INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Projet Arénaire

Arithmétique des Ordinateurs

Rhône-Alpes

THÈME 2B

Table des matières

1. Composition de l'équipe

Le projet ARÉNAIRE *est un projet commun au* CNRS*, à l'École Normale Supérieure de Lyon et à l'Inria. Il fait partie du Laboratoire de l'Informatique du Parallélisme (Lip,* UMR CNRS-ÉNS *Lyon-Inria 5668) de l'École Normale Supérieure de Lyon. Ce projet est localisé à Lyon dans les locaux de l'*ÉNS *Lyon.*

Responsable scientifique
Jean-Michel Muller [Directeur de Recherche au CNRS]

Assistante de projet
Sylvie Boyer [Adjointe Technique de Recherche, 20% sur le projet]

Personnel Inria
Nicolas Brisebarre [Délégation sur un poste de Chargé de Recherche, depuis le 01/09/02]
Catherine Daramy [Ingénieur sur un poste d'accueil ODL, depuis le 01/09/02]
Claude-Pierre Jeannerod [Chargé de Recherche]
Nathalie Revol [Chargée de Recherche]
Arnaud Tisserand [Chargé de Recherche]

Personnel Cnrs
Marc Daumas [Chargé de Recherche]
Gilles Villard [Chargé de Recherche]

Personnel Éns Lyon
Florent Dupont de Dinechin [Maître de Conférences à l'ÉNS Lyon]

Chercheur post-doctorant
Jean-Luc Beuchat [Chercheur Post-Doctoral du Fonds National Suisse de la Recherche Scientifique, depuis le 01/11/01]

Chercheurs doctorants
Sylvie Boldo [Allocataire-monitrice AC, deuxième année de thèse]
Nicolas Boullis [Allocataire-moniteur AC, deuxième année de thèse]
David Defour [Allocataire MENRT, troisième année de thèse]
Pascal Giorgi [Allocataire MENRT, deuxième année de thèse]

2. Présentation et objectifs généraux

Mots clés : *arithmétique des ordinateurs*, *circuits VLSI*, *circuits FPGA*, *opérateurs à faible consommation d'énergie*, *calcul entier*, *calcul approché*, *nombres à virgule flottante*, *fonctions élémentaires*, *fiabilité numérique*, *précision multiple*, *arithmétique d'intervalles*, *calcul formel*, *corps finis*, *algèbre linéaire*.

Le projet ARÉNAIRE contribue à l'élaboration et à la consolidation des connaissances dans le domaine de l'arithmétique des ordinateurs. Chaque étude et chaque solution que nous proposons utilise et enrichit ce socle de connaissances tout en dépendant fortement des cibles choisies. On comptera principalement l'implantation matérielle de l'arithmétique (circuits intégrés ou FPGA), le calcul numérique (virgule flottante) ou le calcul très précis voire exact (précision multiple, arithmétique d'intervalles et précision illimitée). Le choix d'une cible fixe souvent la problématique et les contraintes. Fiabilité, précision et rapidité sont les principaux objectifs que l'on retrouve dans les études du projet. Le projet participe à l'amélioration de l'arithmétique disponible pour les utilisateurs, que ce soit au niveau logiciel ou matériel, sur les calculateurs, les processeurs, les circuits dédiés ou enfouis, etc. Calculer mieux ne veut pas uniquement dire calculer plus vite ou plus précisément. Nos études portent aussi par exemple sur la fiabilité de codes numériques dans leur ensemble ou sur la prise en compte de paramètres plus technologiques tels que la consommation d'énergie dans les circuits intégrés.

Historiquement, les fabricants de machines ont toujours eu besoin d'intégrer au niveau matériel ou logiciel les fonctions arithmétiques de base pour une précision « moyenne », c'est-à-dire des mots de 8 à 64 bits. L'addition ou la multiplication ont été très étudiées, mais leurs performances sont encore critiques en surface

(pour la multiplication) ou en vitesse (pour les deux). La division et la racine carrée sont moins critiques, mais il reste probablement plus de possibilités pour de nouvelles améliorations. Pour les fonctions élémentaires (sinus, cosinus, arc-tangente, exponentielle, logarithme, etc.), les concepteurs ont plutôt travaillé jusqu'à ce jour sur la vitesse ou l'économie des ressources. La recherche sur les algorithmes et les architectures pour la multiplication, pour la division et pour le calcul des fonctions élémentaires ou spéciales est toujours très active. Les solutions sont encore très loin d'être figées. Les membres du projet ont acquis une solide réputation dans ce domaine et ont l'intention de poursuivre leurs travaux dans cette voie.

Concevoir un opérateur matériel n'est pas uniquement une œuvre d'assemblage. Il faut intégrer à la démarche de nombreuses données ou contraintes technologiques. Avec l'évolution rapide des technologies, il est nécessaire, par exemple, de bien maîtriser les outils pour le placement et le routage afin de réaliser des circuits performants. La consommation d'énergie d'un circuit intégré dépend, entre autres, de son activité et celle-ci dépend de la valeur de ses entrées. Il faut donc se donner des modèles réalistes sur les données pour faire des simulations pertinentes. Le mode de représentation des nombres est ici très important. Certains codages sont utilisés par les algorithmes performants, d'autres permettent de limiter la consommation du circuit.

Les contraintes pour la conception d'opérateurs arithmétiques sur FPGA sont un peu différentes du fait de la granularité de ces circuits. Par exemple, les blocs logiques de base implantent des petites tables permettant de réaliser des fonctions à quatre entrées. Cette spécificité va permettre d'utiliser la base 4 avec un codage de bas niveau assez efficace. Ici encore, les techniques classiques d'optimisation ne sont pas les bonnes.

Pour se convaincre du bon fonctionnement de l'opérateur réalisé, on effectue des tests, exhaustifs quand cela est possible, ou on construit la preuve du bon fonctionnement de l'opérateur. Ces approches, relativement bien adaptées à la conception d'opérateurs entiers, ne sont plus recommandables pour la validation d'opérateurs à virgule flottante. Le niveau de détail devient tel que seule une approche très rigoureuse de la preuve peut garantir que l'opérateur n'a pas de faille. Cela nous a amenés à utiliser l'assistant de preuve Coq pour la vérification automatique des preuves formelles de bon fonctionnement de nos algorithmes.

Il faut apporter un soin tout particulier à la spécification formelle des opérateurs dans l'assistant pour qu'elle reflète fidèlement la sémantique usuellement associée. Cela n'est pas toujours simple parce que les opérateurs, déjà complexes dans leur fonctionnement « normal », doivent de plus gérer correctement des situations exceptionnelles (division par zéro, etc.). Cependant, la sémantique des opérations est désormais bien définie. En effet, l'arithmétique des ordinateurs a connu un changement majeur en 1985 avec l'apparition de la norme IEEE-754, qui spécifie les formats de représentation des nombres et les opérations arithmétiques en virgule flottante. Cette norme s'est rapidement imposée.

Avec uniquement des opérateurs normalisés à virgule flottante, nous construisons des bibliothèques de calcul en précision étendue et des opérateurs performants (précision, vitesse, etc.) pour le produit scalaire, les fonctions élémentaires et spéciales, qui ne sont pas inclus dans la norme et ne sont donc pas toujours bien implantés. Dans le cadre de l'Action de Recherche Coopérative « AOC » (Arithmétique des Ordinateurs Certifiée) de l'Inria, nous avons travaillé avec des membres des projets LEMME (Nice-Sophia-Antipolis) et SPACES (Lorraine) à la preuve de propriétés sur les nombres à virgule flottante et à la preuve de bon fonctionnement de nos algorithmes (l'ARC est terminée, mais cette collaboration continue). Notre développement d'une spécification formelle de l'arithmétique à virgule flottante en Coq nous permet de valider nos preuves.

Dans le cadre d'un contrat entre l'Inria et ST Microelectronics, nous avons proposé des algorithmes de division par une constante utilisant les opérateurs disponibles sur un circuit DSP construit par ST.

Quelle que soit la cible, matérielle ou logicielle, le choix du système de numération est important. L'exemple caricatural est celui du système logarithmique où l'on représente un nombre par son logarithme en base 2. Dans ce système, la multiplication et la division sont deux opérations faciles et qui n'introduisent pas d'erreur d'arrondi. L'addition est par contre très difficile à réaliser. Un exemple plus classique est celui des systèmes de numération *redondants (carry-save, borrow-save*, etc.) qui sont utilisés à l'intérieur de nombreux multiplieurs et diviseurs. Les entrées comme les sorties des opérateurs sont représentées dans un système usuel, seuls les calculs internes ont recours à ces systèmes redondants. Pour un microprocesseur, l'arithmétique à virgule

flottante semble incontournable, même si on peut certainement en améliorer les implantations existantes, mais sur des systèmes dédiés à des applications particulières, de nombreux modes de représentation des nombres peuvent s'avérer mieux adaptés. Un troisième exemple est celui du calcul en ligne. C'est un calcul où l'opérateur commence à travailler et à produire des chiffres du résultat sans connaître exactement les données, mais seulement les chiffres les plus significatifs. Pour faire du calcul en ligne, nous utilisons une notation redondante en numération de position (en matériel) ou une expansion redondante en nombres à virgule flottante (en logiciel).

La maîtrise des erreurs d'arrondi dans les calculs et, de manière plus générale, la fiabilité numérique des systèmes est un sujet de plus en plus important. On effectue des calculs considérablement plus volumineux que dans les années 1970, alors que les formats de représentation des nombres et, par conséquent, la précision de chaque opération prise individuellement, n'ont presque pas changé. Dans de nombreux domaines, l'imprécision de l'arithmétique flottante peut avoir des conséquences tragiques. Ce constat incite à proposer des arithmétiques alternatives, leur utilisation permettant de cerner, contrôler ou contourner les problèmes numériques.

Au-delà de la virgule flottante - en précision simple ou étendue - on fait appel à d'autres types d'arithmétiques. Nous nous intéressons d'une part, en collaboration avec le projet SPACES (Inria Lorraine), à l'arithmétique d'intervalle en précision arbitraire qui permet d'obtenir des encadrements certifiés et précis de solutions. De tels intervalles fournissent des informations de nature exacte, par exemple quand la question est de majorer ou de minorer le résultat d'un calcul (optimisation globale). Nous nous intéressons d'autre part à des arithmétiques exactes en calcul formel qui permettent de calculer dans des domaines algébriques tels que les corps finis, les entiers à précision illimitée et les polynômes (cryptologie, calculs mathématiques).

À travers le développement d'algorithmes et de bibliothèques de programmes, nous cherchons à appréhender l'impact de ces arithmétiques sur les méthodes de résolution de problèmes. Dans le but de les améliorer, nous voulons cerner leurs possibilités respectives, leurs contraintes associées et leurs limites inhérentes.

3. Fondements scientifiques

3.1. Implantations matérielles de l'arithmétique

3.1.1. Méthodes à base de tables

Lorsqu'une fonction est difficile à calculer mais que son domaine (discret) compte un petit nombre de valeurs, on peut envisager de pré-calculer une fois pour toutes l'ensemble des résultats pour les stocker dans une table. Par la suite, le calcul pourra être remplacé par une lecture de la table. C'est une solution en général rapide mais consommatrice de ressources. Les algorithmes font donc un usage parcimonieux des tables pour trouver le bon compromis entre la taille des tables et la vitesse d'exécution.

À l'un des extrêmes de ce compromis, on trouve quelques algorithmes récents, dus à Matula et Das Sarma[58], Schulte et Stine[73], Wong et Goto[74], Muller[70] et de Dinechin et Tisserand[59], qui calculent des fonctions arbitraires d'un argument en n'utilisant que des tables et des additions. Leur avantage est la vitesse : le temps de calcul est celui de quelques lectures de tables et quelques additions. La recherche sur ce type d'algorithmes est loin d'être terminée et leur pertinence augmente avec l'intégration des circuits.

3.1.2. Opérateurs arithmétiques pour FPGA

Ces quinze dernières années ont vu l'apparition d'un nouveau type de circuit intégré qui peut concurrencer les microprocesseurs dans certaines applications. Il s'agit des réseaux de cellules reconfigurables (*field programmable gate arrays* ou FPGA). Ces composants sont des matrices formées d'un très grand nombre de cellules élémentaires identiques, chaque cellule pouvant être programmée séparément pour se comporter comme quelques portes logiques et quelques bits de mémoire. Les cellules sont reliées par des connexions également configurables dans un ensemble de topologies possibles.

Un composant peut ainsi réaliser un circuit logique arbitraire avec des performances proches de celles qu'aurait un circuit intégré spécifique. Le coût en est cependant bien moindre puisque le FPGA est un circuit

produit en grande série et programmer un FPGA est bien plus simple que concevoir un circuit intégré. On peut reconfigurer le FPGA un nombre quelconque de fois, soit pour corriger ou améliorer sa configuration, soit pour qu'un seul FPGA remplace successivement plusieurs circuits spécifiques au cours de la vie d'une application.

La programmation des FPGA, pour des raisons historiques, s'apparente actuellement plus à de la synthèse de circuits VLSI qu'à de la programmation au sens usuel. Les bibliothèques arithmétiques pour FPGA sont très rudimentaires. Les implantations publiées ou vendues utilisent pour l'essentiel des algorithmes développés pour le VLSI, alors que les métriques de temps et de surface sont très différentes dans les FPGA. De plus, la souplesse supplémentaire des FPGA (reconfigurabilité et adaptabilité à l'application) est rarement utilisée pour optimiser les opérateurs arithmétiques.

De manière plus pratique, les langages utilisés sont également hérités directement du monde du VLSI. Des langages plus adaptés font leur apparition comme JBits[1] chez Xilinx. Un ensemble de classes Java permet l'intégration simple de composants implantés sur des FPGA et de composants logiciels classiques. À plus long terme, on aimerait que la surcharge d'un opérateur ou d'une fonction dans un programme classique permette de déléguer son exécution à un composant spécifique implanté dans un FPGA.

3.1.3. Opérateurs arithmétiques asynchrones

La quasi-totalité des circuits intégrés numériques actuels sont cadencés par un signal d'horloge distribué sur toute la surface du circuit. La mise en œuvre de cette synchronisation globale est de plus en plus complexe. Avec l'augmentation des fréquences et la réduction des technologies, des problèmes comme le routage des signaux d'horloge, les marges de bruit, les pics de courant aux fronts d'horloge, etc. deviennent critiques. Une alternative est l'utilisation de circuits asynchrones. Dans ces circuits, les échanges de données sont cadencés localement par un protocole de communication implanté au plus bas niveau. Il n'y a donc plus d'horloge globale.

Les circuits asynchrones offrent une caractéristique particulièrement intéressante en arithmétique des ordinateurs : c'est la possibilité de calculer en temps moyen. En effet, chaque échange de données étant cadencé localement, il est possible de concevoir des opérateurs dont le temps de calcul effectif est totalement dépendant des valeurs des entrées. On caractérise alors le temps de calcul de l'opérateur par son temps moyen de calcul ainsi que certaines autres caractéristiques statistiques relatives à sa distribution. La notion d'optimisation des cas les plus probables permet d'envisager de nouvelles solutions à tous les niveaux.

Les circuits asynchrones offrent d'autres caractéristiques intéressantes comme la mise en veille naturelle, une meilleure répartition du spectre d'émission électromagnétique, moins de bruits sur les signaux d'alimentation. Enfin, dans les technologies fortement sub-microniques, les délais dans les interconnexions voient leur importance croître, ce qui se traduit par une difficulté accrue dans la conception des horloges. L'asynchronisme n'a pas ce problème.

3.2. Arithmétique à virgule flottante

Un nombre à virgule flottante est représenté par un triplet (s, n, e) associé à la valeur

$$(-1)^s \times n \times \beta^e,$$

où β est la base du système. Dans la pratique, $\beta = 2$ ou 10, mais étudier le système indépendamment de la valeur de β permet de mieux comprendre son fonctionnement. Un opérateur arithmétique manipulant des nombres à virgule flottante est plus complexe que le même opérateur restreint aux seuls nombres entiers. Il faut calculer correctement l'arrondi spécifié par l'utilisateur parmi les quatre proposés par la norme IEEE-754 (norme qui spécifie les formats de représentation des nombres et les opérations arithmétiques en virgule flottante), manipuler à la fois les mantisses et les exposants des opérandes, traiter les divers cas d'exception (infinis, nombres « dénormalisés », etc.).

[1] http://www.xilinx.com/products/jbits/index.htm.

Sur de nombreux processeurs, l'unité de calcul à virgule flottante est plus puissante que l'unité de calcul entier. En effet, il s'agit de la « vitrine » que les constructeurs développent et mettent en avant. Il est ainsi important de bien en maîtriser le fonctionnement pour y intégrer des algorithmes ou pour améliorer les performances des applications. Une bonne connaissance des spécificités des normes actuelles peut permettre d'obtenir de nouveaux développements très efficaces en terme de vitesse ou de précision.

3.2.1. Spécifications et preuves formelles

Des problèmes parfois très médiatisés (bug du Pentium, $2001!/2000! = 1$ sur Maple v7) ont montré que l'arithmétique est parfois difficile à manier ou à implanter sur ordinateur. Peu d'outils permettent de faire des preuves rigoureuses sur des données flottantes. Pourtant, grâce à la norme IEEE-754, les opérations arithmétiques sont complètement spécifiées, ce qui permet de construire des preuves d'algorithmes et de propriétés. Mais il est difficile de présenter synthétiquement une preuve face à l'avalanche de cas particuliers générés par ces calculs. La formalisation des preuves, dans le cadre de notre collaboration avec les projets LEMME et SPACES (ARC AOC), permet d'utiliser un assistant de preuve tel que Coq[64] pour garantir que chaque cas particulier a été envisagé et traité correctement.

Les systèmes tels que Coq permettent de définir de nouveaux objets et de dériver des conséquences formelles de ces définitions. Grâce à la logique d'ordre supérieur, on peut établir des propriétés sous une forme très générale. Par exemple, nous avons utilisé des quantificateurs universels pour établir des propriétés indépendamment de la base d'écriture des nombres à virgule flottante ou pour un mode d'arrondi arbitraire. Les preuves sont construites de façon interactive en guidant l'assistant avec des tactiques de haut niveau. À la fin de chaque preuve, Coq construit un objet interne qui contient tous les détails des dérivations et garantit que le théorème est valide.

3.2.2. Fonctions élémentaires et arrondi correct

De nombreuses bibliothèques de calcul des fonctions élémentaires sont actuellement disponibles. Les fonctions en question sont typiquement celles définies par la norme C99. Des constructeurs, tels Sun, Compaq, HP ou Intel, proposent de telles bibliothèques avec leurs compilateurs lorsque leurs processeurs n'intègrent pas ces fonctions au niveau matériel. Les développeurs de Linux ont eux aussi proposé plusieurs implantations. La plupart de ces bibliothèques s'attachent à respecter les propriétés mathématiques des fonctions implantées : monotonie, symétries, parfois domaine d'arrivée.

Concernant l'arrondi correct du résultat, il n'est pas exigé par la norme IEEE-754 : il a été considéré lors de la rédaction de la norme que l'arrondi correct était impossible à assurer à un coût raisonnable. C'est toutefois l'un des objectifs de nos recherches que de fournir des environnements informatiques où toutes les primitives numériques, y compris les fonctions élémentaires, sont complètement spécifiées.

Actuellement, deux bibliothèques offrent des fonctions élémentaires avec arrondi correct. Celle de Ziv[2] propose uniquement l'arrondi au plus près. La bibliothèque virgule flottante en précision arbitraire MPFR[3], elle, offre les quatre modes d'arrondi de la norme IEEE-754. Ces deux bibliothèques sont comparativement beaucoup plus lentes que les bibliothèques usuelles, ce qui est l'obstacle principal à leur utilisation généralisée.

La cause de cette lenteur est la suivante. Lors de l'évaluation d'une fonction élémentaire, on doit calculer sur une précision interne supérieure à la précision requise pour le résultat afin de savoir décider l'arrondi. La question qui se pose est de savoir quelle est la précision du calcul intermédiaire suffisante pour que l'arrondi de l'approximation coïncide toujours avec l'arrondi du résultat exact. Si l'on n'a pas de réponse à cette question, la bibliothèque doit pouvoir calculer en précision arbitraire en augmentant la précision du calcul jusqu'à savoir arrondir. C'est ce que font les deux bibliothèques précédentes. Si au contraire on connaît la précision nécessaire dans le pire des cas, on peut remplacer les algorithmes en précision arbitraire par des algorithmes en précision certes étendue, mais bornée. Cela permet des optimisations considérables et permet au final d'offir l'arrondi correct avec un surcoût acceptable - et également borné.

[2] http://oss.software.ibm.com/mathlib/.
[3] http://www.mpfr.org.

Des travaux précédents auxquels nous avons participé nous ont donné des bornes sur la précision intermédiaire pour certaines fonctions algébriques. Nous avons également trouvé les « pires cas », c'est-à-dire les valeurs demandant la plus grande précision lors des calculs intermédiaires, pour certaines fonctions élémentaires en arithmétique à virgule flottante en double précision. Ces bornes connues, nous pouvons à présent concevoir des algorithmes de calcul des fonctions spécifiquement optimisés pour cette précision.

L'implantation d'une telle bibliothèque nécessite également l'étude d'algorithmes de calcul en précision étendue (mais pas arbitraire), ainsi que l'étude de méthodes plus générales pour les trois étapes de l'évaluation des fonctions élémentaires : la réduction d'argument, le calcul d'une approximation et la reconstruction du résultat.

3.3. Algorithmes et arithmétiques

Les besoins en arithmétiques des ordinateurs pour le calcul scientifique sont variés et dépassent largement le cadre de l'arithmétique flottante en simple ou multiple précision. Pour des calculs garantis et l'encadrement sûr de solutions on utilise des arithmétiques d'*intervalles*. On manipule des données imprécises issues de mesures physiques, on cherche à optimiser des critères ou à résoudre un ensemble de contraintes... Quand on attend une solution essentiellement exacte, on utilise le calcul formel et ses arithmétiques à *précision « illimitée »* ou *symboliques* (cryptologie, combinatoire, géométrie...).

Les grands systèmes généralistes tels que MatLab ou Maple offrent depuis quelques années la possibilité de « jongler » avec plusieurs de ces modes de calcul. Le fonctionnement de ces systèmes est convaincant et ceux-ci ont un impact substantiel sur la qualité des résultats obtenus. On atteint cependant leurs limites quand il s'agit de développer des modules logiciels hautes-performances facilement réutilisables comme composants d'autres ensembles ou quand il s'agit aussi de tirer parti d'outils externes existants. C'est dans cette démarche de développement de bibliothèque et de réutilisation de code que nous nous inscrivons.

L'utilisation de l'algèbre linéaire s'est généralisée dans de nombreuses branches scientifiques. En particulier dans le domaine du calcul exact, de nombreux progrès reposent de manière cruciale sur la facilité d'emploi, l'efficacité et la réutilisation de fonctionnalités de base sur les matrices. D'un point de vue applicatif, les techniques modernes, par exemple en cryptologie[71], en résolution d'équations polynomiales[69][62] ou en topologie algébrique[60] (formes normales), se fondent sur des représentations matricielles de grandes dimensions et structurées (matrices creuses, faible rang de déplacement, résultants...). Ce sont ces aspects qui nous intéressent principalement. Une facette « plus théorique » concerne la complexité et la recherche de nouveaux algorithmes. Une facette « plus pratique » veut diffuser les acquis algorithmiques et améliorer leurs implications logicielles par le développement de bibliothèques. Dans ce contexte ce sont les arithmétiques dans des corps finis, des arithmétiques de nombres rationnels ou algébriques et des arithmétiques de polynômes qui sont mises en jeu.

Nos collaborations principales sont avec les projets Inria APACHE (développement C++), SPACES (arithmétique multiprécision) et des contacts avec GALAAD (algèbre linéaire). Au plan international nous participons à l'action LinBox. Tout en menant des recherches algorithmiques dans les domaines privilégiés que sont l'*algèbre linéaire* exacte et l'*optimisation globale* avec ou sans contraintes, nous nous intéressons à fournir des fonctionnalités arithmétiques spécifiques (mixtes intervalles/multiprécision, corps finis...), à l'utilisation et à l'impact d'arithmétiques de diverses provenances.

3.3.1. Algorithmes numériques en arithmétique d'intervalles en précision arbitraire

On atteint maintenant les limites de validité théorique de certains algorithmes numériques. Pour obtenir des résultats fiables, on peut calculer avec une arithmétique par intervalles. De nouveaux problèmes peuvent être résolus avec cette arithmétique qui calcule sur des ensembles et non pas sur des nombres, en particulier le problème de l'optimisation globale d'une fonction continue, qui est l'application que nous visons. Cependant les encadrements fournis peuvent être trop larges pour que l'information calculée soit pertinente. Une solution pour obvier à ce problème consiste à augmenter la précision des calculs.

Le développement d'une bibliothèque efficace pour l'arithmétique d'intervalles en précision arbitraire, ainsi que d'une bibliothèque d'algorithmes basés sur cette arithmétique, est l'un des volets de nos travaux.

Essayer et comparer les algorithmes existants et éventuellement proposer de nouveaux algorithmes tirant parti de l'arithmétique par intervalles multiprécision constitue l'autre volet de nos travaux sur l'arithmétique d'intervalles.

3.3.2. Algorithmes de calcul pour l'algèbre linéaire formelle

Les méthodes exactes en algèbre linéaire sont en nette évolution depuis quelques années. Cela s'illustre par la complexité des problèmes de base qui évolue encore : celle du calcul du polynôme caractéristique sur un corps abstrait K ou sur un corps fini pour des matrices creuses ; celle de la résolution de systèmes ou du calcul du déterminant sur $\mathbb{Q}$; celle des formes normales d'Hermite ou de Smith sur $\mathbb{Z}$ ou $\mathsf{K}[x]$. Nous travaillons sur ces questions.

Au-delà des avancées théoriques, un objectif est d'arriver aussi à de meilleurs résultats en pratique. Ces études s'appuient notamment sur des algorithmes récursifs souvent probabilistes et par blocs, sur la technique des *baby steps / giant steps* ainsi que sur la méthode de Lanczos-Wiedemann et ses variantes.

3.3.3. Arithmétiques accessibles depuis LinBox

L'action internationale LinBox[4] (voir §6.16 et §6.17) a été démarrée par E. Kaltofen (Université de Caroline du Nord), B.D. Saunders (Université du Delaware) et G. Villard. Elle est le pendant logiciel des études du paragraphe précédent. L'objectif est l'écriture d'une bibliothèque générique, basée sur le « branchement » à la demande (*plug-and-play*) de composants externes (*plug-in*).

Cette bibliothèque (cf. §5.9) est spécialisée pour l'algèbre linéaire exacte creuse et structurée. Le langage C++ est utilisé pour le noyau de la bibliothèque. Les arguments ayant conduit à ce choix sont les avantages du langage (héritage, généricité, diffusion, efficacité) ainsi que les expériences respectives des équipes participantes avec leurs précédents prototypes de bibliothèques (Givaro[5] pour ce qui nous concerne).

Les performances de la bibliothèque sont particulièrement sensibles aux arithmétiques sous-jacentes. Pour les corps finis et leurs extensions algébriques ainsi que pour les entiers et les polynômes, nous expérimentons plusieurs bibliothèques (y compris certaines qui nous sont propres) et poursuivons le développement.

3.3.4. Arithmétique pour la cryptographie

Le développement de cryptosystèmes sûrs et rapides demande souvent l'implantation matérielle ou logicielle d'arithmétiques spéciales, soit dans des ensembles de la forme $\mathbb{Z}/p\mathbb{Z}$ (RSA), soit dans des corps finis généraux. En collaboration avec des équipes expertes en algorithmes de cryptographie, nous travaillons plus particulièrement à l'évaluation de solutions à base de circuits FPGA.

4. Domaines d'application

Mots clés : *opérateur arithmétique*, *matériel*, *circuit dédié*, *chiffrement*, *stabilisation*, *contrôle*, *validation*, *preuve*, *logiciel numérique*.

Notre savoir-faire intéresse les domaines applicatifs pour lesquels la qualité - telle que l'efficacité ou la sûreté - des opérateurs arithmétiques utilisés intervient. Cela concerne des développements orientés vers le matériel, tels que la conception de primitives arithmétiques spécialement optimisées pour l'application visée et le support utilisé (cf. §7.2). Sont également impliqués des aspects de fiabilité numérique des logiciels, comme l'amélioration de la stabilité numérique de certains algorithmes, le calcul de résultats garantis, sous forme de valeurs exactes ou d'encadrements garantis, ou enfin la certification de programmes numériques.

Schématiquement, nos compétences se déclinent en quatre axes avec des domaines d'application qui se croisent.

Aide à la réalisation de circuits DSP. Notre expertise en algorithmes arithmétiques nous permet de conseiller des industriels, tant en ce qui concerne la réalisation d'opérateurs matériels (par exemple

[4] http://www.linalg.org.
[5] http://www-apache.imag.fr/software/givaro.

nous avons développé un multiplieur avec ST Microelectronics) qu'en ce qui concerne le logiciel de base (travail mené sur la division par une constante, intégré dans le compilateur du ST 100, cf. §7.1).

Circuits arithmétiques matériels dédiés. Notre travail en cours (cf. §8.1.1) en cryptographie porte sur l'efficacité dans les protocoles à clé publique et a pour objectif de réaliser des systèmes de cryptage très rapides. Cela permettra de déterminer quelles sont les opérations que l'on sait exécuter efficacement et par conséquent de guider les développements et choix de nouveaux algorithmes de chiffrement (ce qui n'est pas de notre ressort).

Théorie du contrôle. Les domaines visés sont essentiellement la théorie du contrôle et le traitement du signal. Dans les deux cas, l'algèbre polynomiale (et même polynomiale matricielle) est le formalisme de prédilection et pourtant les données sont la plupart du temps connues de façon inexacte. La stabilisation d'algorithmes numériques construits à partir de versions conçues pour le calcul exact est donc une étape incontournable de la mise au point d'outils logiciels pour les chercheurs de ces domaines.

Validation et preuve de logiciels numériques. L'arithmétique par intervalles est utilisée pour des applications critiques, ou *a posteriori* pour analyser la qualité numérique de codes critiques (suivi de trajectoire en physique des particules, simulation d'accident d'une centrale nucléaire). Cependant, un remplacement automatique des données numériques par des intervalles ne suffit pas : l'obtention de résultats pertinents requiert une analyse du programme et une utilisation appropriée de cette arithmétique.

Nous proposons d'autre part des preuves validées en virgule flottante. Nous nous restreignons pour l'instant à de petits programmes pour ne pas envisager trop tôt des systèmes automatiques. Là est l'objectif de nos contacts avec le futur National Institute of Aerospace avec une application d'une dizaine de lignes de code. Un rapprochement avec l'Université Lyon 1 est en cours sur des applications médicales. Dans tous les cas, les programmes doivent être suffisamment critiques pour justifier le travail de validation des preuves.

Nous proposons aussi des méthodes de résolution à base d'arithmétique exacte pour les applications qui ne peuvent bénéficier des validations précédentes.

5. Logiciels

5.1. Description VHDL d'opérateurs arithmétiques pour FPGA Virtex-II

Participants : Jean-Luc Beuchat [correspondant], Arnaud Tisserand.

Mots clés : *opérateurs arithmétiques*, *description VHDL*, *FPGA Virtex-II*.

J.-L. Beuchat et A. Tisserand ont développé des outils générant des descriptions VHDL de multiplieurs (opérandes de plus de 20 bits) et de diviseurs SRT non signés basés sur les multiplieurs 18 bits des circuits de la famille Virtex-II de Xilinx. Ce logiciel ne sera pas distribué, mais les descriptions VHDL générées seront mises à disposition via une page Web.

5.2. Opérateurs arithmétiques pour FPGA

Participants : Jérémie Detrey, Florent de Dinechin [correspondant].

Mots clés : *coeurs arithmétiques*, *opérateurs arithmétiques*, *FPGA*, *JBits*, *Virtex*.

F. de Dinechin et J. Detrey ont écrit en JBits les composants arithmétiques suivants pour FPGA Xilinx Virtex : multiplication par une constante, évaluation de fonctions arbitraires en précision de 8 à 24 bits.

Son statut actuel est celui d'un prototype, distribué sous licence Gnu à l'adresse
http://www.ens-lyon.fr/LIP/Arenaire/.

5.3. Bibliothèque multiprécision à retenue conservée logicielle

Participants : Catherine Daramy, David Defour [correspondant], Florent de Dinechin.

Mots clés : *multiprécision sur quelques centaines de bits*, *bibliothèque C*.

La bibliothèque SCS *(Software Carry Save)* a été développée spécifiquement pour répondre aux besoins de précision étendue que présente la bibliothèque de fonctions élémentaires avec arrondi correct actuellement en cours de développement (cf. §5.4). Ces besoins sont : précision de quelques centaines de bits, portabilité, performance comparable ou meilleure que GMP, légèreté. Elle utilise une structure de donnée qui permet de différer les propagations de retenues lors des additions et multiplications multiprécision *(retenue conservée logicielle)*. Cette structure gaspille un peu d'espace mais permet un code plus simple et efficace tout en restant portable (C ANSI standard), et offrant plus de parallélisme intrinsèque que les actuels processeurs superscalaires savent bien exploiter.

Cette bibliothèque est pour le moment à l'état de prototype, distribué sous licence Gnu à l'adresse

http://www.ens-lyon.fr/LIP/Arenaire/.

Seules sont implémentées l'addition et la multiplication (éventuellement fusionnées) et les conversions vers/depuis les flottants.

5.4. Bibliothèque de fonctions élémentaires avec arrondi correct

Participants : Catherine Daramy, David Defour [correspondant], Florent de Dinechin, Jean-Michel Muller.

Mots clés : *bibliothèque de fonctions élémentaires*, *libm*, *double précision*, *arrondi correct*.

Cette bibliothèque est réalisée dans le cadre d'une ODL de l'Inria, qui a permis l'accueil de C. Daramy.

L'objectif de cette bibliothèque est d'offrir

- toutes les fonctions élémentaires spécifiées par la norme C99 (fonctions trigonométriques et hyperboliques avec leurs inverses, fonctions exponentielles et logarithmes...),
- l'arrondi correct du résultat en double précision,
- le choix du mode d'arrondi comme dans la norme IEEE-754,
- une performance dans un facteur deux comparée à une bibliothèque standard,
- la portabilité sur différents processeurs.

Statut : en développement. Un prototype sera terminé fin 2003. Le langage utilisé est le C.

5.5. Bibliothèque de propriétés de l'arithmétique à virgule flottante

Participants : Sylvie Boldo [correspondante], Marc Daumas.

Mots clés : *virgule flottante*, *preuve d'algorithme*, *méthode formelle*, *Coq*.

Notre bibliothèque de preuves et de propriétés est basée sur la bibliothèque créée par l'ARC AOC et distribuée par L. Théry. Les théorèmes sont présentés sous la forme la plus générale possible, par exemple indépendamment de la base ou du mode d'arrondi. La spécification suit un chemin de preuve passant par des résultats intermédiaires connus et documentés dans la littérature. Cette approche nous garantit qu'un lecteur extérieur à notre développement pourra en extraire du savoir sans devoir d'abord se familiariser avec tout notre formalisme.

S. Boldo maintient à jour sur Internet la base de données de faits prouvés avec les scripts de preuve à l'adresse

http://www.ens-lyon.fr/~sboldo/coq/

Nous avons ainsi travaillé avec P. Markstein (HP) pour valider en Coq des propriétés qu'il a identifiées comme clés dans la validation de la bibliothèque de calcul de l'IA-64 développée à HP.

5.6. Évaluation fidèle d'un polynôme à partir de la méthode de Horner

Participants : Sylvie Boldo, Marc Daumas [correspondant].

Mots clés : *virgule flottante*, *erreur d'arrondi*, *méthode formelle*, *Coq*, *Maple*.

Ce programme Maple disponible sur Internet à l'adresse http://www.ens-lyon.fr/~daumas/SoftArith/index.html.fr est basé sur les travaux en Coq présenté au §6.13. Il est distribué avec un exemple dû à Fike en 1967 et un exemple tiré de la thèse de C. Moreau-Finot. Le programme garantit que les polynômes proposés sont des implantations fidèles de la fonction exponentielle.

5.7. Bibliothèque d'arithmétique par intervalles à précision multiple

Ce développement de logiciels s'effectue en étroite collaboration avec les membres du projet SPACES*, Inria Lorraine.*
Participants : Nathalie Revol [correspondante], Fabrice Rouillier [SPACES].

Mots clés : *bibliothèque C*, *arithmétique par intervalles*, *précision arbitraire*, *MPFI*.

MPFI (*Multiple Precision Floating-point Interval arithmetic library*) est une bibliothèque d'arithmétique par intervalles en précision arbitraire basée sur MPFR[57]. Elle implante les opérations arithmétiques et les fonctions élémentaires avec pour arguments des intervalles représentés avec une précision arbitraire et retourne les résultats les plus précis possibles. On peut la télécharger à l'adresse http://www.ens-lyon.fr/~nrevol/nr_software.html. On la trouve également sur le CD des logiciels libres de l'Inria, ainsi que sur le CD distribué lors de l'école d'été sur les Outils de Calcul Symbolique Numérique Collaboratif (Giens, septembre 2002). Cette bibliothèque est écrite en C et est co-développée avec F. Rouillier

5.8. Bibliothèque pour l'arithmétique des polynômes approchés

Cette bibliothèque a été écrite en collaboration avec G. Labahn (Symbolic Computation Group, Université de Waterloo, Ontario, Canada).
Participant : Claude-Pierre Jeannerod [correspondant].

Mots clés : *bibliothèque Maple*, *algorithmes semi-numériques*, *primalité et PGCD approché*.

Snap (*Symbolic-Numeric Algorithms for Polynomials*) est une bibliothèque Maple pour l'arithmétique des polynômes univariés dont les coefficients ne sont connus qu'à une précision finie donnée. Elle permet notamment de tester la primalité de deux tels polynômes et, le cas échéant, d'estimer la distance à l'ensemble des paires ayant un PGCD non trivial. Plusieurs types de PGCD approché (epsilon-PGCD, PGCD de Schönhage) peuvent également être calculés. Les algorithmes implantés font largement appel à l'algèbre linéaire structurée et ont une stabilité numérique établie. Cette bibliothèque est utilisable comme module de la version 8 du logiciel Maple (http://www.maplesoft.com). Le guide d'utilisation est disponible à l'adresse http://www.ens-lyon.fr/~cpjeanne.

5.9. Bibliothèque d'algèbre linéaire

Ce logiciel est développé dans le cadre d'une action internationale entre le Canada, les États-Unis et la France (cf. §8.3.1).
Participants : Pascal Giorgi, Claude-Pierre Jeannerod, Gilles Villard [correspondant].

Mots clés : *bibliothèque générique C++*, *calcul matriciel*, *algèbre linéaire*, *calcul exact*, *corps finis*.

LinBox est une bibliothèque C++ générique dédiée au calcul exact sur des matrices denses, creuses ou structurées. Avec les notions d'interface standard et d'adaptateur, elle est orientée sur l'utilisation de composants externes et la connexion avec des logiciels généralistes de type Maple. Elle importe et implante diverses arithmétiques sur les corps finis et les entiers en précision infinie. Ses fonctionnalités concernent le calcul du déterminant, du noyau, la résolution de systèmes linéaires (solvabilité, systèmes diophantiens...) et le calcul de formes normales (cf. §6.16 et §6.17).

La première distribution date d'août 2002 et peut être téléchargée à l'adresse :
http://www.linalg.org.

6. Résultats nouveaux

6.1. Introduction

Les résultats obtenus dans le projet ARÉNAIRE sont présentés dans l'ordre suivant : tout d'abord ce qui concerne le point de vue du matériel, à l'exclusion des fonctions élémentaires, est développé dans les §6.2, §6.3, §6.4 et §6.5, ensuite tous les travaux sur les fonctions élémentaires, aussi bien matériels que logiciels, sont regroupés dans les §6.6, §6.7, §6.8, §6.9 et §6.10, puis les résultats qui portent sur les aspects logiciels et qui n'ont pas été abordés se trouvent en fin de partie dans les §6.11, §6.12, §6.13, §6.14, §6.15, §6.16 et §6.17.

6.2. Opérateurs arithmétiques pour FPGA

Participants : Jean-Luc Beuchat, Florent de Dinechin, Jérémie Detrey, Arnaud Tisserand.

Mots clés : *opérateurs arithmétiques*, *opérateurs modulaires*, *FPGA*.

Des générateurs d'opérateurs arithmétiques optimisés ont été étudiés et développés pour la multiplication et la division entières, l'addition et la multiplication modulaires et les fonctions élémentaires.

J.-L. Beuchat et A. Tisserand ont développé des outils générant des descriptions VHDL de multiplieurs (opérandes de plus de 20 bits) et de diviseurs non signés basés sur les multiplieurs 18 bits des circuits de la famille Virtex-II de Xilinx [19]. Ces outils s'avèrent plus efficaces que les logiciels commerciaux. J.-L. Beuchat a été contacté par des représentants de Synplicity, Altera et ST Microelectronics intéressés par ce travail. À l'exception d'un groupe qui utilise les petits multiplieurs pour implanter des décalages, J.-L. Beuchat et A. Tisserand étaient les seuls à proposer des opérateurs arithmétiques exploitant efficacement ces nouveaux éléments de base des FPGA (voir les actes de la conférence FPL'02[63]).

J.-L. Beuchat a ensuite étudié l'implantation d'additionneurs et de multiplieurs modulaires sur FPGA. L'arithmétique modulaire intervenant par exemple dans le système de représentation RNS (*Residue Number System*) ou en cryptographie, il est important de disposer de briques de base performantes pour ce type de calculs. Afin de comparer aisément diverses architectures en fonction de la taille des opérandes, J.-L. Beuchat a à nouveau développé des générateurs de code VHDL paramétrables [39].

F. de Dinechin et J. Detrey ont implanté, dans le système JBits, un générateur d'opérateurs pour les fonctions élémentaires utilisant la méthode multipartite [25]. Ce générateur permet par exemple d'obtenir des sinus rapides jusqu'à 20 bits de précision, alors que les bibliothèques constructeur ne vont que jusqu'à 10 bits.

6.3. Multiplication par des constantes

Participants : Nicolas Boullis, Arnaud Tisserand.

Mots clés : *multiplication par des constantes*, *opérateurs arithmétiques matériels*, *VHDL.*

N. Boullis et A. Tisserand étudient les opérateurs matériels optimisés pour la multiplication d'un vecteur par une matrice constante. Ils développent un générateur automatique pour de tels opérateurs.

N. Boullis et A. Tisserand travaillent à la génération automatique d'opérateurs matériels optimisés pour des applications linéaires. Ces applications linéaires peuvent s'écrire sous la forme d'une multiplication d'un vecteur par une matrice constante. De telles opérations sont très fréquentes dans les domaines du traitement du signal et du multimédia, comme par exemple pour la transformée de Fourier, la DCT ou des filtres numériques.

Leurs travaux se basent sur un algorithme proposé par V. Lefèvre [66], qui gère la multiplication d'un nombre par une ou plusieurs constantes. L'algorithme a été étendu pour gérer la multiplication d'un vecteur par une matrice constante et améliorer l'opérateur généré en terme de vitesse d'exécution, ainsi qu'en termes de rapidité et de surface.

Ils ont développé un prototype de générateur automatique pour de tels opérateurs [22]. Ce prototype donne déjà des résultats au moins aussi bons que ceux que l'on trouve dans la littérature. Une version étendue a aussi été soumise à la conférence Arith'16.

6.4. Cryptographie

Participants : Jean-Luc Beuchat, Pascal Giorgi, Arnaud Tisserand, Gilles Villard.

Mots clés : *opérateurs arithmétiques pour la cryptographie*.

L'implantation sur FPGA de RC6 et IDEA, deux algorithmes de cryptage à clé privée a été étudiée. Les implantations logicielles des algorithmes de chiffrement ne permettent généralement pas de chiffrer des données en temps réel et il est nécessaire d'étudier des processeurs spécialisés dans ce domaine. Les circuits FPGA sont très intéressants pour ce type d'application : d'ici quelques années, les cartes réseau des ordinateurs contiendront probablement un processeur chargé de chiffrer/déchiffrer les données. La reconfiguration permettra par exemple de passer d'un système à clé privée à un système à clé publique.

Nous avons réalisé des processeurs implantant les algorithmes à clé privée IDEA[65] et RC6[72] [37]. Le choix d'un algorithme de calcul dépend de la famille de FPGA sur laquelle sera implanté le processeur. Des générateurs de code VHDL proposant divers algorithmes à l'utilisateur ont donc été développés. Les circuits ainsi obtenus sont les plus performants actuellement disponibles. Ces travaux sur RC6 et IDEA ont été respectivement soumis à la revue IEEE *Transactions on Very Large Scale Integration Systems* et à la conférence Ersa 2003.

6.5. Intégration d'opérateurs arithmétiques en Alpha/HandelC

Ce travail est le fruit d'une collaboration avec T. Risset, du projet COMPSYS*, Inria Rhône-Alpes, et J.M. Spivey et M. Manjunathaiah, du Computing Laboratory, Université d'Oxford.*

Participants : Florent de Dinechin, Tanguy Risset [Compsys], John M. Spivey [OUCL], Muniyappa Manjunathaiah [OUCL].

Mots clés : *conception rapide pour FPGA*, *Alpha*, *HandelC*, *opérateurs arithmétiques*.

F. de Dinechin a développé en Alpha des opérateurs arithmétiques pour étudier et illustrer l'intégration des langages Alpha et HandelC sur une application réelle. F. de Dinechin a collaboré avec T. Risset, J.M. Spivey et M. Manjunathaiah sur la synthèse rapide de fonctions pour les FPGA. L'objectif est d'interfacer deux langages offrant des modèles de parallélisme complémentaires :

- le modèle data-parallèle pour les cœurs de calcul intensif grâce à Alpha, développé par le projet COMPSYS, et
- le modèle CSP *(communicating sequential processes)* pour les parties à contrôle intensif grâce à Handel, développé à l'OUCL.

L'exemple choisi était le filtre à réponse impulsionnelle finie *(FIR)*, pour lequel F. de Dinechin a exprimé en Alpha les opérateurs arithmétiques nécessaires (addition et multiplication) [26].

6.6. Évaluation en ligne de fonctions élémentaires sur FPGA

Participants : Jean-Luc Beuchat, Arnaud Tisserand.

Mots clés : *arithmétique en ligne*, *fonctions élémentaires*, *FPGA*, *tables*.

J.-L. Beuchat et A. Tisserand ont développé des générateurs d'opérateurs optimisés de multiplication et de division entières pour les circuits FPGA de la famille Virtex-E. L'arithmétique en ligne est une arithmétique sérielle dans laquelle toutes les opérations s'effectuent chiffre de poids fort en tête grâce à un système de numération redondant limitant les propagations de retenue dans l'addition[61]. S'il existe des méthodes de conception d'opérateurs calculant des fonctions arithmétiques et algébriques, l'évaluation des fonctions élémentaires n'avait pas encore été étudiée jusqu'au niveau de l'implantation matérielle.

J.-L. Beuchat et A. Tisserand ont proposé et implanté sur FPGA une architecture exploitant des petites tables et des évaluations polynomiales pour calculer des fonctions élémentaires [18]. Ils ont complété ce travail en améliorant l'outil de génération des polynômes et en étudiant plus rigoureusement la précision des opérateurs obtenus [9].

6.7. Évaluation de fonctions à l'aide de tables

Participants : David Defour, Florent Dupont de Dinechin, Jean-Michel Muller.

Mots clés : *évaluation matérielle de fonctions*, *FPGA*, *tables*, *coefficients tronqués*.

D. Defour, F. de Dinechin et J.-M. Muller ont défini et évalué une nouvelle méthode d'évaluation matérielle des fonctions élémentaires utilisant des tables et des petits multiplieurs. L'erreur liée à l'utilisation de coefficients tronqués pour les polynômes d'approximation a également été réduite. Depuis l'invention de la méthode des tables bipartites par Matula et DasSarma en 1995, de nombreuses équipes s'intéressent à des méthodes « hybrides » (lecture dans une table suivie d'un tout petit nombre d'opérations arithmétiques) d'évaluation de fonctions. D. Defour, F. de Dinechin et J.-M. Muller ont présenté une nouvelle méthode pour l'évaluation matérielle des fonctions élémentaires pour des petites précisions (jusqu'à 30 bits) [23]. La nouveauté est d'utiliser de petits multiplieurs, ce qui est très adapté à une implantation dans les FPGA modernes qui en sont pourvus.

Par la suite, J.-M. Muller s'est concentré sur le problème suivant. Les coefficients d'approximation polynomiale utilisés dans ce type de méthode sont nécessairement tronqués. Il est clair que le meilleur approximant parmi les « approximants tronqués » n'est pas forcément obtenu en « tronquant » le meilleur approximant. Une méthode permettant de partiellement compenser la perte de précision due à la troncature est proposée [52].

6.8. Fonctions élémentaires avec arrondi correct

Participants : Catherine Daramy, David Defour, Florent de Dinechin, Jean-Michel Muller.

Mots clés : *fonctions élémentaires*, *double précision*, *arrondi correct*.

D. Defour, F. de Dinechin, C. Daramy et J.-M. Muller développent une bibliothèque portable et performante de calcul des fonctions élémentaires en double précision avec arrondi correct. Ce travail a requis la conception d'une bibliothèque de calcul en précision étendue adaptée, ainsi qu'une étude sur l'impact de la mémoire cache sur la performance des algorithmes de calcul des fonctions élémentaires.

L'arrondi correct d'une fonction élémentaire s'obtient en deux étapes. Dans cette bibliothèque, la première étape est similaire à ce qui est fait dans une bibliothèque usuelle et est donc d'une vitesse comparable. Sa spécificité est de savoir détecter les cas pour lesquels la précision n'est pas suffisante pour assurer l'arrondi correct.

Pour ces cas (de l'ordre d'un sur quelques milliers), la seconde étape est lancée, qui réalise le calcul avec une précision garantissant l'arrondi correct. La précision requise pour la seconde étape a été déterminée par le projet ARÉNAIRE pour un certain nombre de fonctions usuelles sur leurs intervalles les plus utiles, ce qui permet de borner le temps de calcul dans ces cas-ci : ce temps de calcul est au pire celui de la seconde étape, ce qui explique que nous nous attachions à l'optimiser (voir plus bas).

Lorsque la précision nécessaire à l'arrondi correct n'est pas connue, il faut augmenter la précision du calcul jusqu'à permettre l'arrondi correct. Dans ce cas, on ne sait pas borner le temps de calcul, c'est pourquoi nous continuons nos recherches pour déterminer les précisions maximales nécessaires.

D. Defour et F. de Dinechin ont étudié un format de représentation des nombres en multiprécision bien adapté à cette utilisation. Ce format utilise des mots machine en réservant des bits pour les calculs intermédiaires. L'idée est de différer les propagations de retenues lors des additions et multiplications multiprécision *(retenue conservée logicielle)*. Cette structure gaspille un peu d'espace mais permet un code plus simple et offrant plus de parallélisme intrinsèque que l'approche de GMP, ce que les actuels processeurs superscalaires savent bien exploiter. Ceci a été vérifié sur différentes architectures de processeurs [24].

De plus, D. Defour a étudié l'impact de l'utilisation des caches de données des processeurs sur le temps de calcul d'une fonction élémentaire [47].

6.9. Travail sur la spécification de l'arithmétique flottante

Ce travail est mené en collaboration avec le projet SPACES*, Inria Lorraine.*

Participants : David Defour, Jean-Michel Muller, Nathalie Revol.

Mots clés : *fonction élémentaire*, *norme IEEE-754*, *arrondi correct*.

La norme IEEE-754 pour l'arithmétique flottante ne propose actuellement aucune spécification pour les fonctions élémentaires : une telle spécification est proposée et argumentée. Une des conséquences de nos travaux des années précédentes sur le dilemme du fabricant de tables est qu'il est maintenant envisageable de proposer une spécification des fonctions élémentaires en virgule flottante. Les grandes lignes d'une telle spécification ont été proposées dans [31]. Elle consiste à exiger l'arrondi correct des fonctions élémentaires, ou, au minimum, une qualité garantie : erreur limitée, respect des propriétés mathématiques des fonctions calculées.

6.10. Test de la qualité des fonctions élémentaires sur ordinateur

Ce travail s'effectue en collaboration avec le projet SPACES*, Inria Lorraine.*
Participants : Nathalie Revol, Paul Zimmermann [SPACES].

Mots clés : *bibliothèque mathématique*, *fonction élémentaire*, *qualité*, *arrondi*.

Il s'agit de développer un outil de test des fonctions élémentaires en arithmétique flottante : erreur, direction des arrondis, respect des propriétés mathématiques des fonctions. Le point de départ de ce travail est la collaboration avec HP-Intel sur le processeur IA-64 : HP-Intel nous a fourni deux IA-64 pour que, entre autres, nous testions la qualité des fonctions élémentaires développées pour ce processeur. Dans un premier temps, nous avons testé la drection des arrondis et la précision de ces fonctions : sur des échantillons de nombres aléatoires (typiquement de 1 à 10 millions), nous avons mesuré les erreurs moyenne et maximale entre la valeur exacte et la valeur calculée, pour les fonctions mathématiques usuelles (log, exp, sin, atan, cosh, asinh...), pour les quatre modes d'arrondi et pour la double précision. Nous avons ensuite étendu ces tests pour prendre en compte les différentes précisions (double étendue et quadruple) et d'autres critères, à savoir le respect de l'ensemble d'arrivée ($]-\pi/2, \pi/2[$ pour l'arc-tangente par exemple), la monotonie et la parité de la fonction s'il y a lieu.

6.11. Propriétés de diverses implantations de la virgule flottante

Participants : Sylvie Boldo, Marc Daumas.

Mots clés : *virgule flottante*, *erreur d'arrondi*, *méthode formelle*, *Coq*.

S. Boldo et M. Daumas valident des preuves formelles de propriétés de l'arithmétique à virgule flottante. Ces travaux sont possibles grâce à leur transcription générique en Coq spécialisable à la norme IEEE-754 qui continue à évoluer.

> « I approve of what you're doing. I regard it as necessary. My hope is that continuing in that direction will ultimately lead to specifications and implementations which are not merely verifiable, but also maintainable. »
>
> W. Kahan, ACM Turing Award, 1989

Des propriétés sont déjà disponibles dans la littérature sur la valeur exacte de l'erreur d'arrondi pour les opérateurs normalisés IEEE-754 d'addition, de multiplication, de division et de racine carrée. Cependant tous les travaux précédents ont supposé sans étude approfondie qu'il n'y avait pas de dépassement de capacité vers les nombres de très petit exposant (*underflow*). S. Boldo et M. Daumas ont montré que l'hypothèse d'absence d'*underflow* est trop contraignante. Ils ont énoncé une nouvelle condition nécessaire et suffisante pour que l'on puisse représenter la valeur exacte de l'erreur d'arrondi. Ils ont ensuite présenté des exemples de comportements surprenants quand cette condition n'est pas respectée. Il s'agit d'une étude pour laquelle un outil de validation de preuve semble s'imposer. Ces travaux ont été proposés à la conférence Arith [42].

Le théorème de Sterbenz est souvent utilisé pour prouver la correction d'algorithmes de l'arithmétique normalisée IEEE-754. S. Boldo et M. Daumas ont prouvé que ce théorème est toujours vrai en l'absence d'*underflow* sur le TMS 320 de Texas Instrument [21]. Il s'agit d'un circuit de traitement numérique du signal

utilisé pour des applications militaires et avioniques (référence SMJ), qui ne respecte pas la norme IEEE-754. Ils ont aussi prouvé plusieurs propriétés de la représentation des nombres utilisée par le TMS 320. Pour cette étude aussi, un outil de validation de preuve semble s'imposer. Ces travaux seront proposés à la revue Software Tools for Technology Transfer.

Cette expertise en matière de preuve validée pour l'arithmétique à virgule flottante a été reconnue par le groupe de travail chargé de la révision de la norme IEEE-754 sur l'implantation de l'arithmétique à virgule flottante (voir la citation au début de ce paragraphe). En plus des propriétés ponctuelles en cours de développement citées plus haut, S. Boldo et M. Daumas restent en contact avec plusieurs membres de ce groupe pour faire évoluer leur spécification et pour présenter leur point de vue particulier sur les outils de validation des preuves tels que Coq.

6.12. Division en virgule flottante

Participants : Nicolas Brisebarre, Jean-Michel Muller.

Mots clés : *division*, *division par une constante*, *multiplieurs-accumulateurs*.

L'utilisation de l'opération multiplication-accumulation permet de construire de nouveaux algorithmes de division flottante. Une accélération d'un tel algorithme et la détermination des opérandes qui réalisent l'erreur maximale ont été conduites. La spécification des quatre opérations arithmétiques (présente dans la norme IEEE-754 de 1985) ainsi que la spécification du *fused Mac* (qui sera présente dans la prochaine norme) permettent de construire des algorithmes utilisant ces diverses opérations et de prouver leur comportement. Une conséquence intéressante en est un algorithme de division, proposé par Cornea, Markstein et Golliver, basé sur l'itération de Newton, et qui permet d'effectuer des divisions avec arrondi correct sur une architecture avec jeu d'instructions IA-64, en n'utilisant que des *fused Mac*. J.-M. Muller s'est intéressé à l'accélération de cet algorithme dans des cas particuliers [53].
Depuis son arrivée dans Arénaire, N. Brisebarre a travaillé sur deux questions posées dans [53]. La première demande de trouver un algorithme rapide qui permette d'obtenir les nombres flottants réalisant l'erreur maximale pouvant se produire lors de l'utilisation de la « méthode naïve » proposée. Il y a apporté une réponse partielle permettant de trouver en quelques heures les nombres flottants réalisant l'erreur maximale jusqu'au cas de la double précision IEEE-754 (53 bits) et d'exhiber une suite de nombres flottants donnant une grande erreur (possiblement maximale) en toute précision. La seconde question est une conjecture partiellement démontrée dans [53] et il travaille à compléter la preuve.

6.13. Validation d'une approximation fidèle avec la règle de Horner

Participants : Sylvie Boldo, Marc Daumas.

Mots clés : *virgule flottante*, *erreur d'arrondi*, *méthode formelle*, *Coq*.

Les polynômes sont utilisés dans de nombreuses applications et enfouis dans des bibliothèques telles que `libm`. S. Boldo et M. Daumas ont proposé un critère de fidélité pour une étape du schéma de Horner. Vient ensuite un programme Maple qui vérifie ce critère sur un polynôme entier associé à un domaine pour l'indéterminée et une éventuelle erreur de troncature. Un exemple d'utilisation est donné avec l'approximation des fonctions élémentaires.

L'évaluation du monôme $AX + Y$ apparaît comme une opération clé dans l'évaluation d'un polynôme par la méthode de Horner. De plus, dans certaines applications telles que l'évaluation des fonctions élémentaires, on utilise ce monôme avec la condition que AX est relativement petit devant Y.

S. Boldo et M. Daumas ont montré que sous certaines conditions, on peut obtenir une évaluation fidèle de $AX + Y$ sans utiliser d'opérateur *fused Mac* et même en présence d'une erreur importante sur les approximations de A et de X [20]. Une implantation est fidèle si elle produit toujours un arrondi correct IEEE (vers le haut ou vers le bas) sans pour autant laisser le choix du mode à l'utilisateur *a priori* ni lui indiquer le mode utilisé *a posteriori*. Cette étude se poursuit pour obtenir une condition simple sur la nature du polynôme évalué par la méthode de Horner.

6.14. Arithmétique d'intervalles en précision arbitraire

Le travail sur la bibliothèque MPFI *est commun avec le projet* SPACES*, Inria Lorraine.*
Participants : Nathalie Revol, Fabrice Rouillier [SPACES].

Mots clés : *arithmétique par intervalles*, *précision arbitraire*, *calcul certifié*, *adaptation automatique de la précision*.

La bibliothèque MPFI a été complétée, ses performances ont été étudiées et elle a été utilisée pour implanter l'algorithme de Newton par intervalles, avec adaptation automatique de la précision. Le travail sur la bibliothèque MPFI (voir §5.7) d'arithmétique par intervalles en précision arbitraire s'est poursuivi [33][34] : les fonctions trigonométriques et toutes les fonctions disponibles dans MPFR[57] ont été implantées de façon à rendre le résultat le plus précis possible. Un test comparatif a été mené sur l'élimination de Gauss appliquée à des H-matrices de dimensions 300×300 [54] : comparé avec `fi_lib`[67], le surcoût lié à l'utilisation d'une arithmétique flottante multiprécision est de 5 pour des intervalles en double précision et de seulement 12 pour 10 fois plus de précision.

L'algorithme de Newton pour une seule variable a été adapté pour tirer parti de cette arithmétique [16] et une étude préliminaire de l'algorithme de Hansen-Sengupta pour la résolution de systèmes linéaires a été conduite [32], en particulier pour ce qui concerne l'adaptation dynamique de la précision.

6.15. Modèles de Taylor et arithmétique flottante

Ce travail s'effectue en collaboration avec M. Berz (U. Michigan, USA*) et K. Makino (U. Illinois,* USA*).*
Participante : Nathalie Revol.

Mots clés : *développement de Taylor*, *calcul certifié*, *erreurs d'arrondi*.

L'utilisation des propriétés de l'arithmétique flottante IEEE-754 a permis de démontrer la correction des opérations arithmétiques dans le logiciel Cosy qui implante les modèles de Taylor. Calculer avec un modèle de Taylor signifie déterminer un développement de Taylor d'ordre quelconque, souvent élevé, de la fonction calculée ainsi qu'un intervalle qui est un encadrement à la fois du reste de Lagrange et des erreurs d'arrondi commises au cours du calcul. Tous les calculs s'effectuent en arithmétique flottante à précision fixée, cependant les résultats sont garantis. Ces modèles ont été développés par M. Berz, J. Hoefkens et K. Makino[68][56]. Ils ont réalisé le logiciel Cosy, qui correspond à la traduction informatique des modèles de Taylor, et l'ont utilisé pour des problèmes de physique des particules et d'astrophysique.

L'utilisation des propriétés de l'arithmétique flottante IEEE-754 a permis de démontrer que les algorithmes implantés dans Cosy fournissent bien un encadrement des erreurs d'arrondi.

6.16. Algèbre linéaire formelle

Participants : Claude-Pierre Jeannerod, Gilles Villard.

Mots clés : *algèbre linéaire*, *calcul symbolique*, *complexité*, *déterminant*, *inversion*.

C.-P. Jeannerod et G. Villard ont étudié la complexité algorithmique des opérations de base de l'algèbre linéaire. Les arithmétiques considérées ici sont des arithmétiques exactes (corps finis, entiers, rationnels, polynômes).

Les recherches en algorithmique matricielle exacte se sont poursuivies. À long terme les objectifs sont de mieux appréhender les complexités de problèmes de base tels que le calcul du déterminant, de la matrice inverse ou de formes normales de matrices sur des domaines d'entrées symboliques comme $\mathbb{F}_q$, $\mathbb{Z}$, $\mathbb{Q}$ ou $\mathsf{K}[x]$. Avec E. Kaltofen (Université de Caroline du Nord, USA) d'une part et M. Giesbrecht et A. Storjohann (Université de Waterloo, Ontario, Canada) d'autre part, nous proposons des états de l'art du domaine [12][14].

Avec J.-G. Dumas (Laboratoire LMC, Grenoble) nous avons travaillé sur une étude comparative, théorique et expérimentale, de plusieurs algorithmes calculant le rang d'une matrice sur un corps fini. Orienté vers les grandes matrices creuses, ce travail consiste notamment en la comparaison de méthodes d'élimination creuse avec renumérotation et de méthodes de type Krylov / Lanczos [28].

Nous approfondissons les études de complexité pour des matrices polynomiales comme préliminaires à des développements ultérieurs autour de la complexité pour des matrices à coefficients entiers. Nous traitons de formes normales de matrices sur $\mathbb{Z}[x]$ avec B. Beckermann (Laboratoire Ano, Lille) et G. Labahn (Université de Waterloo, Ontario, Canada) [36]. Nous améliorons la meilleure complexité connue d'un facteur n pour l'inversion d'une matrice polynomiale générique $n \times n$ de degré d. L'algorithme réduit le problème au produit de matrices polynomiales pour calculer l'inverse en temps quasi-optimal $n^3 d \log^{O(1)} nd$ (ce qui est de l'ordre de la taille de la sortie) [35][50]. Nous proposons aussi un nouvel algorithme de colonne réduction [55] et améliorons nos précédents résultats sur $\mathbb{Z}$ avec E. Kaltofen. Ces derniers résultats amélioraient la complexité du calcul du déterminant d'une matrice entière (modèle binaire) et du calcul du déterminant sans division (modèle algébrique). Nous montrons que l'approche s'étend au calcul du polynôme caractéristique [55].

Nous travaillons aussi avec G. Labahn sur des aspects mixtes formels / numériques. Nous améliorons l'efficacité pratique d'un test de primalité de polynômes approchés en exploitant au mieux la structure Sylvester des matrices utilisées, notamment par des techniques de mise à jour de factorisations QR successives. Cette approche est à la base du module Maple Snap pour l'arithmétique des polynômes approchés [30].

6.17. Logiciel LinBox

Participants : Pascal Giorgi, Gilles Villard.

Mots clés : *arithmétique exacte*, *algèbre linéaire*.

Dans le cadre du développement de la bibliothèque LinBox, P. Giorgi et G. Villard ont développé les aspects de généricité ainsi que des versions « par blocs » d'algorithmes d'algèbre linéaire pour les matrices creuses de grande taille.

P. Giorgi et G. Villard ont participé à la mise au point de la première distribution de la bibliothèque LinBox (cf. §5.9). La version comporte les éléments de base (arithmétique des corps finis, des entiers et structures de données vectorielles et matricielles) pour l'algorithmique de plus haut niveau. La bibliothèque est générique, programmée à l'aide de classes *template* C++. Ils ont plus particulièrement travaillé sur les aspects de généricité relativement aux arithmétiques utilisées. Les composants externes sont connectés via des adaptateurs (*wrappers*) et les entités LinBox sont des domaines qui fournissent les opérations de base sur leurs éléments. Les interfaces standard auxquelles ces adaptateurs et domaines doivent souscrire sont des objets C++ complètement définis appelés *archétypes*. P. Giorgi et G. Villard ont développé et testé divers adaptateurs au niveau des couches arithmétique des corps finis / extensions algébriques et de l'arithmétique matricielle [27]. À un plus haut niveau le cœur algorithmique notamment pour les grandes matrices creuses est l'approche Krylov / Lanczos [28]. Plusieurs implantations par blocs en fonction des entrées (la matrice et les projections) sont maintenant fournies, elles seront la base de développements en arithmétique entière notamment pour la résolution de systèmes diophantiens.

7. Contrats industriels

7.1. ST Microelectronics

Participants : Jean-Michel Muller (50%), Arnaud Tisserand (50%).

Mots clés : *algorithmes de division par des constantes*, *DSP*.

Suite au contrat avec la division compilation de ST Microelectronics commencé en 2000, nous avons développé une nouvelle version de notre algorithme de division par des constantes. Un article commun sur ce sujet sera prochainement soumis pour publication.

7.2. POSIC S.A.

Participant : Arnaud Tisserand.

Mots clés : *capteur de position*, *primitive arithmétique optimisée*, *FPGA*.

Dans le cadre d'un contrat entre l'Inria et la société POSIC S.A. (Neuchâtel, Suisse), nous avons étudié et implanté sur circuits programmables FPGA un algorithme d'interpolation pour un capteur de position. Cet algorithme est basé sur des primitives arithmétiques spécialement conçues et optimisées pour ce problème. Le travail réalisé dans ce contrat sera utilisé par la société POSIC S.A. pour une pré-série de ces produits.

8. Actions régionales, nationales et internationales

8.1. Actions nationales

8.1.1. ACI Cryptologie

Participants : Jean-Luc Beuchat, Arnaud Tisserand, Gilles Villard.

Mots clés : *architecture matérielle pour la cryptographie*, *chiffrement*, *FPGA*.

Nous avons obtenu en 2002 le financement d'un projet baptisé OpAC (Opérateurs Arithmétiques pour la Cryptographie) dans le cadre de l'« ACI Cryptologie » du MENRT. C'est un projet joint avec des membres des laboratoires Lirmm (coord. J.-C. Bajard) et GTA (coord. P. Elbaz-Vincent) de Montpellier, travaillant dans divers domaines (géométrie arithmétique, théorie des nombres, calcul symbolique, arithmétique des ordinateurs, algorithmique, microélectronique).

La motivation de ce projet repose sur le constat suivant : les circuits et opérateurs dédiés à la cryptographie utilisent des arithmétiques (système de représentation des nombres et algorithmes associés) classiques qui ne sont pas forcément les mieux adaptés, ni au type de calcul, ni aux objets mathématiques manipulés. Le projet étudie les calculs sur les objets mathématiques utilisés par les protocoles cryptographiques à clé publique. Les objectifs concernent la mise au point de représentations et algorithmes de base performants pour la définition d'architectures matérielles adaptées. À Lyon nous nous focalisons sur la réalisation matérielle, pour circuits FPGA, d'opérateurs arithmétiques dédiés aux corps finis et aux courbes elliptiques dans le contexte d'algorithmes de chiffrement.

8.1.2. ACI Jeunes Chercheurs « Arithmétique sur FPGA »

Participants : Jean-Luc Beuchat, Nicolas Boullis, Jérémie Detrey, Florent de Dinechin, Arnaud Tisserand.

Mots clés : *CAO*, *FPGA*, *opérateur arithmétique*.

L'action concertée incitative *Arithmétique sur* FPGA (acceptée en 2000) a permis de renouveler des licences d'outils de CAO, et financé des missions de J.-L. Beuchat, N. Boullis, J. Detrey, A. Tisserand.

8.2. Actions européennes

8.2.1. Action intégrée Alliance avec l'université de Cardiff

Participant : Arnaud Tisserand.

Mots clés : *système logarithmique de représentation des nombres*, *FPGA*.

A. Tisserand est le coordinateur français d'une action intégrée dans le cadre du programme franco-britannique Alliance. Cette collaboration avec l'équipe du professeur N. Burgess de l'université de Cardiff (Pays de Galles) porte sur la réalisation matérielle d'opérateurs arithmétiques, et en particulier sur l'implantation sur FPGA des opérations arithmétiques dans le système logarithmique de représentation des nombres. Cette collaboration a financé un séjour à Cardiff d'un membre du projet et un séjour d'un mois d'un doctorant gallois à Lyon.

8.2.2. Action Intégrée Alliance From Functions to FPGA

Participants : Florent de Dinechin, Tanguy Risset, John M. Spivey, Muniyappa Manjunathaiah.

Mots clés : *conception rapide pour FPGA*, *Alpha*, *HandelC*.

F. de Dinechin a participé à une action intégrée *Alliance* avec T. Risset, du projet COMPSYS, et J.M. Spivey et M. Manjunathaiah, de l'OUCL *(Oxford University Computing Laboratory)*, sur le thème : compilation rapide

de fonctions vers FPGA. Ce financement a permis des séjours de M. Manjunathaiah à Lyon et T. Risset à Oxford.

8.3. Actions internationales

8.3.1. Action LinBox

Participants : Pascal Giorgi, Claude-Pierre Jeannerod, Gilles Villard.

Mots clés : *bibliothèque générique C++*, *algèbre linéaire*, *calcul exact*, *corps fini*.

Nous poursuivons notre action de recherche collaborative LinBox autour du développement d'algorithmes efficaces en algèbre linéaire et de leur traduction en une bibliothèque de programmes (cf. §5.9). Le projet qui implique des chercheurs américains, canadiens a au départ été financé par une action incitative CNRS / NSF (1998-2000). Il a été soutenu en 2002 par la NSF côté américain et par une action JemStic du CNRS. Cette deuxième phase du projet a conduit à une première version publique `linbox-0.1.3` de la bibliothèque. Les développements s'orientent vers une gestion mémoire générique et sur l'algorithmique de plus haut niveau (cf. §6.17).

8.3.2. Action franco-marocaine

G. Villard collabore avec J. Della Dora et F. Jung du Laboratoire LMC-Imag de Grenoble, avec S. El Hajji de l'Université Mohamed V et avec A. Hilali de l'Institut National des Postes et Télécommunications de Rabat dans le cadre d'une action intégrée MENRT. En particulier, G. Villard co-encadre le travail de A. El Ghazi, doctorant à Rabat.

9. Diffusion des résultats

9.1. Organisation de conférences, édition de numéros spéciaux de journaux

M. Daumas, D. Michelucci (U. Bourgogne), J-M. Moreau (U. Lyon 1) et P. Langlois (U. Perpignan) organisent à Dijon une école thématique sur la question de l'*Adéquation Algorithmes-Arithmétique* pour mars 2003.

J.-M. Muller a été membre du comité de programme de Asap'02 (IEEE *International Conference on Application-specific Systems, Architectures and Processors*) et du comité scientifique de Scan'2002 (*Imacs-Gamm Conference on Scientific Computing, Computer Arithmetic and Validated Numerics*). Il a co-édité (avec J.-M. Chesneaux et C. Frougny) un numéro spécial de la revue *Theoretical Computer Science* (vol. 279 N°1-2, mai 2002).

N. Revol a co-édité avec S. El Hajji et P. van Dooren un numéro spécial du *Journal of Computational and Applied Mathematics* sur l'algèbre linéaire et l'arithmétique, qui paraîtra en 2003.

G. Villard a été membre du comité de programme de Issac'02 (*International Symposium on Symbolic and Algebraic Computation*), Lille, juillet 2002, ainsi que du comité de programme de Casc'02 (*Computer Algebra in Scientific Computing*), Big Yalta, septembre 2002. Il a édité un numéro spécial du *Journal of Symbolic Computation* (33(5) :519-520, 2002) « International Symposium on Algebraic and Symbolic Computation » suite à la conférence du même nom en 2001.

9.2. Enseignement de 3ème cycle

J.-M. Muller donne un cours de 24h (*Arithmétique des ordinateurs*) au DEA d'Informatique Fondamentale à l'ÉNS Lyon en 2002-2003. Depuis 2001-2002, F. de Dinechin et A. Tisserand ont proposé un cours intitulé *conception d'architectures matérielles* au DEA de l'ÉNS Lyon. Depuis 2001-2002 également, N. Revol et G. Villard donnent un cours intitulé *algorithmique et arithmétiques* commun aux DEA d'Informatique Fondamentale et d'Analyse Numérique à l'ÉNS Lyon.

9.3. Autres enseignements et responsabilités

F. de Dinechin assure son service de Maître de Conférences au sein du Magistère d'Informatique et de Modélisation de l'ÉNS Lyon. Il est responsable du programme « Europe » du MIM.

S. Boldo et N. Boullis assurent un monitorat à l'ÉNS Lyon et à l'Insa Lyon. P. Giorgi assure des vacations à l'université Claude Bernard - Lyon 1 (année 2002-2003). D. Defour a effectué des vacations à l'ÉNS Lyon et à l'IUT de Bourg-en-Bresse.

M. Daumas, F. de Dinechin et G. Villard ont participé comme examinateurs aux concours d'entrée aux Écoles Normales Supérieures.

M. Daumas rend disponible sur Internet le cours d'Y. Bertot à l'ÉNS Lyon pour faire connaître Coq et appréhender les problèmes liés à l'enseignement par les nouvelles technologies de l'information.

N. Revol est responsable d'un module de l'École Doctorale *Mathématiques et Informatique Fondamentale*.

S. Boldo a été représentante des élèves au Conseil Scientifique de l'ÉNS Lyon en 2001-2002. Elle a été élue représentante des chercheurs au conseil de laboratoire en 2002. N. Boullis est représentant des doctorants en informatique auprès de l'École Doctorale.

F. de Dinechin et A. Tisserand ont participé à la Semaine de la Science.

J.-B. Bianquis a effectué son stage de 2e année du Magistère MMFAI de la Région Parisienne sous la direction de G. Villard.

9.4. Animation de la communauté

N. Brisebarre est membre de la Commission de Spécialistes de 25ème section de l'Université J. Monnet de Saint-Étienne.

M. Daumas fait partie du comité de direction du GDR ARP du CNRS. Il a participé à l'organisation des journées AriNews (2 jours chaque fois) à Paris en janvier et à Perpignan en novembre, dans le cadre de l'action transversale « Arithmétique des ordinateurs » commune aux GDR ARP et ALP du CNRS.

Il est membre assesseur de la Commission de Spécialistes de 27e section de l'ÉNS Lyon.

M. Daumas et N. Revol ont participé à l'organisation de journées de travail (2 jours) de l'Action de Recherche Coopérative de l'Inria « AOC » (Arithmétique des Ordinateurs Certifiée) à Lyon en juin 2002.

F. de Dinechin fait partie du comité de rédaction de *Technique et science informatiques*. Il est membre suppléant de la Commission de Spécialistes de 27e section de l'ÉNS Lyon.

J.-M. Muller est directeur du Lip (UMR CNRS-ÉNS Lyon-Inria 5668, 85 personnes). Il est responsable de l'action spécifique *Arithmétique des Ordinateurs* du CNRS. Il a été membre élu de la commission d'évaluation de l'Inria jusqu'en juin 2002. Il est membre des commissions de spécialistes de 27e section de l'ÉNS Lyon et de l'Université de Provence.

N. Revol est membre des Commissions de Spécialistes de 26e section de l'Université Joseph Fourier de Grenoble et de l'Université des Sciences et Technologies de Lille (suppléante).

A. Tisserand a été responsable des moyens informatiques du laboratoire jusqu'en février 2002.

G. Villard est co-responsable de l'action spécifique « Calcul formel »[6] du département Stic du CNRS (RTP 23 : « Mathématique de l'informatique ») avec J.A. Weil (Inria Sophia Antipolis). Démarrée en janvier 2002, l'action implique l'ensemble de la communauté française. Elle s'intéresse au renforcement des activités et à l'ampleur prise par l'interaction entre le calcul formel et d'autres domaines. G. Villard est également membre suppléant de la commission de spécialistes de $26/34^{es}$ sections de l'Université des Sciences et Technologies de Lille.

[6] http://www.ens-lyon.fr/~gvillard/AS_STIC

9.5. Participations à des jurys

F. de Dinechin a participé au jury de thèse d'E. Fabiani à l'Irisa.

J.-M. Muller a participé aux jurys d'admissibilité des concours CR1, CR2 et DR2 de l'Inria. Il a été membre des jurys d'HDR de P. Zimmermann (26/11/01) et J.-M. Moreau (17/12/01), et du jury de thèse de M. Mayero (21/12/01).

9.6. Participation à des colloques, séminaires, invitations

9.6.1. Séminaires et exposés

- J.-L. Beuchat, S. Boldo, N. Boullis, D. Defour, P. Giorgi, N. Revol ont donné un exposé aux journées Arinews, Paris, 23 et 24 janvier 2002.
- J.-L. Beuchat, S. Boldo, N. Boullis, F. de Dinechin, J.-M. Muller, ont donné un exposé aux journées Arinews, Banyuls, 5 et 6 novembre 2002.
- J.-L. Beuchat a présenté un exposé au Laboratoire de Systèmes Logiques, École Polytechnique Fédérale de Lausanne, 1er mai 2002. Il a présenté un exposé à la Haute École Valaisanne, 7 mai 2002. J.-L. Beuchat a présenté un exposé au Laboratoire d'Informatique, de Robotique et de Microélectronique, Université Montpellier 2, 9 octobre 2002.
- S. Boldo a donné un exposé aux journées AOC, Lyon, 11 et 12 juin 2002.
- N. Brisebarre est invité à présenter ses travaux au Séminaire de Théorie Algébrique des Nombres du LACO, Université de Limoges, 27 janvier 2003. Il donnera un exposé invité au Séminaire de Théorie des Nombres, Algorithmique et Cryptographie du Grimm, Université Toulouse 2, en février 2003.
- M. Daumas a présenté un exposé au Séminaire d'Analyse Numérique du Laboratoire de Mathématiques d'Orsay (février 2002). Il a présenté les travaux sur « Coq and formal specification for 754R » au groupe de travail sur la révision de la norme Ansi-IEEE-754 à Cuppertino, Californie (julllet 2002). M. Daumas a présenté un ICASE colloquium au Langley Research Center de la Nasa en Virginie (août 2002). Il a donné un séminaire au School of Computational Science and Information Technology de Florida State University à Tallahassee (juillet 2002). M. Daumas a présenté un cours invité au 8ème Symposium sur les Architectures Nouvelles de Machines conjoint avec les 14es RENcontres francophones du PARallélisme à Hamamet, Tunisie (avril 2002). Il a donné une conférence plènière au 10th GAMM-IMACS International Symposium on Scientific Computing, Computer Arithmetic, and Validated Numerics (Scan 2002) à Paris (septembre 2002).
- D. Defour a présenté un exposé au Groupe de Recherches En Informatique, Image, Instrumentation de Caen, 23 avril 2002.
- P. Giorgi a présenté un exposé aux journées calcul formel libre (Workshop on Open Source Computer Algebra), Lyon, 21 au 23 mai 2002.
- C.-P. Jeannerod a donné un exposé invité à la conférence ACA (Applications of Computer Algebra, Volos, Grèce, 25-28 juin 2002). Il donnera un exposé invité de Iciam 2003 (International Congress on Industrial and Applied Mathematics, Sydney, Australie, 7-11 juillet 2003).
- N. Revol a été conférencière invitée au Forum des Jeunes Mathématiciennes et des Jeunes Informaticiennes, à Paris, en mars 2002, et au colloque de l'Irem à Lyon en juin 2002.
- A. Tisserand a présenté un séminaire du Magistère d'Informatique et Modélisation de l'ÉNS Lyon en février 2002.
- G. Villard est orateur invité à la conférence « Siam Conference on Applied Linear Algebra » qui se tiendra à Williamsburg, Virginie en juillet 2003. Il a exposé au séminaire du projet Algorithmes, Inria Rocquencourt (27/05/02).

9.6.2. Invitations

- C.-P. Jeannerod a été invité à passer le mois d'avril 2002 au Symbolic Computation Group de l'Université de Waterloo (http://www.scg.uwaterloo.ca/SCG). Cela a notamment permis de poursuivre la collaboration commencée en 2001 avec G. Labahn sur des problèmes numériques de l'arithmétique des polynômes.

10. Bibliographie

Bibliographie de référence

[1] éditeurs M. DAUMAS, J.-M. MULLER., *Qualité des calculs sur ordinateur : vers des arithmétiques plus fiables.* Masson, 1997, http://www.ens-lyon.fr/~jmmuller/livre_masson.html.

[2] N. J. HIGHAM. *Accuracy and stability of numerical algorithms.* SIAM, 1996, http://www.maths.man.ac.uk/~higham/asna.

[3] P. MARKSTEIN. *IA-64 and elementary functions : speed and precision.* Prentice Hall, 2000, http://www.markstein.org/.

[4] J.-M. MULLER. *Elementary functions, algorithms and implementation.* Birkhauser, 1997, http://www.ens-lyon.fr/~jmmuller/book_functions.html.

[5] J. VON ZUR GATHEN, J. GERHARD. *Modern Computer Algebra.* Cambridge University Press, 1999, http://www-math.uni-paderborn.de/mca/.

Livres et monographies

[6] *Special issue « Real Numbers and Computers » of Theoretical Computer Science.* éditeurs J.-M. CHESNEAUX, C. FROUGNY, J.-M. MULLER., volume 279, mai, 2002.

[7] *Special issue on « Linear Algebra and Arithmetic » of the Journal of Computational and Applied Mathematics.* éditeurs S. EL HAJJI, N. REVOL, P. VAN DOOREN., 2003, à paraître.

Articles et chapitres de livre

[8] J.-L. BEUCHAT, J.-O. HAENNI, H. FABIO RESTREPO, C. TEUSCHER, F. J. GÓMEZ, E. SANCHEZ. *Approches matérielles et logicielles de l'algorithme de chiffrement* IDEA. in « Technique et science informatiques », numéro 2, volume 21, 2002, pages 203-224.

[9] J.-L. BEUCHAT, A. TISSERAND. *Évaluation polynomiale en ligne de fonctions élémentaires sur* FPGA. in « Technique et science informatique », à paraître.

[10] L. CHEN, W. EBERLY, E. KALTOFEN, B. SAUNDERS, W. TURNER, G. VILLARD. *Efficient matrix preconditioners for black box linear algebra.* in « Linear Algebra and its Applications », volume 343-344, 2002, pages 119-146.

[11] T. GAUTIER, H. HONG, J. ROCH, W. SCHREINER, G. VILLARD. *Parallel computer algebra systems.* éditeurs J. GRABMEIER, E. KALTOFEN, V. WEISPFENNING., in « Computer Algebra Handbook », Springer-Verlag, Heidelberg, Germany, 2002, pages 146-150.

[12] M. GIESBRECHT, A. STORJOHANN, G. VILLARD. *Algorithms for matrix canonical forms.* éditeurs J. GRABMEIER, E. KALTOFEN, V. WEISPFENNING., in « Computer Algebra Handbook », Springer-Verlag, Heidelberg, Germany, 2002, pages 39-41.

[13] C.-P. JEANNEROD. *On Matrix Perturbations with Minimal Leading Jordan structure.* in « Journal of Computational and Applied Mathematics », 2003, à paraître.

[14] E. KALTOFEN, G. VILLARD. *Computing the sign or the value of the determinant of an integer matrix, a complexity survey.* in « Journal of Computational and Applied Mathematics », 2003, à paraître.

[15] V. LEFÈVRE, J.-M. MULLER. *On-the-Fly Range Reduction.* in « Journal of VLSI Signal Processing », à paraître.

[16] N. REVOL. *Newton iteration using multiple precision interval arithmetic : the univariate case.* in « Numerical Algorithms », à paraître.

[17] B. SAUNDERS, A. STORJOHANN, G. VILLARD. *Matrix rank certification.* in « Elect. J. Linear Algebra », à paraître.

Communications à des congrès, colloques, etc.

[18] J.-L. BEUCHAT, A. TISSERAND. *Opérateur en-ligne sur* FPGA *pour l'implantation de quelques fonctions élémentaires.* in « 8ème SYMPosium en Architectures nouvelles de machines », pages 267-274, Hamamet, Tunisie, avril, 2002.

[19] J.-L. BEUCHAT, A. TISSERAND. *Small Multiplier-based Multiplication and Division Operators for Virtex-II Devices.* in « 12th International Conference on Field-Programmable Logic and Applications (FPL) », série Lecture Notes in Computer Science (LNCS), volume 2438, Springer, éditeurs M. GLESNER, P. ZIPF, M. RENOVELL., pages 513-522, Montpellier, France, septembre, 2002.

[20] S. BOLDO, M. DAUMAS. *Faithful rounding without fused multiply and accumulate.* in « SCAN'2002 », Paris, France, 2002, http://scan2002.lip6.fr/abstracts/boldo.pdf.

[21] S. BOLDO, M. DAUMAS. *Properties of the subtraction valid for any floating point system.* in « 7th International Workshop on Formal Methods for Industrial Critical Systems », pages 137-149, Málaga, Spain, 2002, http://www.inrialpes.fr/vasy/fmics/workshop-7/proceedings.pdf.

[22] N. BOULLIS, A. TISSERAND. *Génération automatique d'architectures de calcul pour des opérations linéaires : application à l'IDCT sur FPGA.* in « 8ème SYMPosium en Architectures nouvelles de machines », pages 283-290, Hamamet, Tunisie, avril, 2002.

[23] D. DEFOUR, F. DE DINECHIN, J.-M. MULLER. *A new scheme for table-based evaluation of functions.* in « 36th Conference on signals, systems and computers », IEEE Computer Society Press, Asilomar, Pacific

Grove, California, USA, novembre, 2002.

[24] D. DEFOUR, F. DE DINECHIN. *Software carry-save for fast multiple-precision algorithms.* in « 35th International Congress of Mathematical Software (ICMS 2002) », World Scientific, éditeurs A. M. COHEN, X.-S. GAO, N. TAKAYAMA., pages 29-39, Beijing, China, août, 2002, Updated version of LIP research report 2002-08.

[25] F. DE DINECHIN, J. DETREY. *Multipartite Tables in JBits for the Evaluation of Functions on FPGAs.* in « IEEE Reconfigurable Architecture Workshop, International Parallel and Distributed Symposium », Fort Lauderdale, Florida, avril, 2002, Updated version of LIP research report 2001-44.

[26] F. DE DINECHIN, T. RISSET, M. MANJUNATHAIAH, M. SPIVEY. *Design of highly parallel architectures with Alpha and Handel.* in « Forum on Design Languages », Marseille, France, septembre, 2002.

[27] J.-G. DUMAS, T. GAUTIER, M. GIESBRECHT, P. GIORGI, B. HOVINEN, E. KALTOFEN, B. SAUNDERS, W. TURNER, G. VILLARD. *LinBox : A Generic Library for Exact Linear Algebra.* in « 35th International Congress of Mathematical Software (ICMS 2002) », World Scientific, éditeurs A. M. COHEN, X.-S. GAO, N. TAKAYAMA., pages 40-50, Beijing, China, août, 2002.

[28] J.-G. DUMAS, G. VILLARD. *Computing the rank of large sparse matrices over finite fields.* in « CASC'2002, The Fifth International Workshop on Computer Algebra in Scientific Computing », Springer-Verlag, Big Yalta, Crimea, Ukraine, 2002.

[29] C.-P. JEANNEROD. *A reduced form for perturbed matrix polynomials.* in « International Symposium on Symbolic and Algebraic Computation (ISSAC'02) », ACM Press, éditeurs T. MORA., pages 131-137, Lille, France, juillet, 2002.

[30] C.-P. JEANNEROD, G. LABAHN. *The Snap package for arithmetic with numeric polynomials.* in « 35th International Congress of Mathematical Software (ICMS 2002) », World Scientific, éditeurs A. M. COHEN, X.-S. GAO, N. TAKAYAMA., pages 61-71, Beijing, China, août, 2002.

[31] J.-M. MULLER. *Proposals for a Specification of the Elementary Functions.* in « SCAN'2002 », Paris, France, 2002, http://scan2002.lip6.fr/abstracts/muller.pdf.

[32] N. REVOL. *Reliable and accurate solutions of linear and nonlinear systems.* in « SIAM Conference on Optimization », Toronto, Canada, 2002.

[33] N. REVOL, F. ROUILLIER. *Motivations for an arbitrary precision interval arithmetic and the MPFI library.* in « Validated Computing », Toronto, Canada, 2002.

[34] N. REVOL, F. ROUILLIER. *MPFI : a library for arbitrary precision interval arithmetic.* in « SCAN'2002 », Paris, France, 2002, http://scan2002.lip6.fr/abstracts/revolrouillier.pdf.

[35] G. VILLARD. *Exact computation of the determinant and of the inverse of a matrix.* in « Workshop on Complexity, Foundations of Computational Mathematics FoCM'02 », Minneapolis, Minnesota, USA, août, 2002.

Rapports de recherche et publications internes

[36] B. BECKERMANN, G. LABAHN, G. VILLARD. *Normal forms for general polynomial matrices.* rapport technique, numéro LIP 2002-1, LIP, janvier, 2002.

[37] J.-L. BEUCHAT. *High Throughput Implementations of the* RC6 *Block Cipher Using Virtex-E and Virtex-II Devices.* rapport technique, numéro 4495, Inria, juillet, 2002, http://www.inria.fr/rrrt/rr-4495.html.

[38] J.-L. BEUCHAT. *Modular Multiplication for FPGA Implementation of the* IDEA *Block C ipher.* rapport technique, numéro 4558, Inria, septembre, 2002, http://www.inria.fr/rrrt/rr-4558.html.

[39] J.-L. BEUCHAT. *Some Modular Adders and Multipliers for Field Programmable Gate Arrays.* rapport technique, numéro LIP 2002-37, LIP, 2002.

[40] J.-L. BEUCHAT, A. TISSERAND. *Évaluation polynomiale en ligne de fonctions élémentaires sur* FPGA. rapport technique, numéro 4557, Inria, septembre, 2002, http://www.inria.fr/rrrt/rr-4557.html.

[41] J.-L. BEUCHAT, A. TISSERAND. *Small Multiplier-based Multiplication and Division Operators for Vir tex-II Devices.* rapport technique, numéro 4494, Inria, juillet, 2002, http://www.inria.fr/rrrt/rr-4494.html.

[42] S. BOLDO, M. DAUMAS. *Necessary and sufficient conditions for exact floating-point operations.* rapport technique, numéro 4644, Inria, 2002, http://www.inria.fr/rrrt/rr-4644.html.

[43] S. BOLDO, M. DAUMAS. *Properties of the subtraction valid for any floating point system..* rapport technique, numéro 4473, Inria, 2002, http://www.inria.fr/rrrt/rr-4473.html.

[44] N. BOULLIS, A. TISSERAND. *Génération automatique d'architectures de calcul pour des opérations linéaires : application à l'*IDCT *sur* FPGA. rapport technique, numéro 4486, Inria, novembre, 2002, http://www.inria.fr/rrrt/rr-4486.html.

[45] D. DEFOUR, F. DE DINECHIN. *Software Carry-Save for Fast Multiple-Precision Algorithms.* rapport technique, numéro LIP 2002-8, LIP, février, 2002.

[46] D. DEFOUR, F. DE DINECHIN, J.-M. MULLER. *A new scheme for table-based evaluation of functions.* rapport technique, numéro 4637, Inria, 2002, http://www.inria.fr/rrrt/rr-4637.html.

[47] D. DEFOUR. *Cache-Optimised Methods for the Evaluation of Elementary Functions.* rapport technique, numéro LIP 2002-38, LIP, 2002.

[48] J.-G. DUMAS, T. GAUTIER, M. GIESBRECHT, P. GIORGI, P. HOVINEN, E. KALTOFEN, B. SAUNDERS, W. TURNER, G. VILLARD. *LinBox : A Generic Library for Exact Linear Algebra.* rapport technique, numéro LIP 2002-15, LIP, 2002.

[49] C.-P. JEANNEROD, G. LABAHN. *SNAP user's guide.* rapport technique, numéro CS-2002-22, University of Waterloo, April, 2002.

[50] C.-P. JEANNEROD, G. VILLARD. *Inversion of generic matrix polynomials.* rapport technique, Inria, December, 2002.

[51] E. KALTOFEN, G. VILLARD. *Computing the sign or the value of the determinant.* rapport technique, numéro LIP 2002-1, Inria, LIP, janvier, 2002.

[52] J.-M. MULLER. *« Partially rounded » Small-Order Approximations for Accurate, Hardware-Oriented, Table-Based Methods.* rapport technique, numéro 4593, Inria, 2002, http://www.inria.fr/rrrt/rr-4593.html.

[53] J.-M. MULLER. *Accelerating Floating-Point Division when the Divisor is known in Advance.* rapport technique, numéro 4532, Inria, 2002, http://www.inria.fr/rrrt/rr-4532.html.

[54] N. REVOL, F. ROUILLIER. *Motivations for an arbitrary precision interval arithmetic and the MPFI library.* rapport technique, numéro 4498, Inria, 2002, http://www.inria.fr/rrrt/rr-4498.html.

[55] G. VILLARD. *Algorithmique en algèbre linéaire exacte.* Mémoire d'habilitation à diriger des recherches, version préliminaire, novembre, 2002.

Bibliographie générale

[56] M. BERZ, J. HOEFKENS. *Verified high-order inversion of functional dependencies and interval Newton methods.* in « Reliable Computing », numéro 5, volume 7, 2001, pages 379-398.

[57] D. DANEY, G. HANROT, V. LEFÈVRE, F. ROUILLIER, P. ZIMMERMANN. *The MPFR library.* http://www.mpfr.org, 2001.

[58] D. DAS SARMA, D. W. MATULA. *Faithful bipartite ROM reciprocal tables.* in « Proceedings of the 12th Symposium on Computer Arithmetic », éditeurs S. KNOWLES, W. H. MCALLISTER., pages 17-28, Bath, England, 1995, http://computer.org/proceedings/arith/7089/7089toc.htm.

[59] F. DE DINECHIN, A. TISSERAND. *Some improvements on multipartite table methods.* in « Proceedings of the 15th Symposium on Computer Arithmetic », éditeurs N. BURGESS, L. CIMINIERA., pages 128-135, Vail, Colorado, 2001, http://computer.org/proceedings/arith/1150/1150toc.htm.

[60] J. DUMAS, B. SAUNDERS, G. VILLARD. *On efficient sparse integer matrix Smith normal form computations.* in « Journal of Symbolic Computation, *special issue on* Computer Algebra and Mechanized Reasoning », numéro 1-2, volume 32, 2001, pages 71-99.

[61] M. D. ERCEGOVAC, T. LANG. *On-Line Arithmetic : A Design Methodology and Applications in Digital Signal Processing.* in « IEEE Acoustics, Speech, and Signal Processing Society Workshop on VLSI Signal Processing », pages 252-263, novembre, 1988.

[62] J.-C. FAUGÈRE. *A new efficient algorithm for computing Gröbner bases F_4.* in « Journal of Pure and Applied Algebra », numéro 1-3, volume 139, 1999, pages 61-88.

[63] *Field-Programmable Logic and Applications - Reconfigurable Computing Is Going Mainstream.* éditeurs M. GLESNER, P. ZIPF, M. RENOVELL., série Lecture Notes in Computer Science, numéro 2438, Springer, 2002.

[64] G. HUET, G. KAHN, C. PAULIN-MOHRING. *The Coq Proof Assistant : A Tutorial : Version 6.1.* Technical Report, numéro 204, Inria, 1997, http://www.inria.fr/rrrt/rt-0204.html.

[65] X. LAI. *On the Design and Security of Block Ciphers.* série ETH Series in Information Processing, Hartung-Gorre Verlag Konstanz, 1992.

[66] V. LEFÈVRE. *Multiplication par une constante.* in « Réseaux et Systèmes Répartis, Calculateurs Parallèles - numéro spécial sur l'Arithmétique des Ordinateurs », numéro 4-5, volume 13, 2001, pages 465-484.

[67] M. LERCH, G. TISCHLER, J. WOLFF VON GUDENBERG, W. HOFSCHUSTER, W. KRÄMER. *The Interval Library filib++ 2.0 - Design, Features and Sample Programs.* Research Report, numéro 2001/4, Universität Wuppertal, Germany, 2001.

[68] K. MAKINO, M. BERZ. *Higher order verified inclusions of multidimensional systems by Taylor models.* in « Nonlinear Analysis », volume 47, 2001, pages 3503-3514.

[69] B. MOURRAIN. *Computing the isolated roots by matrix methods.* in « Journal of Symbolic Computation », numéro 6, volume 26, 1998, pages 715-738, http://www.sciencedirect.com/.

[70] J.-M. MULLER. *A few results on table based methods.* in « Reliable Computing », numéro 3, volume 5, 1999, pages 279-288.

[71] A. M. ODLYZKO. *Discrete logarithms : the past and the future.* in « Designs, Codes, and Cryptography », numéro 2/3, volume 19, 2000, pages 129-145, http://www.wkap.nl/article.pdf?253937.

[72] R. L. RIVEST, M. J. B. ROBSHAW, R. SIDNEY, Y. L. YIN. *The RC6 Block Cipher.* 1998, http://www.rsasecurity.com/rsalabs/rc6.

[73] M. J. SCHULTE, J. E. STINE. *Approximating elementary functions with symmetric bipartite tables.* in « IEEE Transactions on Computers », numéro 8, volume 48, 1999, pages 842-847.

[74] W. F. WONG, E. GOTO. *Fast evaluation of the elementary functions in single precision.* in « IEEE Transactions on Computers », numéro 3, volume 44, 1995, pages 453-457.