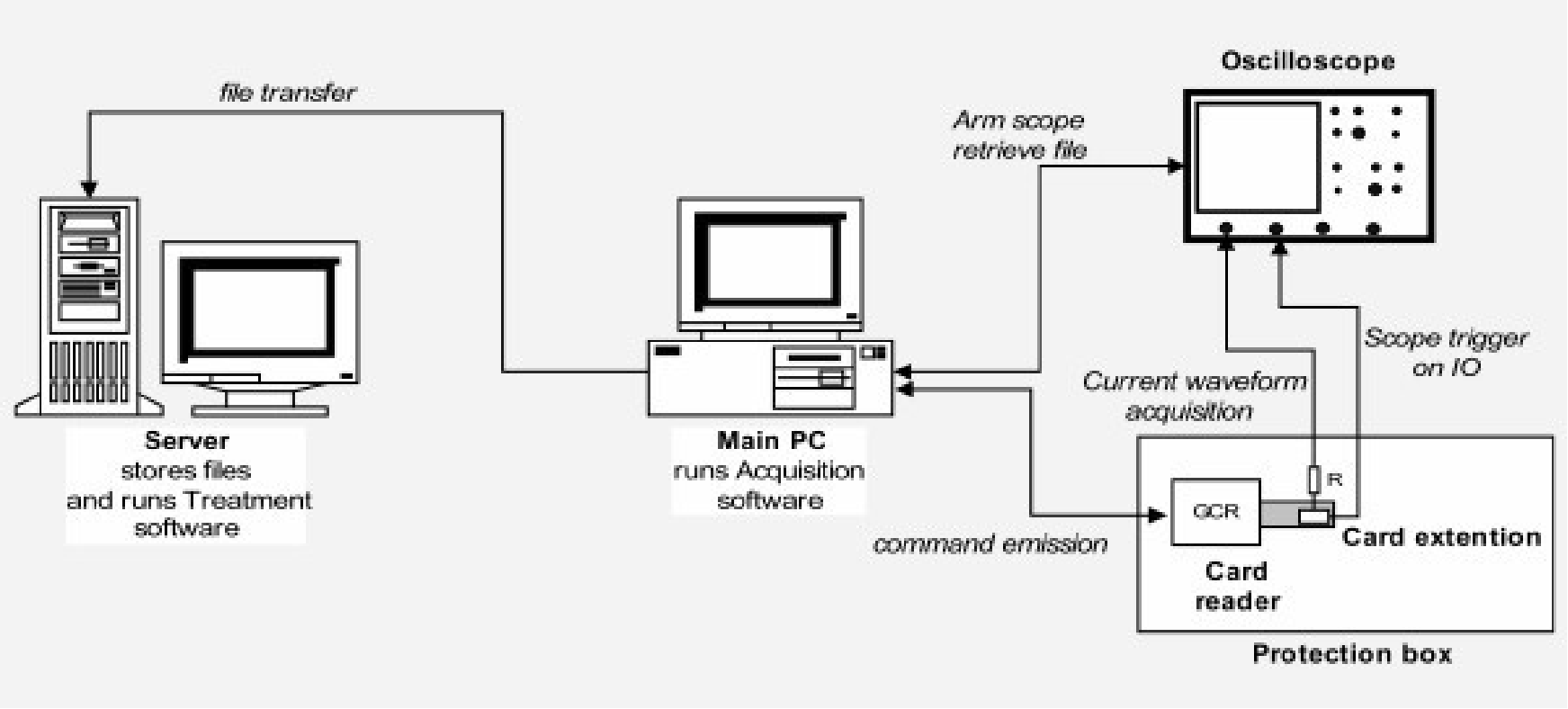
- Perturber le fonctionnement du composant sans le dépackager
- *Glitching*
  - Perturber l'alimentation en courant du composant pendant son fonctionnement peut provoquer des sauts d'instruction
  - Perturber l'horloge extérieure peut provoquer des corruptions de données ou des sauts d'instruction
- Température
  - Modifications aléatoires dans la RAM
  - Opérations de lecture erronées dans les NVMs

Passive non-invavise attacks: side-channel attacks

- Idée: révéler le secret en observant les propriétés physiques du composant
- *Timing attack*
  - Mesure le temps d'exécution
- *Power attack*
  - Mesure la consommation de courant
  - Peu coûteux en équipement: un PC avec un oscilloscope et une petite résistance sur l'alimentation en courant du composant
  - Attaque très efficace
  - Deux méthode basiques: attaque simple (SPA) et différentielle (DPA)

Power Analysis

- Equipement



Conclusion

Conclusions

- Il n’existe pas de protection absolue
  - Avec assez de temps et ressources toute contremesure peut être cassée
- L’objectif en pratique est de rendre un système tellement cher à attaquer que ce n’est plus rentable pour un attaquant
- Domaine en constante évolution
  - Protections doivent tenir compte des possibles évolutions du matériel et techniques des attaquants