Zeitschrift: bulletin.ch / Electrosuisse

Herausgeber: Electrosuisse

Band: 98 (2007)

Heft: 3

Artikel: Field Programmable Gate Arrays (FPGA) versus Digital Signal Processors (DSP)

Autor: Johnsen, O. / Crausaz, P. / Magliocco, C.

DOI: https://doi.org/10.5169/seals-857413

Field Programmable Gate Arrays (FPGA) versus Digital Signal Processors (DSP)

Les FPGA comme alternatives aux DSP pour le traitement du signal

Les FPGA ont une puissance de traitement très élevée, car ces dispositifs peuvent effectuer des multiplications et des additions à plus de 100 MHz et des centaines de multiplicateurs et additionneurs peuvent être implémentés en parallèle. Les multiplicateurs peuvent être optimisés individuellement en ajustant pour chaque multiplicateur la largeur de données des signaux et des coefficients. Avec les outils de développements actuels, la conception ne demande pas beaucoup plus d'effort que pour des processeurs.

O. Johnsen, P. Crausaz, C. Magliocco, D. Prêtre, F. Corthay, G. Maître, M. Kocher

Les départements d'électricité de plusieurs sites de la Haute école spécialisée de la Suisse occidentale (HES-SO) ont développé en commun un know-how pour l'implémentation de filtres sur FPGA. Dans cet article, on décrit les principales étapes qui sont:

- la spécification et le calcul du filtre,
- la conversion du filtre en arithmétique fixe avec analyse des performances et
- la synthèse du filtre qui inclut la spécification du filtre en VHDL (Very High Speed Integrated Circuit Hardware Description Language).

Comme exemple, on a sélectionné les filtres récursifs, car ceux-ci posent le plus de problèmes concernant les arrondis, la saturation et la stabilité. Des programmes Matlab et des descriptions VHDL ont été développés pour réaliser l'implémentation.

Comparaison DSP et FPGA

Ces deux technologies ont chacune leurs avantages et propriétés que l'on va énumérer ci-après.

Le DSP

Le DSP est très rapide, programmable et séquentiel. Des outils de développement confortables existent. Il est bon marché et a parfois plusieurs unités arithmétiques en parallèle. Il y a souvent des possibilités de connexion multiprocesseurs. Ces deux techniques permettent une augmentation importante de la vitesse.

La FPGA

La FPGA peut être considérée comme une électronique programmable ou configurable. Elle est moins rapide que le DSP. Par contre, elle est hautement parallèle. On peut en effet implémenter plusieurs centaines de multiplicateurs et additionneurs conduisant à un débit très élevé. On peut aussi optimiser individuellement chaque unité. Ainsi, alors que sur un DSP, toutes les multiplications sont 16×16 ou 32×32 bits, avec une FPGA, un multiplicateur peut être par exemple 7×13 bits et un autre 34×19 bits, conduisant à une utilisation optimale des ressources. Les FPGA sont proposées par plusieurs fabricants, en particulier Xilinx, Altera et Actel qui proposent des technologies et des solutions différentes. Lors de grandes séries, elles peuvent être remplacées par des Application Specific Integrated Circuits (ASIC). On espère par cet article apporter une aide aux choix de la technologie pour le traitement numérique du signal.

Introduction aux filtres numériques

Les filtres numériques permettent d'effectuer des opérations similaires aux filtres analogiques. Nous allons ici uniquement faire un rappel sommaire. Pour plus d'informations, on peut se référer à l'un des nombreux livres sur le domaine, par exemple [1] et [2]. On a deux types principaux de filtres, les filtres récursifs et les filtres non récursifs. On va considérer le filtre récursif, car le filtre non récursif est un cas particulier et qui pose moins de problèmes d'arrondi et de stabilité que les filtres récursifs. Les filtres numériques comme les filtres analogiques sont généralement décomposés en une cascade de cellules du deuxième ordre appelées cellules biquadratiques, comme montré sur les figures 1 et 2.

La fonction de transfert d'une cellule est

$$G(z) = \frac{b_0 + b_1 \cdot z^{-1} + b_2 \cdot z^{-2}}{1 + a_1 \cdot z^{-1} + a_2 \cdot z^{-2}}$$

et son équation aux différences: $y(n) = b_0 \cdot x(n) + b_1 \cdot x(n-1) + b_2 \cdot x(n-2) - a_1 \cdot y(n-1) - a_2 \cdot y(n-2)$.

La figure 2 montre le schéma bloc d'une telle cellule ainsi que les formats des signaux et coefficients – qui seront expliqués plus tard.

On observe que 3 opérations sont nécessaires pour le filtrage numérique:

- la multiplication (par une constante),
- l'addition (ou plutôt le cas particulier de l'accumulation) et
- l'élément de retard.

Ces 3 opérations sont aisées à effectuer sous forme programmée (DSP) ou câblée (FPGA).

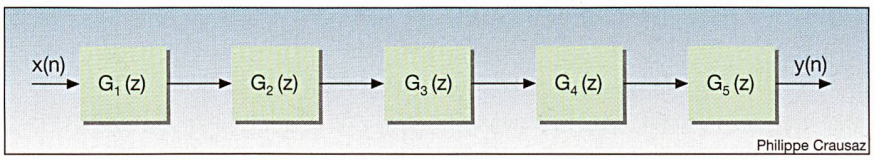

Figure 1 Illustration de la décomposition d'un filtre numérique en cellules du 2e ordre

Calcul de filtres numériques

De nombreux logiciels permettent de calculer les filtres numériques. Ce calcul peut être séparé en plusieurs parties. Il y a de nombreuses options, on considère uniquement les parties les plus importantes.

Spécifications

Les spécifications d'un filtre consistent généralement à spécifier le gabarit, c'est-à-dire les limites de la réponse fréquentielle.

Selection du type de filtre

Le type (Butterworth, Tchébycheff, elliptique, ...) indique le comportement de la réponse fréquentielle en bande passante et bloquante. Il influence pour un gabarit donné, l'ordre du filtre et donc sa complexité. L'ordre d'un filtre n'est pas le seul critère de complexité, en effet, le type de filtre a une influence sur le nombre d'opérations.

Structures directes et transposées

Il existe deux structures principales de filtres, les structures 1 et 2 ainsi que leurs versions transposées. Celle de la figure 2 est la structure 2 transposée qui est canonique.

Les logiciels fournissent les coefficients des cellules biquadratiques.

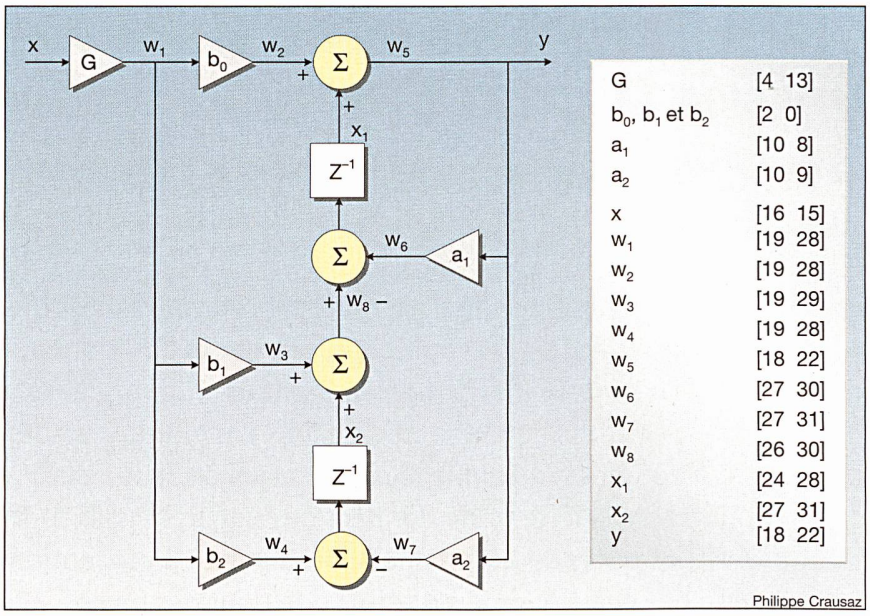

Figure 2 Schéma bloc d'une cellule du 2e ordre (directe 2 transposée) ainsi que les formats des signaux et coefficients

Le format [10 8] signifie que l'on a 10 bits, dont 8 après la virgule.

Arithmétique à précision limitée

Les coefficients et signaux internes sont à précision limitée. Cela signifie que l'on doit effectuer des quantifications. La quantification des coefficients et des signaux internes a des conséquences très différentes, c'est pourquoi on les considère séparément.

On travaille généralement en signé et en complément à 2. Par exemple avec 8 bits, on peut représenter 256 valeurs entre −128 et +127. Lorsque l'on travaille en précision limitée, on spécifie 2 quantités: le nombre de bits et la position de la virgule. Le format [19 17] signifie que l'on a 19 bits, dont 17 après la virgule. Dans ce cas, on peut représenter des nombres entre ±2 avec 17 bits de précision après la virgule. Comme autre exemple, le format [2 0] donne les valeurs {−2; −1; 0; 1}, le format [2 1] les valeurs {−1; −0,5; 0; 0,5} et le format [2 −1] les valeurs {−4, −2; 0; 2}.

Pour minimiser l'erreur de quantification, il faut suffisamment de bits de poids faible. En plus de la quantification, on a la saturation (overflow) qui apparaît lorsque l'on dépasse les valeurs autorisées. Pour éviter la saturation, il faut un nombre suffisant de bits de poids fort (avant la virgule). Le dilemme est que l'on désire minimiser le nombre total de bits pour réduire la complexité de la FPGA.

On peut observer sur la figure 2 qu'on a en principe 5 coefficients, pour une cellule (filtre) biquadratique donc 5 multiplications, même 6 avec G. D'habitude, une seule cellule a un coefficient G. Pour diminuer la complexité, on met généralement le coefficient b_0 à une puissance de 2. Comme souvent $b_2 = \pm b_0$ et $b_1 = \pm 2 \cdot b_0$, le nombre de vraies multiplications est quelquefois réduit à 2.

Arrondi des coefficients

L'arrondi des coefficients change légèrement la réponse fréquentielle des filtres. Elle ne provoque aucun bruit. Plus les pôles et zéros sont proches du cercle unité, plus on est sensible à la précision des coefficients. Avec la décomposition en cellules biquadratiques, on est moins sensible à l'arrondi que si l'on réalise le filtre entier en une cellule. La réponse fréquentielle après arrondi des coefficients est aisée à calculer. L'équation ci-dessous donne un exemple de coefficients pour un filtre numérique composé de 3 cellules biquadratiques (équation 1).

Les formats des coefficients sont indiqués par l'équation 2.

Les FPGA et les logiciels de description logique telle que le VHDL ne connaissant pas le point décimal, les coefficients sont transformés en entiers. Ainsi 0,890625 en [10 9] devient $0{,}890625 \cdot 2^9 = 456$. La position du point décimal n'est pas importante pour les multiplications, mais elle l'est pour les additions (le point doit être aligné).

Quantification des signaux

On va considérer ici les moyens d'éviter la saturation et de minimiser le bruit de quantification. La quantification des

Equation 1

$$SOS = \begin{pmatrix} \overbrace{1}^{b_0} & \overbrace{0 \quad -1}^{b_1} & \overbrace{1}^{b_2} & \overbrace{1}^{a_0=1} & \overbrace{-1.83984375}^{a_1} & \overbrace{0.890625}^{a_2} \\ 1 & 0.5 & 1 & 1 & -1.80859375 & 0.92578125 \\ 1 & -2 & 1 & 1 & -1.94140625 & 0.96484375 \end{pmatrix}$$

Equation 2

$$SOS_{Format} = \begin{pmatrix} [2\ 0] & [2\ 0] & [2\ 0] & [2\ 0] & [10\ 8] & [10\ 9] \\ [2\ 0] & [2\ 1] & [2\ 0] & [2\ 0] & [10\ 8] & [10\ 9] \\ [2\ 0] & [2\ 0] & [2\ 0] & [2\ 0] & [10\ 8] & [10\ 9] \end{pmatrix}$$

signaux ne modifie pas la réponse fréquentielle, mais provoque du bruit.

Pour éviter la saturation, il faut pouvoir déterminer en chaque point du système les valeurs extrémales possibles du signal et ensuite choisir le nombre de bits avant la virgule en conséquence. Il existe 3 critères, appelés l_1, l_2 et l_∞. Il faut se référer à la littérature pour choisir le critère. Si on applique ces règles (sélection correcte du nombre de bits avant la virgule), il n'y a aucun risque de saturation.

Il y a 2 méthodes pour supprimer les bits les moins significatifs, l'arrondi et la troncation. L'arrondi est préférable, car il provoque une erreur comprise entre ±½ LSB (Least Significant Bit) alors que la troncation provoque une erreur comprise entre 0 et 1 LSB pour les quantités positives et entre 0 et –1 LSB pour les quantités négatives. On utilise généralement la troncation, car l'arrondi nécessite beaucoup de calcul (surface) et la troncation avec plus de précision est préférable à l'arrondi.

A chaque endroit où on effectue un arrondi ou une troncation, on a une erreur. Cette erreur peut être assimilée à un bruit appelé bruit de quantification. Des programmes de simulation ont été développés avec Matlab pour déterminer l'effet total de toutes les sources de bruit du filtre sur la sortie. La simulation donne des résultats très proches des mesures expérimentales. Ces programmes choisissent un format des données proches de l'optimal. La figure 2 montre un exemple d'une cellule biquadratique (structure directe transposée) et des formats des signaux.

Pour minimiser ce bruit, on effectue souvent une seule troncation (ou arrondi) par cellule biquadratique. On observe avec les formats des signaux de la figure 2 que seul l'additionneur donnant w_5 effectue une troncation importante. On peut vérifier que les autres troncations n'ont absolument aucune influence sur la sortie. Les multiplicateurs en particulier ne font pas de troncation. La sortie w_5 a 22 bits après la virgule. Il ne semble pas logique que l'additionneur ait des entrées à 28 bits, une troncation avant sommation semble donc s'imposer. On peut montrer qu'une telle troncation augmente le bruit de quantification, mais que si l'on prend des précautions, elle peut être faible. Comme exemple, la simulation et les résultats expérimentaux ont montré une baisse négligeable des performances en tronquant certains des signaux de la figure 2 aux formats suivants:

w_2 [14 23] au lieu de [19 28]
w_3 [15 25] au lieu de [19 29]
w_4 [17 26] au lieu de [19 28]
w_6 [21 24] au lieu de [27 30]
w_7 [22 26] au lieu de [27 31]
w_8 [21 24] au lieu de [26 30]
x_1 [19 23] au lieu de [24 28]
x_2 [21 25] au lieu de [27 31]

Réalisation de la FPGA

La réalisation de la FPGA passe par le langage VHDL. VHDL est un langage de description de systèmes logiques. Il est devenu l'outil de base pour les développements en électronique digitale. Le flow utilisé pour la réalisation de la FPGA est montré sur la figure 3.

Avec les scripts Matlab développés plus haut, on obtient une description complète des filtres. Cette description inclut en particulier le type de structure, le nombre de cellules biquadratiques, les coefficients ainsi que les formats des coefficients et des signaux.

Un script Matlab convertit les valeurs des coefficients et des formats en génériques VHDL, mais ne génère pas la structure complète du filtre. Quelques structures de filtres ont été écrites en VHDL une fois pour toutes. Le script choisit une des structures existantes et la configure.

Il est possible de simuler la description VHDL du filtre à l'aide d'un simulateur VHDL (Modelsim). Cette simulation a lieu dans le domaine temporel. Un test bench, c'est-à-dire un module écrit en VHDL non synthétisable, permet de générer un signal chirp (sinus dont la fréquence augmente avec le temps) automatiquement et de l'appliquer à l'entrée du filtre. On peut ainsi vérifier son comportement en fréquence.

Ce même fichier VHDL sera utilisé pour la synthèse, c'est-à-dire la transformation du VHDL en schéma logique (flip-flops et portes logiques). Les synthétiseurs peuvent être fournis soit par un fabricant de FPGA (par exemple XST de Xilinx), soit par une société indépendante (par exemple Leonardo de Mentor Graphics).

Avant de lancer la synthèse, on doit configurer le script Matlab et spécifier le type de synthétiseur ainsi que la technologie utilisée (par exemple Spartan ou Virtex).

Pour certains synthétiseurs, par exemple XST, il est aussi possible de spécifier un attribut définissant la manière d'implémenter les multiplicateurs. L'attribut «lut» force le synthétiseur à utiliser les blocs lut (Look-up-Table/Table de vérité) des slices de la FPGA, le slice étant la cellule de base programmable des FPGA. Dans une Spartan 2, on dispose par exemple de 2 luts par slice. Chaque lut a 4 entrées et une sortie, ce qui correspond à une ROM asynchrone de 1 bit à 16 adresses. Le passage par un grand nombre de portes logiques provoque un temps de propagation des signaux important. Avec l'attribut «pipe_lut», on insère les registres ce qui limite les temps de propagation. Cette solution permet un gain de la fréquence d'horloge, au détriment d'une légère augmentation du nombre de cellules de base (slices) et du délai entre l'entrée et la sortie de la FPGA. Nous n'avons pas utilisé cet attribut, car on doit insérer manuellement des registres dans le code VHDL. L'attribut «block» force l'utilisation des multiplicateurs précâblés qui sont présentes sur certaines FPGA, tel que la Virtex, mais pas la Spartan. Ces multiplicateurs fortement optimisés à for-

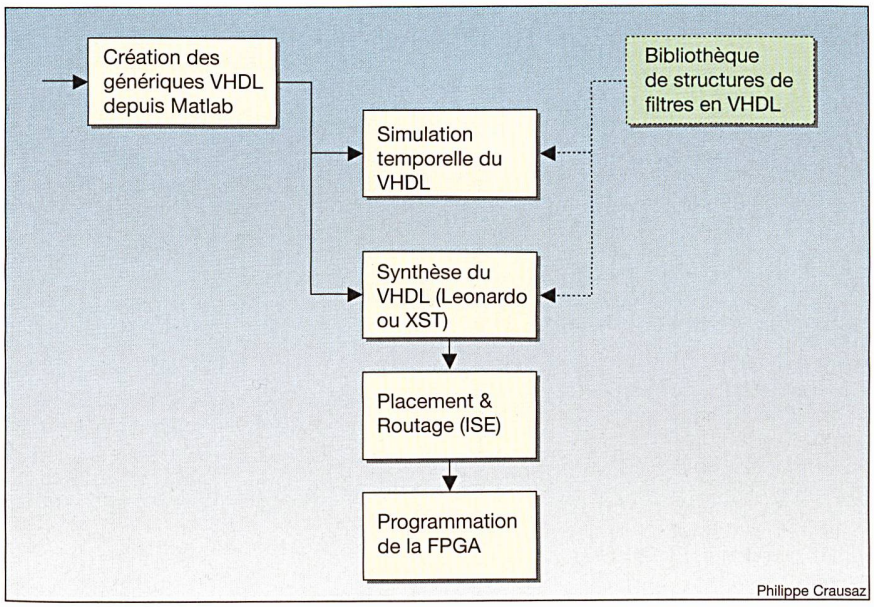

Figure 3 Etapes de la réalisation de la FPGA

mat fixe (18×18 bits) présentent l'inconvénient d'être placés sur la périphérie du circuit, ce qui peut entraîner, dans certains cas, un retard inacceptable sur les signaux après le placement et routage. L'attribut «auto» laisse le choix de l'implémentation au synthétiseur. En résumé, l'attribut pour une Spartan est uniquement le «lut», alors que pour une Virtex, les attributs «auto» et «block» peuvent être aussi utilisés.

Leornardo a beaucoup moins d'attributs, en effet il effectue l'optimisation en choisissant lui-même les attributs. Notre expérience est que le nombre de portes est plus faible (donc meilleur) avec Leonardo, mais que si l'on choisit avec attention les attributs, on arrive presque au même nombre de portes avec XST.

L'utilisation des logiciels de synthèse de FPGA n'est pas immédiate, et un soin dans son utilisation permet un gain de place. Les multiplications par 1 ou par une puissance de 2 sont correctement détectées et supprimées, respectivement remplacées par un décalage. En outre, quelquefois, en changeant le signe d'un coefficient et en remplaçant pour compenser l'addition par la soustraction (ou vice-versa), on gagne de la place (environ 5%). Pour garder la possibilité d'inverser le signe d'un coefficient, il faut veiller à donner suffisamment de bits pour exprimer ce coefficient, par exemple pour –64, il faut utiliser 8 bits au lieu de 7. En apparence, cela augmente le nombre de cellules de base, mais en réalité, on obtient une diminution. Comme autre exemple, si on doit multiplier par 31, il est plus avantageux de multiplier par 32 (ce qui se réduit à un décalage) et de soustraire ensuite l'opérande. Certains synthétiseurs effectuent automatiquement cette optimisation, d'autres pas.

Après synthèse, un logiciel par exemple ISE Xilinx permet d'implémenter ce schéma logique dans la FPGA (on parle de «placement and routage»). Toutes ces opérations (synthèses et «placement and routage») peuvent être lancées directement depuis Matlab à l'aide des scripts développés.

Résultats de la réalisation

Un ensemble de 6 filtres chacun du 6e ordre ont été conçus et réalisés, soit un total de 18 filtres du 2e ordre qui ont été chargés sur une Spartan 2 XC2S200 de Xilinx. Ces filtres ont permis la réalisation d'un équaliseur audio. La sortie de chacun des filtres était connectée à un atténuateur (diviseur par des puissances de 2). Un additionneur combine les 6 signaux de l'équaliseur. Ensemble, ils occupent 90% de la FPGA. Les filtres et la sortie finale ont été testés. Aussi bien les réponses fréquentielles que le bruit sont conformes aux simulations. Comme actuellement, on a des FPGA 25 fois plus grandes, on peut considérer qu'il est possible de mettre 450 filtres du 2e ordre sur une FPGA. La rapidité des FPGA a été démontrée, car même si cette application est basse fréquence, elle a été testée à plus de 50 MHz, montrant bien les possibilités des FPGA.

Conclusion

Les FPGA sont bien adaptées au traitement numérique du signal. Dans cette analyse, cela a été montré pour le cas particulier des filtres. Les dispositifs ont suffisamment de puissance pour permettre des traitements complexes en parallèle. Les outils de développement rendent le temps de développement raisonnable et la simulation permet un bon dimensionnement et une bonne analyse des performances. Les différents outils de synthèse n'ont pas tous les mêmes performances, parce qu'ils n'effectuent pas le même type d'optimisations.

Références

[1] S. Mitra: Digital Signal Processing A Computer Based Approach. Mc Graw Hill, 1998.

[2] A. Oppenheim, R. Schafer: Discrete-time Signal Processing. Prentice Hall, 1999.

Informations sur les auteurs

Ottar Johnsen a reçu son diplôme d'ingénieur électricien et le titre de Dr ès sciences techniques de l'EPFL en 1974 et 1979. De 1979 à 1983, il a été Member of Technical Staff, Bell Laboratories, Holmdel, Etats-Unis, et de 1983 à 1984 ingénieur système à Tricom, Berne. Depuis 1984, il est professeur de traitement du signal et de télécommunications à l'Ecole d'ingénieurs et d'architectes de Fribourg. Il a été chercheur invité à l'Université de Californie, Santa Barbara, et professeur invité à l'Ecole d'ingénieurs de Bergen, Norvège.
EIA-FR, Boulevard de Pérolles 80, case postale 32, 1705 Fribourg, ottar.johnsen@hefr.ch

Philippe Crausaz a obtenu son diplôme d'ingénieur en électricité à l'EPFL en 1984. Après avoir travaillé dans les circuits intégrés à Neuchâtel, il rejoint la HES-SO de Fribourg en 2000 en tant que collaborateur scientifique. Il s'est spécialisé dans le VHDL et les FPGA.
EIA-FR, Boulevard de Pérolles 80, case postale 32, 1705 Fribourg, philippe.crausaz@hefr.ch

Prof. ***Claude Magliocco*** a obtenu son diplôme de mécanique à l'ETS de l'Etat de Vaud de Lausanne en 1975 et d'électronique en 1981 à la HTL de Zürich. Après avoir travaillé chez BBC/ABB et Landis & Gyr/Siemens AG, notamment dans le développement d'ASIC, il exerce depuis 1992 une activité d'enseignement et de développement à l'EIA-FR. Il dirige ou participe à plusieurs projets HES-SO, un projet CTI avec la maison Syderal à Gals et un mandat avec l'université de Fribourg. Les plus récents sont Fast_PCB, System on Chip (SOC), Virtual Guiding System pour hélicoptère (VGS), Can/Spw/PCI Bus Coupler et Space-Ergometer.
EIA-FR, Boulevard de Pérolles 80, case postale 32, 1705 Fribourg, claude.magliocco@hefr.ch

Prof. ***Denis Prêtre*** est ingénieur EPFZ en génie électrique. Après avoir obtenu son diplôme en 1989, il a travaillé six ans chez Landis & Gyr (Genève) dans le domaine de l'informatique embarquée et trois ans chez Vibro-Meter à Fribourg dans le domaine du traitement du signal. Il est professeur à la HE-Arc depuis janvier 2000. Ses domaines d'activité sont le portage d'algorithmes de traitement du signal sur DSP et sur FPGA. Il travaille actuellement sur un important projet industriel où des algorithmes complexes sont exécutés en temps réel par deux FPGA de type VIRTEX-IV et un DSP de type Blackfin. Une autre activité récente a été de réaliser des cartes embarquées «Voice Over IP – WLAN» entièrement autonomes.
HE-Arc, 2610 *St-Imier, denis.pretre@he-arc.ch*

Prof. Dr ès sc. ***François Corthay*** est ingénieur EPF en microtechnique. Il a obtenu son diplôme de l'EPFL en 1985 puis son doctorat en sciences à l'Université de Neuchâtel en 1992. Il exerce une activité d'enseignement et de développement à la HEVs.
Haute Ecole valaisanne, case postale 2134, 1950 Sion 2, francois.corthay@hevs.ch

Dr ès sc. techn. ***Gilbert Maître*** a obtenu un diplôme d'ingénieur en électrotechnique de l'EPFZ en 1987 et un titre de docteur ès sciences de l'Université de Neuchâtel en 1994. Il a travaillé comme chercheur à l'institut IDIAP de Martigny de 1994 à 1998. De 1998 à 2005, il a enseigné le traitement du signal à l'Ecole d'ingénieurs du Valais, devenue entre temps Haute Ecole valaisanne. Depuis 2005, il continue à effectuer des tâches de recherche et d'enseignement auprès de la HEVs et travaille d'autre part comme informaticien à la centrale du Réseau romand des bibliothèques.
Haute Ecole valaisanne, case postale 2134, 1950 Sion 2, gilbert.maitre@hevs.ch

Dr ès sc. techn. ***Michel Kocher*** est professeur à l'Ecole d'ingénieurs de Genève. Diplômé ingénieur électricien en 1979, Michel Kocher obtient son doctorat en 1983 à l'Ecole polytechnique fédérale de Lausanne. Il poursuit ses activités de recherche en codage d'images aux laboratoires Bell aux Etats-Unis, puis à l'Institut de recherche d'Atlas Copco. Il est l'auteur ou le coauteur d'une dizaine de publications sur le codage d'images.
Ecole d'ingénieurs de Genève, 4, rue de la Prairie, 1202 Genève, michel.kocher@hesge.ch

Zusammenfassung

Field Programmable Gate Arrays (FPGA) kontra Digital Signal Processors (DSP)

FPGA als Alternative zu DSP bei der Signalaufbereitung. FPGAs weisen sehr hohe Verarbeitungsleistungen auf, sind sie doch in der Lage, Multiplikationen und Additionen bei über 100 MHz auszuführen, wobei sich Hunderte Multiplikatoren und Addierer parallel einsetzen lassen. Multiplikatoren lassen sich individuell optimieren, indem die jeweilige Daten- und Koeffizientenbreite einzeln eingestellt wird. Mit den heute verfügbaren Entwicklungstools erfordert das Konzept nur wenig mehr Aufwand, als dies bei Prozessoren der Fall ist.