Synthèse d'Architecture Multi-horloges pour la Conception Faible Consommation sur FPGA

GHIZLANE LHAIRECH-LEBRETON, PHILIPPE COUSSY, ERIC MARTIN
Lab-STICC, Université de Bretagne-Sud
prenom.nom@univ-ubs.fr

Thème – Synthèse de haut niveau, conception sur FPGA, faible consommation.

Problème traité – Conception faible consommation sur FPGA.

Originalité – Réduction de la consommation dans les FPGA en utilisant la synthèse de haut niveau hiérarchique pour générer automatiquement des architectures à horloges multiples, gel d'horloge...

Résultats –Différentes applications du traitement de signal ont été synthétisées et pour les mêmes contraintes de débit et un surcoût en surface négligeable. Un gain en consommation de 11% en moyenne a été mesuré.

Résumé – L'optimisation de la consommation est devenue un challenge des plus importants dans la conception des systèmes numériques. Bien que les FPGAs soient de plus en plus utilisés, ils sont toujours considérés comme peu efficaces en termes de consommation comparés aux ASICs. Dans cet article, nous adressons ce problème en proposant une approche de conception basée sur la synthèse de haut niveau. Le flot de conception présenté permet de générer automatiquement des architectures à multiple domaines d'horloge afin de réduire la fréquence d'horloge de certaines parties du circuit, de réduire la complexité de l'arbre d'horloge, de diminuer le nombre de longs fils qui ont un impact important sur la consommation dans les circuits programmables et de suspendre l'horloge lorsque cela est possible. Les premières expériences qui ont été réalisées sur des applications de traitement du signal et sur un FPGA Virtex-5 de chez Xilinx montrent qu'une réduction de 11% de la consommation peut être atteinte.

1 Introduction

La consommation d'énergie dans les FPGAs est directement liée à la tension d'alimentation, la fréquence d'horloge et le taux d'activité [1]. Cette consommation, comme indiqué dans [2], est dominée par les interconnexions (45% de la puissance totale) et par le réseau de distribution d'horloge (40%) alors que les blocs logiques ne consomment, quant à eux, que 15%. Ainsi, les interconnexions et le réseau de distribution d'horloge sont les principaux contributeurs à la consommation. Traditionnellement, les architectures que l'on retrouve dans les FPGAs sont « plates » i.e. sans hiérarchie et ne contiennent qu'un seul domaine d'horloge. L'horloge doit donc être répartie sur la totalité du circuit afin d'alimenter les composants nécessitant le signal d'horloge (par exemple, les registres). Ceci aboutit à l'utilisation d'un réseau complexe de distribution d'horloge qui s'étend sur une large surface. De plus, cet arbre de distribution de l'horloge est toujours actif bien que toutes les parties du circuit ne fonctionnent pas en même temps.

Nous adressons la conception faible consommation des applications de traitement numérique du signal (TNS) sur FPGA. Pour ce faire, nous utilisons la synthèse de haut niveau hiérarchique [3] afin de générer automatiquement des architectures constituées de plusieurs blocs ayant chacun leur propre horloge. De plus, grâce aux composants de gestion d'horloges (Digital Clock Manager DCM chez Xilinx ou Phase-Locked Loop PLL chez Altera) présents dans les FPGAs actuels [4][5] la communication entre les blocs peut être synchrone. Notre approche permet ainsi de (1) réduire la fréquence d'horloge dans certaines parties du circuit, (2) mettre en œuvre le gel d'horloge, (3) réduire le nombre de longs fils dans le circuit final au travers de la hiérarchie, (4) réduire la complexité du réseau d'horloge grâce à l'utilisation de plusieurs domaines d'horloge et (5) réduire le surcoût en surface par l'introduction d'une nouvelle architecture synchrone multi-horloge au lieu d'utiliser des architectures Globalement Asynchrones Localement Synchrones GALS [6] classiquement utilisées dans les MPSoC.

2 Etat de l'art

Dans cette section, nous présentons les travaux appartenant aux trois principaux domaines de recherche liés à notre approche à savoir : (1) la synthèse hiérarchique de haut niveau (H-HLS), (2) les architectures multi-horloges et (3) les approches de haut niveau pour la conception faible consommation sur FPGA.

Ils existent plusieurs travaux sur la synthèse hiérarchique dans la littérature. Dans [7] une technique permettant le partage de ressources entre différents niveaux hiérarchiques est présentée. Dans [8], les auteurs introduisent un modèle de représentation comportementale pour automatiser l'utilisation de la synthèse hiérarchique et permettre de spécifier localement des contraintes d'ordonnancement entre des opérations. Dans [9], la caractérisation des protocoles d'interface es formalisé sous la forme d'une machine à états finis qui spécifie l'ensemble des données et des signaux de contrôle requis pour l'intégration d'un composant complexe. Cependant, dans tous ces travaux, la synthèse hiérarchique est uniquement utilisée pour faire face à la complexité des systèmes et aux problèmes de réutilisation des composants. L'optimisation de la consommation d'énergie, quand à elle, n'est pas adressée. Dans [10], les auteurs présentent une approche de synthèse hiérarchique pour la conception d'architectures optimisées en consommation et en surface à partir de graphes de flux de données hiérarchiques sous contraintes de débit. Néanmoins, ni la conception à horloge multiples ni le gel d'horloge ne sont adressés.

Les travaux sur les architectures à horloge multiples ont principalement été menés sur les systèmes Globalement Asynchrone Localement Synchrone (GALS) [6]. Les architectures GALS ont été proposées afin notamment d'améliorer les performances dans les architectures (MP)SoC ciblant la technologie ASIC. Les travaux dans ce domaine peuvent être divisés en trois catégories : le partitionnement, les dispositifs de communication, et les architectures dédiées. Seuls les travaux visant la conception de dispositifs de communication à faible temps de latence seront présentés ici. Dans [11], un wrapper de synchronisation optimisé en surface est introduit pour les architectures LIS (Latency Insensitive System). Le wrapper est utilisé pour garantir une communication insensible au temps de latence entre deux blocs ayant la même fréquence d'horloge. Afin d'assurer la communication au sein d'une architecture à domaine d'horloge multiples, les auteurs de [12] ont proposé une FIFO à horloge mixte de faible latence, alors que dans [13] un wrapper de communication point à point est présenté. Une architecture de wrapper asynchrone qui supporte un transfert de données style poignée de main à 4-phase est également proposée dans [14] et [15].

Quelques rares approches de synthèse automatique de haut niveau pour la conception faible puissance sur FPGA ont été proposées. Ainsi, une technique de réduction de la tension d'alimentation a été proposée par [16] pour contrôler l'alimentation du FPGA en utilisant un contrôleur de tension dans un ordinateur personnel. Cependant, cette méthode permet la réduction de la tension de l'ensemble du système. Les auteurs de [17] proposent des circuits FPGA conçus avec un dispositif de double tension Vdd/Vt afin de réduire efficacement la puissance dynamique et statique. Dans [18], un flot de synthèse de haut niveau faible puissance, appelé LOPASS est présenté. Deux algorithmes ont été proposés pour la réduction de la consommation d'énergie : (1) un recuit simulé qui effectue simultanément la sélection des ressources, l'assignation des opérateurs, l'ordonnancement, l'assignation des registres, et la génération de chemin de données et (2) un algorithme amélioré de mariage sur graphe bipartite pondéré pour réduire le nombre total de multiplexeurs.

A notre connaissance, aucun des travaux existants n'a proposé un flot de conception entièrement dédié à la conception faible puissance sur FPGA considérant à la fois : les interconnexions, le réseau d'horloge et la hiérarchie. Nous avons pour objectif de réduire automatiquement la consommation d'énergie par (1) la réduction de la fréquence d'horloge, (2) l'utilisation du gèle d'horloge, (3) la réduction de la quantité de longs fils dans l'architecture finale par le biais de la hiérarchie (4) la diminution de la complexité du réseau horloge grâce à l'utilisation de multiples domaines d'horloge et (5) la réduction du surcoût en surface en introduisant une nouvelle architecture multi-horloge synchrone au lieu d'utiliser les systèmes GALS. Dans ce cadre, nous proposons une approche de synthèse de haut niveau automatisée pour la conception faible consommation sur FPGA.

3 Approche proposée

Notre flot de synthèse de haut niveau, présenté en figure 1, prend en entrée une spécification décrite en langage C / C++ ainsi qu'une contrainte de débit sous la forme d'une cadence (i.e. un intervalle d'initiation) et d'une période d'horloge nommée *horloge système*. La spécification d'entrée peut contenir des *opérations élémentaires* spécifiées en utilisant les symboles du langage (+,-...) et des *opérations composites* exprimées par des appels de fonctions. La bibliothèque d'opérateurs contient des opérateurs simples et complexes. Les opérateurs simples sont utilisés pour implémenter des opérations élémentaires (registres, additionneurs…). Les opérateurs complexes sont utilisés pour implémenter des opérations composites. Un composant complexe est un composant séquentiel composé d'un contrôleur et d'un chemin de données.

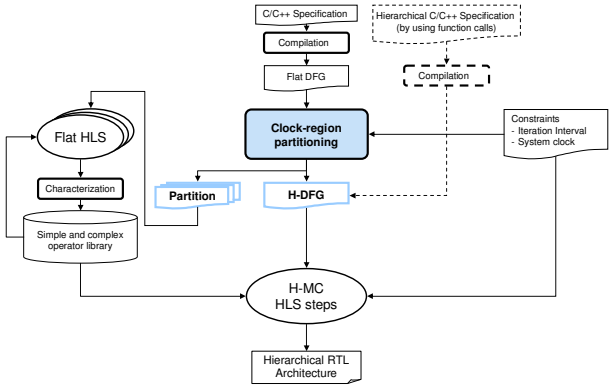

Figure 1 : Flot de synthèse proposé

Nous traduisons dans un premier temps la spécification initiale en un graphe flot de données hiérarchiques (H-DFG) en fonction des directives de compilation : le concepteur peut (1) mettre en ligne les fonctions en utilisant dans le code source le mot clé *inline* ou (2) conserver les appels de fonctions. Lorsqu'il existe au moins une fonction non mise en ligne, nous effectuons une synthèse hiérarchique. Le H-DFG qui est généré contient dans ce cas des nœuds hiérarchiques représentant les appels de fonction. L'étape d'allocation définit ensuite le nombre d'opérateurs a priori nécessaires (borne inférieure) à mettre en œuvre pour respecter la contrainte de débit. L'ordonnancement repose sur un algorithme de liste où la mobilité des opérations est utilisée comme critère de priorité [19]. L'assignation des opérations sur les opérateurs et des données dans les registres est réalisée à l'aide d'un algorithme MWBM [20] dont l'objectif est de minimiser le cout des interconnexions (multiplexeurs).

Afin d'optimiser la consommation, notre approche synthétise dans un premier temps chaque fonction sous la forme d'un opérateur complexe si celui-ci n'est pas déjà présent dans la bibliothèque. Si cet opérateur complexe contient des operateurs simples multi-cycle, alors son horloge sera ralentie (les opérateurs multi-cycle requièrent par définition plus d'un cycle pour délivrer leurs résultats). De ce fait, les opérateurs complexes sont synthétisés en amont de l'application et sont cadencés avec des fréquences d'horloge plus faibles que celle de l'horloge système quand cela est possible. Nous obtenons ainsi, une synthèse automatique d'architectures hiérarchiques à horloges multiples.

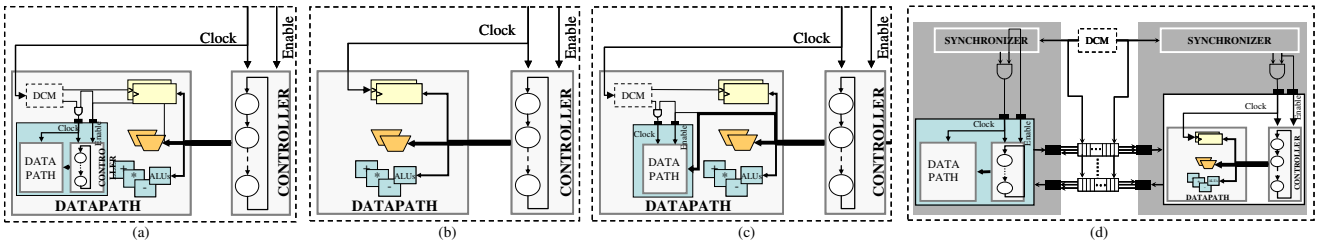

Figure 2. Architecture (a) à contrôle distribué (b) Flat (c) contrôle centralisé (d) LIS/GALS

Dans notre flot de synthèse, un opérateur complexe a son propre contrôleur et chemin de données. Il peut donc fonctionner indépendamment de son environnement. Par conséquent, lors de la génération de l'architecture RTL, nous générons un signal de validation de la commande afin d'activer l'opérateur complexe lorsque ce dernier doit fonctionner. Ce même signal de validation est utilisé pour geler l'horloge du composant lorsqu'il est inactif. La communication synchrone entre un composant complexe et son environnement se fait grâce à des registres générés lors de l'étape d'assignation des registres. Afin d'assurer la synchronisation entre différent domaines d'horloge, nous utilisons des ressources dédiées fournies par les FPGA récents. Ce dispositif spécifique est représenté par la boîte en pointillé en figure 2 appelé DCM (Digital Clock Manager). L'architecture ainsi générée est nommée architecture à contrôle distribué et à horloges multiples (DC-MuC). Elle est composée de plusieurs niveaux hiérarchiques. Le niveau supérieur fonctionne à la fréquence d'horloge spécifiée par le concepteur. Chaque niveau hiérarchique inferieur a une fréquence d'horloge sous-multiple à la fréquence système.

4 Expériences

Nous avons validé notre approche en utilisant plusieurs applications du domaine du traitement numérique de signal. Nous avons synthétisé pour cela plusieurs architectures pour chaque application, afin de comparer notre approche à celles proposées dans l'état de l'art, à savoir, les architectures mono horloge et non hiérarchiques. Afin de générer les signaux d'horloge de chaque bloc, nous avons utilisé les gestionnaires d'horloge DCM disponibles sur notre cible Xilinx XC5VLX110 [4]. Un DCM consomme 44mw à 250Mhz.+0.2mW/Mhz. La consommation a été mesurée en utilisant un analyseur de puissance DC qui fournit une tension/courant programmable ainsi que des dispositifs de mesure.

Nous avons synthétisé plusieurs approches architecturales pour chaque exemple, afin de comparer notre approche à l'état de l'art: l'architecture dite Flat est composée d'un seul contrôleur réalisé par une machine à états finis (FSM) et un chemin de données contenant de simples opérateurs, registres et multiplexeurs (voir figure 2.b). Les architectures Flat sont obtenues par la mise en ligne de toutes les fonctions, menant à un DFG non hiérarchique. Cela signifie que la hiérarchie contenue dans la spécification n'est ni maintenue dans le modèle intermédiaire, ni dans l'architecture finale. Lorsque les fonctions ne sont pas mises en ligne, elles peuvent être mises en œuvre par deux types d'opérateurs: « *Datapath* » *(DP)* et « *Datapath + contrôleur*» *(DP+C)*. Ce dernier type est la structure que nous utilisons dans notre approche et a déjà été introduit dans la section précédente. Dans une architecture utilisant des *DP*, les modules ne peuvent pas fonctionner indépendamment du niveau hiérarchique supérieur car ils ne possèdent pas leur propre contrôleur. Ainsi, lors de la synthèse d'un module *DP*, le contrôleur est abstrait en un ensemble de mots de commande (un pour chaque étape de contrôle) représentant la spécification du contrôleur [9]. Cette spécification est ensuite stockée avec le *DP* dans la bibliothèque d'opérateur. Ainsi, lors de la synthèse du niveau hiérarchique supérieur toutes les informations de contrôle nécessaires pour effectuer l'opération requise sont fournies au chemin de données. Comme pour l'architecture utilisant des *DP+C*, les registres sont générés pour assurer une communication synchrone et le signal d'activation pour le gèle d'horloge. Une architecture basée sur des blocs intégrant des opérateurs *DP* et ayant le même signal d'horloge est une architecture mono-horloge à contrôle centralisé nommée CC-MoC. Lorsque plusieurs signaux d'horloge sont utilisés, l'architecture est multi-horloge synchrone à contrôle centralisée CC-MuC (voir la figure 2. c).

Afin de se comparer avec les architectures LIS/GALS [6]/[21], les modules *DP+C* ont été intégrés comme dans les architectures LIS ou GALS. Pour cela, ils doivent être capables de fonctionner de manière autonome et de communiquer via des FIFO. Pour rendre un *DP+C* autonome, nous avons utilisé le wrapper de synchronisation proposés dans [11] à la fois pour le niveau hiérarchique supérieur et le composant complexe (comme détaillé dans la figure 2.d.). Chaque entrée/sortie des modules est une entrée/sortie du niveau hiérarchique supérieur. L'ordonnancement et l'assignation sont effectués indépendamment de l'état du composant complexe. Par ailleurs, l'utilisation de communication asynchrone suppose que les horloges puissent être désynchronisées nécessitant ainsi des FIFOs bi-horloges. Nous avons automatiquement générés des FIFOs bi-horloges similaires au modèle de [12].

Les applications TNS que nous avons utilisées pour nos premières expériences sont les suivantes : une FFT (Fast Fourier Transform) et une DCT (Discrete Cosine Transform) avec plusieurs complexités de calcul. Nous avons fait varier les tailles de la DCT de 16 * 16 à 32*32 et de la FFT de 16 à 32 points. Les figures 3 et 4 illustrent les consommations en puissance mesurées et la surface par application et par type d'architecture. Les résultats que nous avons obtenus montrent que la distribution du contrôle dans l'architecture DC-MuC réduit la consommation et la surface du circuit final (5% de réduction de la surface par rapport à l'architecture Flat et jusqu'à 11% de réduction de puissance). L'architecture GALS réduit la consommation (jusqu'à 7%),

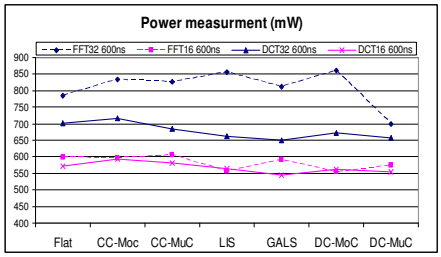

Figure 3. Les mesures de consommation pour les FFT et DCT

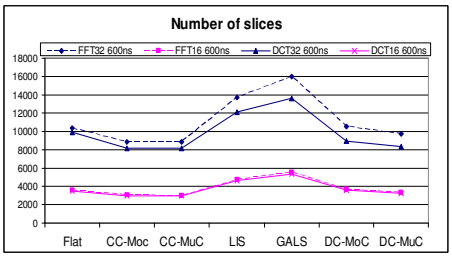

Figure 4. Résultats en surface pour les FFT et DCT

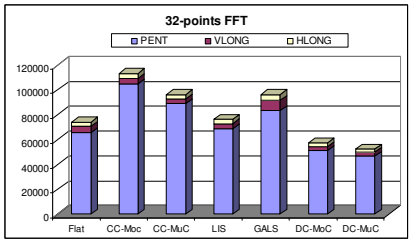

Figure 5. Nombre de fils obtenu pour la FFT 32-points (600ns)

mais présente 50% de surcoût en surface suite à l'utilisation des FIFOs et d'un processeur de synchronisation [5].

Dans l'architecture DC-MuC que nous proposons, le nombre de longs fils peut être réduit de 30% (voir figure 5). Comme mentionné dans l'introduction les interconnexions et les longs fils présentent une composante majeure de la consommation d'énergie. Deux grandes catégories de ressources de routage existent dans les FPGA Xilinx : les fils dits locaux qui sont des connexions courtes et les longs fils. Le détail du nombre de fils peut être extrait de l'outil ISE. Tous les fils VLONG, HLONG et PENT peuvent être considérés comme des connexions longues. Les GENERIC, DOUBLE et GLOBAL sont des connexions courtes [22]. L'évolution de chaque catégorie de fils est donnée dans la figure 5. Dans l'architecture basée sur des *DP+C*, les fils longs ont été réduits. L'analyse des résultats montre une réduction du nombre de longs fils est réduit en utilisant l'architecture basée sur des *DP+C* (au plus 29% pour le multi-horloge synchrone, contre 22% pour le mono-horloge) et une augmentation de 29% dans l'architecture LIS/GALS.

5 Conclusion

Notre approche, qui repose sur la synthèse d'architecture, vise à tirer profit de la structure du FPGA afin de diriger les techniques d'optimisation à haut niveau. Les résultats expérimentaux montrent que l'utilisation du contrôle distribué, de la communication synchrone, de plusieurs domaines d'horloge et du gel d'horloge, permet de réduire la consommation en énergie tout en conservant un surcoût en surface raisonnable.

Dans cette première approche, les architectures hiérarchiques ont été générées en utilisant des appels de fonction dans la spécification algorithmique. Les travaux en cours visent à générer automatiquement un H-DFG en utilisant une méthode de partitionnement automatique. Cette méthode permet d'identifier les domaines d'horloge de manière optimisée et de générer un DFG hiérarchique sans avoir recours aux appels de fonction dans la spécification initiale. Dans nos travaux futurs nous étudierons le couplage de notre approche avec une technique dual-Vdd/dual-Vth afin d'optimiser la consommation d'énergie en réduisant aussi la consommation statique.

6 Bibliographie

[1] C. Piguet "Low Power Electronics Design", CRC Press, 2005

[2] Degalahal, V. et al. "Methodology for high level estimation of FPGA power consumption". ASP-DAC '05.

[3] "An Introduction to High-Level Synthesis", P. Coussy, G. Gajski, A. Takach, M. Meredith Special issue on High-Level Synthesis, IEEE Design and Test of Computers, Vol. 26, Issue 4, July/August, 2009

[4] xilinx.com/support/documentation/ip_documentation/dcm_module.pdf

[5] www.altera.com/support/devices/pll_clock/pll-overview.html

[6] Advanced Research in Asynchronous Circuits and Systems, 2000.

[7] Bringmann and W. Rosenstiel, "Resource sharing in hierarchical synthesis," Int. Conference on Computer-Aided Design, 1997.

[8] Ly, T., Knapp, D., Miller, R., and MacMillen, D."Scheduling using behavioral templates", In Proceedings of the 32nd Annual ACM/IEEE Design Automation Conference, June 12 - 16, 1995.

[9] Fan, N., Chaiyakul, V., and Gajski, D. D., "Usage-Based characterization of complex functional blocks for reuse in behavioral synthesis". In Proceedings of the 2000 Asia and South Pacific Design Automation Conference, ASP-DAC '00.

[10] G. Lakshminarayana and N.K. Jha. High-level synthesis of power-optimized and area-optimized circuits from hierarchical data-flow intensive behaviors. IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems, 18(3), March 1999.

[11] Bomel, P., Martin, E., and Boutillon, E. "Synchronization Processor Synthesis for Latency Insensitive Systems". In Proceedings of the Conference on Design, Automation and Test in Europe 2005.

[12] Chelcea, T. and Nowick, S. M. "A Low-Latency FIFO for Mixed-Clock Systems". In Proceedings of the IEEE Computer Society Annual Workshop on VLSI, 2000.

[13] Moore et al. "Point to Point GALS Interconnect", IEEE ASYNC, 2002

[14] David Bormann, et al . Asynchronous wrapper for heterogeneous systems. In Proceedings of. ICCD, 1997.

[15] J.Muttersbach, T.Villiger, W.Fichtner. Practical design of globally asynchronous locally synchronous systems. In Proc. Int. Symp. On Advanced Research in Asynchronous Circuits and Systems, 2000

[16] C. Chow, L. Tsui, P. Heng, W. Leong, W. Luk and S. Wilton, "Dynamic Voltage Scaling for Commercial FPGAs", IEEE International Conference on Field-Programmable technology, 2005:

[17] F. Li, Y. Lin, L. He and J. Cong, "LowPower FPGA Using Predefined DualVdd/DualVt Fabrics", FPGA'04, February 2224,2004

[18] Chen, D., Cong, J., and Fan, Y, "Low-power high-level synthesis for FPGA architectures", International Symposium on Low Power Electronics and Design, 2003

[19] GAUT: A High-Level Synthesis Tool for DSP applications, P. Coussy, C. Chavet, P. Bomel, D. Heller, E. Senn, E. Martin, "High-Level Synthesis: From Algorithm to Digital Circuits", Springer, Berlin, Germany, 2008

[20] "High-Level Synthesis for Designing Multi-mode Architectures" C. Andriamisaina, P. Coussy, E. Casseau, C. Chavet, IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems (TCAD), November 2010, Vol. 29, Issue 11.

[21] L. P. Carloni, K. L. McMillan, and A. L. Sangiovanni-Vincentelli, ''Theory of Latency-Insensitive Design,'' IEEE Trans. on Computer Aided Design of Integrated Circuits and Systems, 20(9):18, Sept. 2001

[22] www.xilinx.com