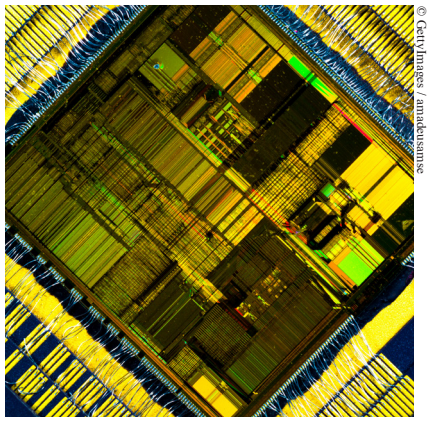
Quand les puces font des maths

L'industrie fournit des logiciels aux concepteurs des puces, omniprésentes dans notre quotidien. Ces logiciels utilisent massivement des notions mathématiques avancées. Les forts en algèbre linéaire, en *machine learning* ou en statistiques et probabilités y trouveront un formidable terrain de jeu.

Cyril Desclèves travaille à Siemens EDA qui compte parmi les leaders mondiaux du marché de l'automatisation de la conception électronique.

Elles sont partout ! Dans votre téléphone portable ou votre ordinateur, bien sûr, mais aussi dans votre voiture ou même votre lave-vaisselle. Sans parler des trains, des satellites, des pacemakers et des appareils d'IRM… Fabriquer une puce (de son vrai nom, un circuit intégré) n'est pas une mince affaire. Totalement banalisée pour le grand public, c'est une prouesse technologique qui consiste à graver dans un matériau semi-conducteur (le silicium est roi) des motifs nanoscopiques en métal, en poly-silicium, etc. L'élaboration et le contrôle du procédé technologique font appel à des spécialistes en physique des semi-conducteur et en photolithographie. Les quelques dizaines de mm^2 (les méga-puces font même quelques cm^2) d'un circuit intégré contiennent des millions, voire des milliards, de « transistors » (qui n'ont rien à voir avec la radio…).



© GettyImages / amadeusamse

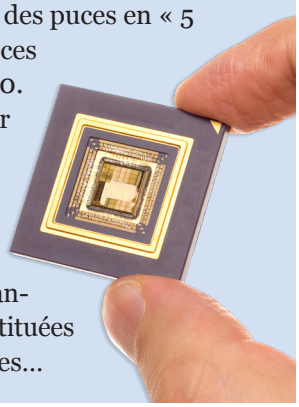
Individuellement, un *transistor* peut être vu comme une sorte d'interrupteur imparfait, qui laisse passer plus ou moins de courant à travers lui, modulé par une tension de contrôle. Mais l'union fait la force, et en connectant savamment un grand nombre de transistors, on peut créer une fonction de stockage (pour votre clé USB par exemple), de calcul (pour les microprocesseurs, au cœur de tout ordinateur), d'amplification, de décodage audio, vidéo, Wi-Fi… Au fond, une puce n'est rien d'autre qu'un circuit élec-

tronique géant, qui va donc manipuler des courants et des tensions, mais que l'on fait tenir dans un tout petit volume. Au fil des ans, le niveau d'intégration a littéralement explosé (voir encadré). On peut ainsi choisir d'embarquer plus de fonctionnalités dans le même volume, ou de réduire le volume à fonctionnalités constantes.

Au passage, on a aussi trouvé comment réduire encore et encore la consommation, comment « empiler » certains circuits les uns sur les autres, toujours pour gagner de la place (on parle de 3D-IC), ou encore à combiner de l'électronique classique avec des dispositifs optique intégrés (la photonique).

De la microélectronique à la nanoélectronique

En 1985, la dimension caractéristique des procédés avancés était le « 1 µm » (le diamètre d'un cheveu est d'environ 100 µm). Un transistor occupait quelque 2 µm². C'était fabuleux. Mais l'idée même d'une puce mémoire de 128 Gb était de la pure science-fiction. En 2020, on produit des puces en « 5 nm ». Le ratio linéaire entre ces deux géométries est de 200. L'électronique nécessaire pour votre ordinateur portable ne tiendrait pas dans une camionnette si l'on utilisait la technologie de 1985. Les sections les plus fines d'un transistor modèle 2020 sont constituées de quelques couches d'atomes...



© Adobe Stock / Pavel Kirichenko

Pas le droit à l'erreur!

Formidable, mais… Quand on lance la fabrication d'un circuit intégré, on mobilise une chaîne de production dans une usine high-tech, et on n'a peu (voire pas du tout !) le droit à l'erreur. Il n'y a pas l'équivalent du fer à souder du monde de la carte électronique, qui permet de recâbler correctement un composant, rebrancher et voir si cela marche mieux… Non, avec un circuit intégré, à la moindre erreur, il faut refaire un cycle de fabrication complet, ce qui prend des semaines et coûte une petite fortune. Le temps perdu, surtout, est un vrai désastre dans notre économie ultra-concurrentielle. Les concepteurs de circuits intégrés sont donc sous pression de produire un schéma comprenant quelques milliards de composants interconnectés, qui fonctionne « du premier coup »… Un véritable défi de tous les instants. Aussi existe-t-il une large panoplie de logiciels ultraspécialisés. Le terme métier est anglo-saxon, c'est l'EDA, pour *electronic design automation*. Ces logiciels assistent le concepteur dans toutes les phases, de l'idée originale jusqu'au test des puces fabriquées. Au vu de la complexité, ces logiciels ne sont pas optionnels – il est impossible de s'en passer. De nombreuses phases sont nécessaires, plus ou moins automatisées : la synthèse logique, le placement–routage, la vérification physique, l'extraction de parasites (oui, les puces ont aussi leurs propres parasites), la simulation, l'émulation…

Au cœur de tous ces logiciels, on trouve… des mathématiques. Pour le placement–routage, on va utiliser massivement des notions de théorie des graphes – le logiciel place les blocs élémentaires qui composent le circuit et les interconnecte de façon à minimiser la surface totale utilisée, mais aussi les longueurs des connexions, etc. Pour la simulation, on utilisera de l'optimisation, de l'algèbre linéaire, du traitement de signal, des régressions et classifications, des probabilités…

Les modèles de transistors

Les modèles des courants I et charges Q des transistors sont des relations de type $I = i(V, P)$ et $Q = q(V, P)$ où P est un vecteur de paramètres. C'est tout à fait semblable à la relation $I = (1/R)V$ (la loi d'Ohm) pour une résistance, ou à $Q = C.V$ pour une capacité C. Mais un transistor est un dispositif fortement non linéaire et les équations nécessaires sont beaucoup plus sophistiquées. Pour les technologies avancées, il faut au bas mot deux cents paramètres, et des milliers de lignes de code pour évaluer $i(V, P)$ et $q(V, P)$, ainsi que leurs dérivées partielles, pour une seule valeur de tension V. Bien sûr, la taille physique du transistor est capitale, et la température de fonctionnement aussi.

Simulation électrique et mathématiques

Intéressons-nous à l'aspect simulation. Avant de lancer la fabrication d'une puce, on va modéliser ses composants et simuler son fonctionnement, de façon à minimiser, et idéalement annuler, le risque de produire ce qu'on appelle un « bout de bois ». On travaille avec de multiples niveaux d'abstraction (système, architecture, registre, porte, et jusqu'au niveau électrique). La simulation électrique s'intéresse spécifiquement à modéliser et vérifier la performance du circuit (fréquence maximale d'utilisation, consommation, bruit…), et non plus seulement sa fonctionnalité logique. Pour simuler électriquement un circuit, il faut disposer d'une description de sa connectivité, et de modèles détaillés de chaque composant actif (transistor) et passif (les interconnexions). Ces modèles fournissent les courants et charges dans les composants (voir encadré). Les paramètres de ces modèles eux-mêmes sont obtenus grâce à des logiciels dédiés, qui font de l'optimisation non linéaire pour déterminer les meilleurs paramètres du modèle, à partir de données mesurées sur des puces tests.

La mise en équations la plus classique et générale consiste à utiliser la loi des courants, qui produit un (grand) système d'équations différentielles non linéaires $F(V, \dot{V}, t) = 0$, où V est le vecteur des inconnues au temps t. La fonction F ne peut être évaluée que point à point. Ce système doit être résolu numériquement. Les concepteurs de circuit utilisent plusieurs types d'analyses.

Analyse statique, analyse temporelle

La première de ces analyses est purement statique. On oublie le temps, et on veut connaître les tensions et courants statiques partout dans le circuit, lorsqu'il est alimenté. On se ramène donc à devoir résoudre un système d'équations non linéaires $F_S(V) = 0$, et le plus classique consiste à utiliser des variantes autour d'une méthode itérative de type Newton–Raphson. Cette *analyse statique* est numériquement difficile, contrairement aux apparences. Il peut exister des solutions multiples, rendant très difficile leur détection. Et la grande dimension n'arrange rien. De fait, il n'est pas rare dans un simulateur commercial de trouver une dizaine de méthodes alternatives à l'idée de base, souvent de type continuation, pour tenter de contourner ces difficultés numériques.

La deuxième de ces analyses, plus critique généralement car extrêmement gourmande en temps de calcul, est l'*analyse temporelle*. On veut cette fois connaître l'évolution dynamique des signaux $V(t)$ lorsqu'on applique des signaux externes $S(t)$ qui varient dans le temps (un signal d'horloge, une rampe, un signal sinusoïdal…). Puis vérifier que les sorties, produites par le circuit

soumis à ces stimuli, basculent bien dans un délai maximal prévu pour le bon fonctionnement, ou que la consommation électrique ne dépasse pas les valeurs autorisées…

Revenons à notre équation générale $F(V, \dot{V}, t) = 0$. Pour approximer $V(t)$, on discrétise le temps en petits intervalles, et on applique un schéma d'intégration implicite, qui consiste à remplacer $\dot{V}(t)$ par une combinaison linéaire de $V(t)$ et quelques valeurs précédent l'instant courant t. On se ramène ainsi de nouveau à devoir résoudre un système de type $F_d(V) = 0$, et cette fois, à chaque pas de temps (qui lui-même prend plusieurs itérations), on doit résoudre un gigantesque système linéaire. Le contrôle dynamique du pas de temps et de l'erreur commise est aussi tout un art.

Un peu (beaucoup) d'algèbre linéaire

Pour notre matrice, l'une des difficultés provient du fait que la dimension n du problème, et donc la taille $n \times n$ de la matrice associée, est très grande (de l'ordre de plusieurs milliers ou millions). Cette matrice pourrait avoir le bon goût d'être symétrique, mais… non. Ou bien avoir une structure particulière, mais… non plus. Elle pourrait au minimum être bien conditionnée… mais non, toujours pas. La seule chose qui la sauve un peu – c'est une conséquence de la connectivité typique d'un circuit électrique, où chaque composant est généralement connecté à quelques voisins seulement –, c'est qu'elle est « creuse », elle contient un très grand nombre de 0. Heureusement, tout un pan de l'algèbre linéaire est spécialisé dans le traitement des matrices creuses. On va pouvoir optimiser sa factorisation, par exemple en introduisant une résolution hiérachique par blocs.

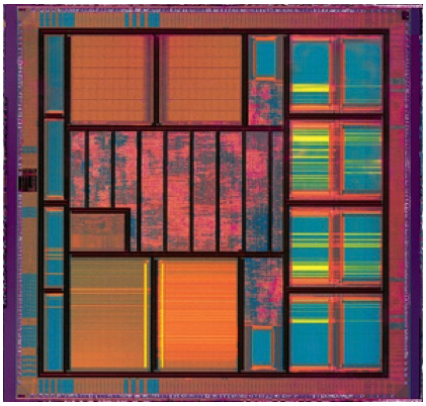
© Adobe Stock / Marek

Pour les circuits opérant avec des signaux radio, on est confronté à un autre problème. Un signal porteur oscille à quelques GHz, mais on veut pouvoir simuler des durées très longues, relativement à la période de cette porteuse, peut-être des millisecondes, soit des millions de cycles de la porteuse. C'est totalement rédhibitoire avec une analyse temporelle telle que décrite précédemment, car ce serait beaucoup trop long. Il existe plusieurs solutions, l'une d'elles consistant à réécrire les équations dans le domaine fréquentiel, pour des signaux supposés quasi périodiques. On décrit les signaux temporels en utilisant des transformées de Fourier discrètes, sous forme d'une série harmonique. On détermine alors les modules et phases des composantes harmoniques, éventuellement modulées, qui deviennent les inconnues du système transformé. On utilise aussi beaucoup d'algèbre linéaire, mais les matrices obtenues se prêtent bien mieux à des méthodes de résolution itératives de type GMRES. Il faut aussi être très à l'aise avec la notion de dualité temps–fréquence, et jongler habilement entre les deux domaines. Dans le même ordre d'idées, l'utilisation d'ondelettes a été testée, mais sans grand succès.

Probabilités d'évènements rares

Prenons une mémoire de 256 Mb. Supposons que chaque cellule de point-mémoire a une probabilité P_{err} de ne pas fonctionner (et que ces évènements sont indépendants). Le plan mémoire entier a alors une probabilité $(1 - P_{err})^{256\,000\,000}$ de fonctionner. Supposons que l'on cherche un rendement de 95 %. Il faut alors que P_{err} vaille environ 2×10^{-10}. Il importe donc de s'intéresser à des probabilités de défaillance « très faibles », pour lesquelles les simulations de type Monte Carlo naïves ne sont d'aucune utilité pratique. En effet, pour observer un évènement dont la probabilité est p, il faut en moyenne $1/p$ essais. Dans notre exemple, il faudrait cinq milliards de simulations pour espérer voir une défaillance. Même avec cent machines disponibles en parallèle et une simulation unitaire prenant une seconde, il faudrait cinq cent soixante-dix-huit jours de calcul. Il va falloir être beaucoup plus astucieux, et utiliser d'une façon ou d'une autre des techniques d'échantillonnage préférentiel.

Des puces à l'humeur variable



© Wikimedia/Commons

On doit aussi modéliser et simuler des aspects électrothermiques (la température en divers endroits d'une même puce n'est pas uniforme) et même le vieillissement. Les forts courants à travers de si petits dispositifs arrivent à dégrader la structure cristalline et affecter les performances, sur des temps longs – car les pannes ne sont pas seulement dues aux méchants industriels qui programment l'obsolescence des produits.

La puce a des sautes d'humeur, c'est bien connu. Alors quand on lance la production en série d'un circuit intégré, non seulement il faut être sûr qu'il va fonctionner, mais il faut aussi être certain que, d'un circuit à l'autre, la variation de performance due à ces sautes d'humeur va rester acceptable. Par exemple, un amplificateur à bas-bruit destiné à amplifier le signal radio reçu par l'antenne de votre portable n'aura jamais exactement le même gain d'une puce à l'autre. La variation de performance est induite par les variations du procédé de fabrication. On imagine assez bien que de produire des structures de 5 nm exactement semblables à travers une plaque (ou *wafer*) de 30 cm de diamètre n'est pas simple. On recense deux grands types de variations aléatoires dans un circuit intégré. Des variations *globales* (ce jour-là, pour ce lot-ci, les réglages de telle ou telle étape ont bougé un tout petit peu, et tout le lot de circuits produits a été affecté – c'est similaire à ce qui se produit dans tout processus industriel). Et des variations *locales*, beaucoup plus dévastatrices pour le rendement de fabrication. En effet deux transistors supposément identiques, gravés l'un à côté de l'autre, ne seront pourtant pas exactement jumeaux.

Ces variations locales se modélisent aussi, et on peut alors, toujours par simulation, estimer leur impact sur la variation des sorties. Revenant à notre modèle de courant $I = i(V, P)$, on attache, à certains des paramètres dans P, une distribution de probabilité, et non pas une valeur fixe. Généralement, c'est une gaussienne standard, entièrement déterminée par une valeur moyenne et un écart type. On gère en effet les

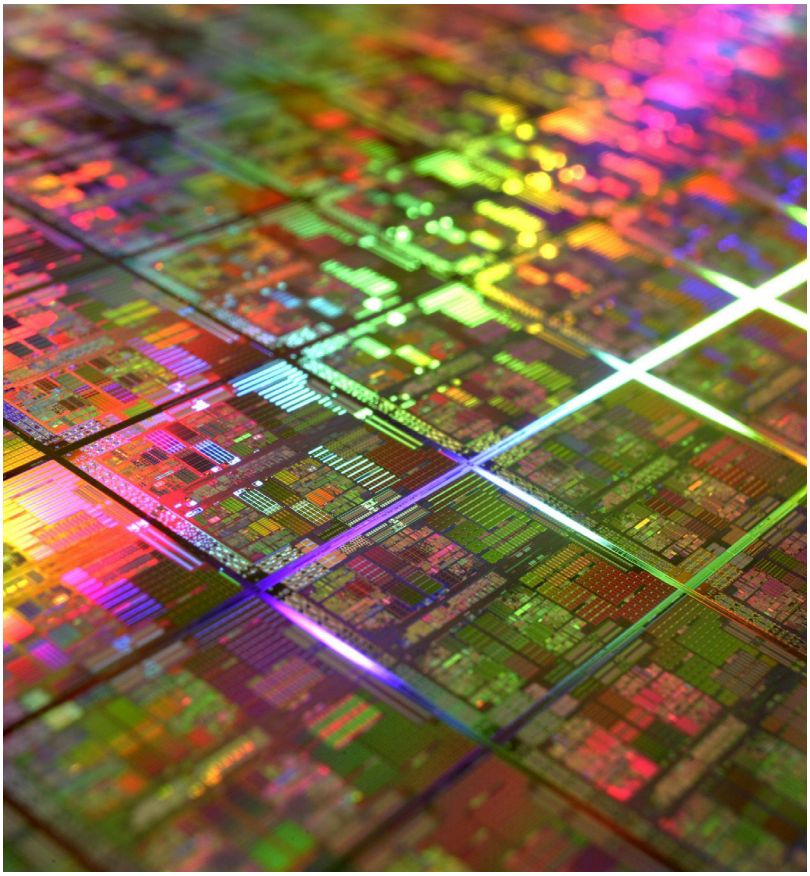
éventuelles corrélations en amont et on peut toujours se ramener à des variables indépendantes et distribuées selon la loi normale standard.

On estime alors la distribution d'une métrique M (un gain, un temps de propagation, un courant…), elle-même résultat d'une simulation : $M = H(P_g)$. De nouveau, la forme de la fonction H est inconnue, c'est une « boîte noire », un code de simulation qui retourne point par point le résultat d'une simulation statique, temporelle ou radio-fréquence, telles que précédemment décrites. On utilise alors une méthode de Monte Carlo pour estimer la distribution de M. Ce qui consiste à tirer N ensembles de valeurs pseudo-aléatoires pour P_g, selon leurs distributions, et à exécuter à chaque fois la simulation correspondante pour calculer $M = H(P_g)$. On obtient ainsi N réalisations pseudo-aléatoires de M à partir desquelles on peut estimer des moyennes, des écarts types, des probabilités ou encore des quantiles.

Un concepteur se pose des problèmes du style : quelle est la probabilité p que le gain de cet amplificateur soit inférieur à 45 dB ? Et quel est l'intervalle de confiance associé à cette estimation de p ?

Tout se passe bien tant que l'on travaille avec des niveaux de probabilité élevés, et que l'on peut se contenter d'utiliser des nombres de tirages N « raisonnables ». Mais quand on cherche des probabilités faibles (10^{-9}), les choses se corsent. Mais pourquoi diable devrait-on se préoccuper de probabilités de l'ordre de 10^{-9} ? Eh bien, à cause de ces variations locales, qui se trouvent être indépendantes (voir encadré). En probabilité comme en statistiques, c'est le monde des *évènements rares*.

Deux autres grands domaines, au moins, qui n'ont rien à voir avec le monde du



© DR.

circuit intégré, s'attaquent aussi à ce sujet. Celui de la finance et celui de l'ingénierie structurelle, mécanique. Les financiers veulent modéliser et estimer les rendements ou les risques de produits financiers. Les mécaniciens s'intéressent à la fiabilité des structures, des ponts aux ailes d'avion, en passant par les cuves des réacteurs nucléaires. Il existe des synergies algorithmiques possibles, mais souvent le problème est assez différent. Chez les puces, la dimension de P_g est immensément plus grande. Avec typiquement deux à quinze paramètres aléatoires par transistor, on est très vite confronté à des problèmes de très grandes dimensions. Chez les mécaniciens, la fonction M est souvent beaucoup plus diabolique encore.

La méthode de Newton–Raphson

La méthode de Newton–Raphson est un algorithme efficace et incontournable pour trouver numériquement une approximation précise d'un zéro d'une fonction réelle. La version scalaire consiste à résoudre l'équation $h(x) = 0$ avec h une fonction de $\mathbb{R}$ dans $\mathbb{R}$ supposée dérivable. On utilise des itérations du type $x_{i+1} = x_i - \lambda_i \dfrac{h(x_i)}{h'(x_i)}$ avec λ_i un paramètre réel. Il existe autant de critères de convergence que d'applications.
Le paramètre λ_i sert à stabiliser les itérations en limitant l'amplitude des corrections.
La version matricielle de l'algorithme fait intervenir la matrice des dérivées partielles $J(x) = \dfrac{\partial h}{\partial x}$, appelée *matrice jacobienne*. On écrit cette fois $x_{i+1} = x_i - \lambda_i J^{-1}(x_i) h(x_i)$. Dans les faits, on n'inverse jamais cette matrice J explicitement, et on résout plutôt $J(x_i)(x_{i+1} - x_i) = -\lambda_i h(x_i)$. Dans le cas qui nous occupe, $h(x)$ est une fort méchante et coûteuse fonction, et $J(x)$ une gigantesque matrice, creuse mais de structure assez ordinaire, et qui ne manque jamais une occasion de devenir numériquement singulière...

Les espaces de grande dimension sont captivants. L'un des dangers quand on travaille sur des problèmes de probabilités consiste à imaginer que ce qui est « évident » sur un petit dessin en dimension 2 reste vrai en grande dimension. Or rien n'est plus faux. Quand la dimension augmente, deux vecteurs aléatoires deviennent « presque certainement orthogonaux », le volume d'une hypersphère tend vers 0, quel que soit son rayon, et autres facéties… Cette géométrie assez contre-intuitive oblige à avoir de bonnes capacités d'abstraction ! Elle a conduit le mathématicien français Yves Meyer (prix Abel 2017) à sensibiliser la communauté à l'étude des espaces euclidiens de grande dimension. C'est aussi dans ce domaine d'analyse de la variabilité que l'on consomme largement les mathématiques relabellisées « Machine Learning » comme les régressions et les classifications avec pénalisation en norme L^1, les modélisations non linéaires, l'algèbre linéaire (encore !), etc.

C.D.

Production de plaquette de silicium semi-conducteurs en laboratoire.