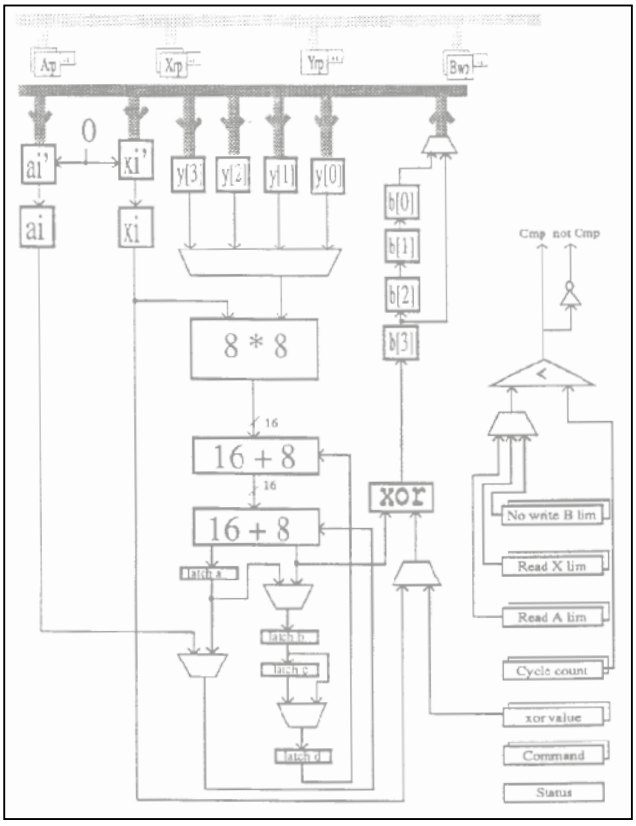
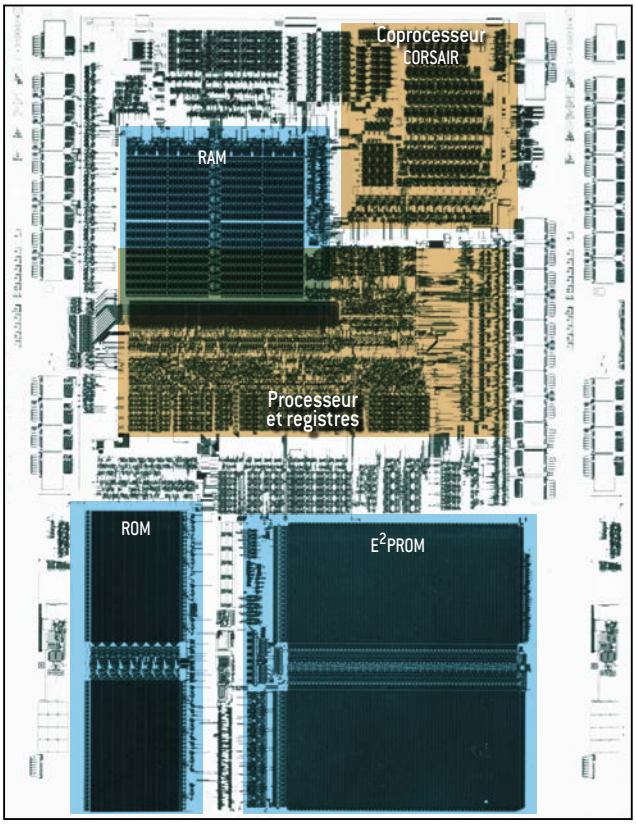




LA PREMIÈRE PUCE À RSA, CORSAIR *(à droite)*, comportait, outre différentes mémoires (ROM, RAM, E²PROM) et un processeur, un coprocesseur qui facilitait la multiplication de grands nombres – le principe sur lequel est fondé l'algorithme RSA. Le schéma décrit l'algorithme de ce coprocesseur. On y distingue un multiplieur de nombres de 8 bits (octet) et deux additionneurs d'octets. Le reste concerne des sélecteurs, pointeurs et des chaînes de registres qui servent à gérer les paramètres de l'ensemble et les adresses des données à traiter. C'est le plus petit coprocesseur jamais conçu. En un seul coup d'horloge, il réalise l'opération $x \times y + a + b$ où chaque élément est un octet. Cette opération est assez magique, car si on imagine que chaque élément est un chiffre décimal entre 0 et 9, le plus grand nombre pouvant être obtenu est $9 \times 9 + 9 + 9 = 99$, soit un nombre de deux chiffres sans report final vers la gauche. Cette opération seule suffit pour réaliser simplement toute multiplication de grands nombres entiers ainsi que leur division avec reste.

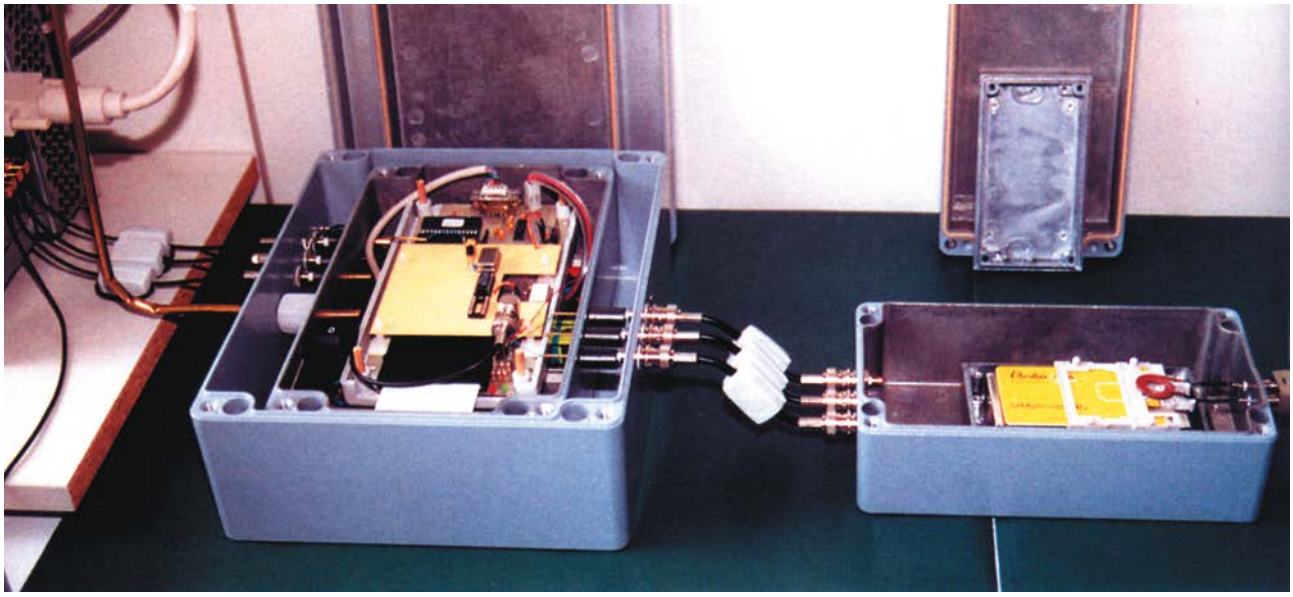
gros par rapport aux capacités des cartes pour y être intégrés, surtout ceux à clé publique. Or les contraintes venant du SCSSI étaient grandes: il fallait éviter le détournement des objets qui utilisaient la cryptographie forte, et la carte à puce ne pouvait donc devenir un réservoir de clés secrètes ou fournir des certificats de sécurité que si elle répondait à ces contraintes. En d'autres termes, ses algorithmes devaient être suffisamment puissants pour empêcher qu'un chiffrement soit facilement associé à un déchiffrement.

La course à l'algorithme

Au début, la carte comportait deux puces, l'une avec le processeur, l'autre avec la mémoire, ce qui offrait *a priori* plus de place pour l'algorithme. Mais on s'était aperçu que la connexion entre les deux, fort visible, permettait d'espionner les échanges processeur-mémoire (en détectant les variations d'intensité du courant consommé, par exemple). La monopuce avait résolu ce problème, mais pas celui de la capacité.

Les premiers algorithmes cryptographiques pour la carte à puce furent donc des algorithmes cryptographiques soit à sens unique, soit inversibles, mais toujours à clé secrète (Telepass 1, TDF, Telepass2, Videopass…). Ces algorithmes étaient bien adaptés à leur contexte restreint en vitesse et en mémoire ainsi qu'aux processeurs 8 bits en usage. Ils utilisaient entre 200 et 300 octets pour leur implémentation complète, ce qui est certainement un exploit. Cependant, l'absence de publication de ces algorithmes en rendait certainement l'usage plus restreint et la diffusion en dehors de la France difficile.

Le vrai départ s'est produit à Eurocrypt 1984, première conférence internationale publique sur la cryptographie, à Paris. Louis Guillou y présenta la carte à puce et l'accès conditionnel. Et lorsqu'on lui demanda pourquoi il n'utilisait pas l'algorithme DES, un algorithme à clés secrètes de 56 bits adopté comme standard en 1976, il répondit: « C'est impossible, car la ROM [mémoire vive] est bien trop petite. » Un défi que je proposai alors à mon équipe de relever. Il y avait en effet au même moment en Belgique un projet de transmission interbancaire sécurisée dénommé TRASEC, dont l'idée était de stocker les clés secrètes des transactions dans une carte à puce sous le contrôle d'un code PIN. Toutefois, l'exécution de l'algorithme DES avec cette clé s'effectuait dans un ordinateur non sécurisé. Intégrer l'algorithme à la carte permettrait d'obtenir



© EISS

AVEC UN DISPOSITIF MODESTE D'ANALYSE DES CARTES À PUCE branché sur un ordinateur, on mesure facilement des temps de calcul ou la consommation électrique, des informations très utiles pour retrouver la clé secrète utilisée dans le cryptosystème RSA des cartes.

cette sécurité manquante. Durant plusieurs mois, nous avons gagné octet par octet, déplaçant les permutations en usage dans le DES et les repliant. Nous avons ainsi réduit l'implantation à un total de 668 octets : chaque octet a coûté environ un jour de travail à une personne…

Le RSA sur une puce

Cependant, pour chacun de nous, le graal cryptographique de la carte à puce n'était pas le DES, mais le RSA. Cet algorithme fondé sur la détermination de deux grands nombres premiers à partir de leur produit était devenu le standard de la cryptographie. L'intégrer à la carte permettrait la signature, la distribution des clés, etc. Paul Barrett, de l'université d'Oxford, en Grande-Bretagne, inventa alors, en 1986, un nouvel algorithme pour le RSA, plus spécifiquement dédié à des processeurs pour traitement du signal. Nous sommes donc partis de cet algorithme. Après quelques tests sur une carte à puce, nous sommes arrivés à près de 100 secondes pour exécuter un algorithme RSA à 512 bits. Et à l'aide de quelques astuces (le stockage des multiples du module public et l'utilisation du théorème des restes chinois, qui permet de résoudre des équations d'arithmétique modulaire), nous avons fourni, toujours pour un module RSA de 512 bits, une signature en moins de 20 secondes. Un beau résultat, mais encore trop lent et trop gourmand en ressources pour être utilisé.

En 1989, Henri Molko, de Philips France et Allemagne, nous proposa un projet audacieux : créer un coprocesseur RSA en symbiose avec un processeur 80C51 pour réaliser une signature RSA sur 512 bits en moins d'une seconde (nous ferons 0,2 seconde). Sans moyens financiers, nous nous sommes lancés dans le projet CORSAIR *(Coprocessor RSA In a Rush)*. En cinq semaines, plusieurs versions du coprocesseur ont été imaginées à partir d'un nouvel algorithme que j'avais trouvé peu avant. Cela conduisit à une première accélération du RSA pour carte à puce par un facteur 500. Un premier prototype en silicium fut obtenu début 1990 : tout fonctionna, sauf l'écriture dans la mémoire morte effaçable de la carte (E^2PROM), « trop » optimisée en aval de notre projet *(voir la figure page 75)*.

Un nouvel algorithme, l'algorithme de Quisquater, avait donc été imaginé. Il se différenciait des autres par le fait que le coprocesseur était indépendant de la longueur de la clé et que l'exécution avait lieu à temps constant de bout en bout, car aucun test n'était effectué en cours de calcul, ce qui, *a priori*, évitait les attaques par mesure du temps d'exécution de l'algorithme.

Une course aux coprocesseurs de deuxième génération fut lancée parmi les fabricants : Fortress, Siemens, Infineon… et Philips. En 1995, Philips élabora un nouveau projet pour reprendre le dessus : FAME, qui nous conduisit à une nouvelle version 500 fois plus rapide que celle de CORSAIR. Son secret ? L'amélioration des circuits électroniques, avec l'ajout d'un multiplieur d'horloge qui augmentait de 16 fois sa fréquence, un multiplieur calculant le produit de nombres codés sur 32 bits (au lieu de 8), un bus (la ligne de transmission de l'information) plus large, une mémoire RAM à double accès très vaste et une meilleure chaîne de traitement au sein du coprocesseur. Ce coprocesseur est toujours utilisé dans des variantes optimisées. Au total, en dix ans, une accélération d'un facteur 250 000 a été obtenue entre la première version logicielle en 8 bits et celle de FAME.

Le développement des grandes applications de la carte à puce coïncida avec la volonté de plusieurs pays de définir une méthodologie pour évaluer la sécurité des systèmes informatiques. Les premiers, les Américains proposèrent à l'Otan, à travers l'*Orange Book*, les critères à respecter pour obtenir un niveau de sécurité donné. Ceux-ci apparaissant vite comme une arme de concurrence efficace pour imposer les produits américains, certains pays européens réagirent en proposant chacun un livre de couleur différente : vert pour les Allemands, jaune pour les Britanniques et… bleu-blanc-rouge pour les Français ! Finalement, la Commission européenne s'empara du sujet et ainsi naquirent les ITSEC (critères d'évaluation de la sécurité de la technologie de l'information) qui permirent aux pays européens de mettre en place leur schéma de certification. Notons que les ITSEC différaient de l'*Orange Book* : celui-ci vérifiait la présence de mécanismes