2010 International Conference on Field Programmable Logic and Applications

# A Cost-Effective Technique for Mapping BLUTs to QLUTs in FPGAs

Marcus Ritt, Carlos Arthur Lang Lisboa, Luigi Carro
Instituto de Informática, PPGC - UFRGS
Email: {marcus.ritt,carlos.arthur.lisboa,carro}@inf.ufrgs.br

Cristiano Lazzari
ALGOS - INESC-ID
Email: lazzari@inesc-id.pt

***Abstract*—Quaternary logic has shown to be a promising alternative for implementing FPGAs, since voltage mode quaternary circuits can reduce the circuits' cost and at the same time reduce its power consumption. In this paper, we study the implementation of circuits in quaternary logic. To obtain cost-effective implementations of quaternary circuits, we propose a mapping from binary to quaternary circuits based on integer linear programming. Our results show that the expected improvements can be achieved, reducing, in average, the number of transistors by 27% and the number of nets by 19%, compared to a binary implementation.**

## I. Introduction

Field Programmable Gate Arrays (FPGAs) are very convenient and flexible hardware platforms for the implementation of these systems, helping designers to cope with one of the more demanding requirements currently imposed on the industry: time to market. Their flexibility comes at a price: in order to allow the many different interconnection schemes of the increasing number of devices in a single FPGA, an enormous area of the circuit must be used to implement all the required switches and wires.

As technology evolves according to Moore's Law, providing ever smaller, faster, and lower voltage devices, the amount of interconnections inside a single chip is increasing significantly [1]. When considering FPGAs, this problem becomes even more critical, since the huge amount of interconnections impacts not only the circuit delay [2], but also the power consumption and area [3].

Multiple-valued logic (MVL) has received increased attention in the last decades because of the possibility to represent the information with more than two discrete levels. As a consequence, there is a possibility to increase the amount of information transferred using a single wire [4].

Most of the efforts in this direction, however, propose the use of current-mode circuits or hybrid implementations of CMOS to build the configurable logic blocks (CLBs) [5]–[7]. While those solutions provided significant area reduction, their high static power consumption and extreme manufacturing complexity did not make them viable alternatives as a replacement for standard CMOS designs.

In order to overcome the inconveniences of current-mode circuits and hybrid implementations, the use of voltage-mode CMOS logic to implement quaternary look-up tables has been proposed in [8], and shown to maintain the compaction allowed by multi-valued logic. First steps to tackle the challenge of low power and high density FPGAs using this voltage-mode MVL technique were presented in [9], where an arithmetic-oriented configurable logic block was proposed.

Our contribution in this paper is a mapping tool to allow automatic implementation of quaternary standard arithmetic circuits. Since there is no quaternary design flow available, we propose a method based on integer linear programming to map binary circuits to quaternary circuits. Section II discusses differences between binary and quaternary look-up table structures. The proposed approach is described in detail in Section III, and the obtained results are discussed in Section IV. The development of quaternary devices in future technologies is discussed in Section V. Section VI summarizes our conclusions and points to future work.

## II. Binary and Quaternary LUTs Overview

General Lookup Tables (LUT) are basically memories, which implement a given logic function. Values are initially stored in the lookup table structure, and once inputs are applied, the logic value in the addressed position is assigned to the output. The capacity of a LUT C is given by $|C| = n \times b^k$ where $n$ is the number of outputs, $k$ is the number of inputs and $b$ is the number of logic values. For example, a 6-input binary lookup table with one output is able to store $1 \times 2^6 = 64$ Boolean values. For the purpose of this work, 6-input 1-output LUTs ($k = 6$,$n = 1$) are used because they are the most complex ones used in currently available FPGAs, such as the Xilinx Spartan-3E [10], and the Altera Ciclone III [11].

A binary function implemented by a Binary Lookup Table (BLUT) is defined as $f$: $\mathbf{B}^k \rightarrow \mathbf{B}$, over variables $X = (x_0, \ldots, x_i, \ldots, x_{k-1})$, where each variable $x_i$ represents a Boolean value **B**. The total number of different functions is given by $|F| = b^{|C|}$ where $b = |\mathbf{B}|$ (i.e. $b = 2$ in the binary case).

Quaternary functions are basically generalizations of binary functions. A quaternary function is defined as $g$: $\mathbf{Q}^k \rightarrow \mathbf{Q}$, over quaternary variables $Y = (y_0, \ldots, y_i, \ldots, y_{k-1})$, where the values of a variable $y_i$, as the values of the function $g(Y)$, can be in $\mathbf{Q}$= $\{0, 1, 2, 3\}$. The number of functions is also given by $|F| = b^{|C|}$, where $b = 4$.

## III. Mapping binary to quaternary circuits

In this section we discuss the problem of mapping a circuit of binary 6-to-1 LUTs (BLUTs) to quaternary 3-to-1-LUTs (QLUTs). The input circuit is represented as a directed acyclic

1946-1488/10 $26.00 © 2010 IEEE
DOI 10.1109/FPL.2010.72

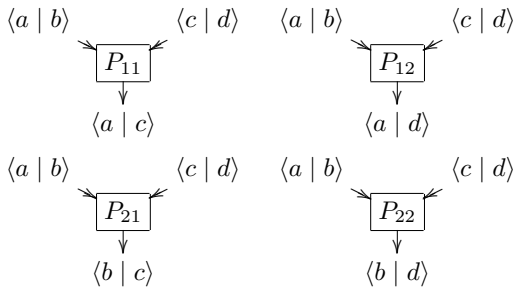


Figure 1. Projections used to combine quaternary nets.

graph (DAG) $G = (B, D)$ over the set of BLUTs $B$ with edges $D \subseteq B \times B$. Each BLUT $i \in B$ has a corresponding set of inputs $I(b)$, satisfying $|I(b)| \leq 6$ and a single output $O(b)$, such that $O(b) \in I(c)$, for a net $(b, c) \in D$.

The goal of mapping binary to quaternary logic is to minimize the number of LUTs and the number of used nets. Necessary conditions for mapping two BLUTs to the same QLUT are that 1) their joint set of inputs is not larger than six and 2) we have to guarantee that the mapping satisfies the partial ordering constraints given by the circuit's DAG.

The proposed mapping process is performed in two stages. In the first stage, we map BLUTs to QLUTs, considering only input and ordering constraints, and minimizing the number of QLUTs used. In the second stage, we map binary nets to quaternary nets. For a quaternary net, which results from pairing of two binary nets $a$ and $b$ we write $\langle a \mid b\rangle$. To make this two-stage approach possible, we have to allow combining two quaternary nets $\langle a \mid b\rangle$ and $\langle c \mid d\rangle$ to produce, for example, a quaternary net $\langle a \mid c\rangle$. This is necessary, if some quaternary input is not available according to the mapping defined in the first stage. We call this a *projection*, and use the four types of projections as shown in Figure 1.

The two stages are modeled as integer linear programs. For the first stage, let $B = [1, n]$ be the set of BLUTs. We need at most $n$ QLUTs and define the set of QLUTs as $Q = [1, n]$. Let $x_{ijk}$ indicate a mapping of BLUTs $i$ and $j$ to QLUT $k$, and let $J \subseteq B^2$ be the set of joinable gates (i.e. $J = \{(b_1, b_2) \mid b_1 \neq b_2 \in B, |I(b_1) \cup I(b_2)| \leq 6\}$). To allow mapping a single BLUT to a QLUT, we add an "empty" gate with index 0 and extend the set of joinable gates to $J_0 = J \cup \{(0, b) \mid b \in B\}$. Then we formulate the mapping problem as the following $0/1$ integer program:

$$\textbf{min.} \quad \sum_{(i,j)\in J_0, k\in Q} x_{ijk} \tag{1}$$

$$\textbf{s. t.} \quad \sum_{j:(i,j)\in J_0, k\in Q} x_{ijk} = 1, \quad \forall i \in B, \tag{2}$$

$$\sum_{(i,j)\in J_0} x_{ijk} \leq 1, \quad \forall k \in Q, \tag{3}$$

$$\sum_{\substack{j:(i_1,j)\in J_0,\\ k\in Q}} k\, x_{i_1jk} \leq \sum_{\substack{j:(i_2,j)\in J_0,\\ k\in Q}} k\, x_{i_2jk}, \quad \forall (i_1, i_2) \in D, \tag{4}$$

$$x_{ijk} \in \{0, 1\}, \quad \forall (i, j) \in J_0, k \in Q.$$

The objective function (1) minimizes the number of QLUTs. Restriction (2) guarantees that every BLUT is mapped to some QLUT, while restriction (3) ensures that each QLUT receives at most one pair of BLUTs. Restriction (4) makes sure that the mapping satisfies the ordering constraints defined by the binary circuit: for every pair of dependent BLUTs $(i_1, i_2) \in J$, $i_1$ has to be mapped on a QLUT of index not greater than the index of $i_2$. The formulation has $O(n^3)$ variables and $O(n^2)$ restrictions.

In the second stage, we map the binary nets pairwise to quaternary nets. This is accomplished by defining a pairing of the input nets of the circuit, a pairing of the input nets of each QLUT and a pairing of the output nets. For a given pairing, not every input pair of every QLUT is available as a primary input pair or an output of another QLUT. In this case, we have to insert an additional projection, to provide the necessary quaternary net. This is always possible, since the mapping of the first stage respects the partial order of the circuit.

Let $I$ be the set of primary binary input variables, $Q_j$ be the set of binary input variables of QLUT $j \in Q$ and $O$ be the set of primary input output variables. We define matching variables $i_{uv} \in \{0, 1\}$ for all $u \neq v \in I$, $q_{juv} \in \{0, 1\}$ for all $u \neq v \in Q_j$, $j \in Q$, and $o_{uv} \in \{0, 1\}$ for all $u \neq v \in O$. The matching variables indicate which pairs of the inputs, QLUT inputs, or outputs are mapped to the same quaternary wire. We further set $i_{uv} = 1$ for each pair available as the output of some QLUT.

To quantify the number of necessary projections, we further define auxiliary variables $r_{uv} \in \{0, 1\}$ for all $u \neq v \in Q_j$, $j \in Q$ indicating that the quaternary input $\langle u \mid v\rangle$ on QLUT $j$ is not available as an input or output of some other QLUT, and variables $p_{uv} \in \{0, 1\}$, for all $u \neq v \in O$ indicating that the quaternary output $\langle u \mid v\rangle$ is not available. Then, the second stage problem can be formulated as the $0/1$ integer program:

$$\textbf{min.} \quad \sum_{\substack{u,v\in O\\ u\neq v}} r_{uv} + \sum_{\substack{u,v\in O\\ u\neq v}} p_{uv} \tag{5}$$

$$\textbf{s. t.} \quad \sum_{v\in I} i_{uv} = 1, \quad \forall u \in I, \tag{6}$$

$$\sum_{v\in Q_j} q_{juv} = 1, \quad \forall j \in Q, u \in Q_j, \tag{7}$$

$$\sum_{v\in O} o_{uv} = 1, \quad \forall u \in O, \tag{8}$$

$$r_{uv} \geq q_{juv} - i_{uv}, \quad \forall j \in Q, u, v \in Q_j, \tag{9}$$

$$p_{uv} \geq o_{uv} - i_{uv}, \quad \forall u, v \in O, \tag{10}$$

$$i_{uv}, q_{juv}, o_{uv} \in \{0, 1\},$$

$$r_{uv}, p_{uv} \in \{0, 1\}.$$

The objective function (5) counts the number of input pairs of QLUTs and output pairs, for which a projection has to be inserted to provide them. Equations (6), (7), and (8) are perfect matching conditions for the primary inputs, the inputs of the QLUTs, and the outputs. Equations (9) and (10) define the auxiliary variables. Equation (9) guarantees, that for each

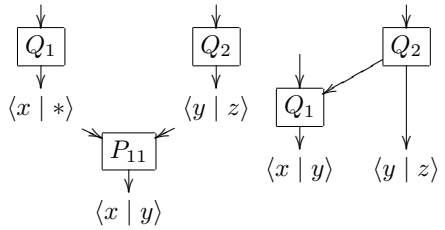


Figure 2. Heuristic used to reduce the number of projections. Left: Circuit before simplification. Right: Network after simplification.

QLUT $j$ and input pair $u, v$, a projection has to be inserted ($r_{uv} = 1$), if the pair actually is an input of the QLUT ($q_{juv} = 1$), but is not available as an input ($i_{uv} = 0$). These restrictions are only inserted, if $u, v$ is not available as an output of a QLUT as defined by the mapping in the first stage. Similarly, equation (10) defines the projections needed for outputs. The formulation has $O(n^2)$ variables and $O(n)$ restrictions, and assumes an even number of inputs and outputs; for circuits with an odd number, we add dummy variables to make a perfect matching possible.

The final step is the post-processing of the mapping obtained by the solution of the second ILP. This step determines the necessary projections to insert into the circuit. For each input of a QLUT or output, which is not available as a primary input or output of a previous QLUT, the algorithm inserts a projection to generate it.

Since the LUT mapping and the wiring is handled separately, there are situations where this final step generates more projections than necessary. We apply a heuristic rule to reduce the number of projections in some of these cases, as shown in Figure 2. If the net $\langle x \mid y \rangle$ should be provided, and $Q_1$ operates over only one relevant binary net, we are able to reduce eliminate the projection $P_{11}$, and can obtain the net $\langle x \mid y \rangle$ directly from $Q_1$.

*Complexity of the problem:* The first stage of the proposed mapping is a problem of obtaining a maximum matching between the BLUTs, which respects additionally the ordering constraints given by the circuit. Maximum matchings are well-known to be computable in polynomial time [12]. It can be shown that the additional ordering constraints make the problem NP-complete [13]. The entire mapping problem therefore is also NP-complete. Note that our approach is a heuristic for the entire mapping problem. Both stages can be solved exactly by integer linear programming, but determining the mapping of gates and nets in two separate stages can produce sub-optimal solutions.

## IV. Computational results

We used IBM ILOG CPLEX 12.1 [14] as a solver to map some benchmark circuits from the literature to quaternary logic. The experiments have been conducted on an Intel Core2 Quad Q6660 running at 2.4 GHz with 3 GB of RAM. We used all four processors of the machine in the deterministic parallel mode of CPLEX. The size and a description of the instances is given in Table I.

The characteristics of the quaternary circuits resulting from the mapping are given in Table II, which reports the number of necessary QLUTs, the number of projections inserted, and the number of (quaternary) inputs, nets, and outputs. The last but one column contains the number of required transistors to implement the circuit, based on a number of 288 transistors per QLUT and 48 transistors per projection [15]. Finally, the last column shows the wall-clock time in seconds required to map each of the circuits from BLUTs to QLUTs.

Table I
Binary circuits used in the computational experiments.

| Inst. | Type | BLUTs | Inp. | Net. | Out. | Tr. |
|---|---|---|---|---|---|---|
| fb4 | 4-bit full adder | 6 | 8 | 1 | 5 | 1584 |
| fb8 | 8-bit full adder | 12 | 16 | 3 | 9 | 3168 |
| fb16 | 16-bit full adder | 24 | 32 | 7 | 17 | 6336 |
| fb32 | 32-bit full adder | 48 | 64 | 15 | 33 | 12672 |
| mb4 | 4-bit multiplier | 28 | 8 | 20 | 8 | 7392 |
| mb8 | 8-bit multiplier | 65 | 16 | 79 | 16 | 25080 |
| c432 | Priority Decoder | 10 | 18 | 5 | 5 | 2640 |
| c499 | ECAT[1] | 62 | 41 | 30 | 32 | 16368 |
| c880 | ALU and Control | 73 | 56 | 49 | 24 | 19272 |
| c1355 | ECAT[1] | 77 | 41 | 118 | 32 | 20328 |
| c1908 | ECAT[1] | 75 | 33 | 108 | 25 | 19800 |
| c2670 | ALU and Control | 133 | 157 | 70 | 63 | 35112 |
| fir44 | 4-bit 4-tap FIR[2] | 36 | 4 | 0 | 36 | 9504 |
| fir84 | 8-bit 4-tap FIR[2] | 66 | 8 | 13 | 53 | 17424 |

[1]Error Correction and Translation. [2]Finite impulse response.

Table III compares the quaternary circuits to their binary counterparts in terms of QLUTs and wires. The column "LUTs" reports the ratio of binary versus quaternary LUTs, and the two remaining columns report the improvement in the number of transistors and wires compared with the binary implementations.

The results in Table III show that the mapping of binary to quaternary LUTs is often close to optimal. In the worst case (instance c499), the solver could map in average 1.29 BLUTs to a single QLUT. Good mapping results were to be expected, because the first stage of our approach minimizes the number of QLUTs. Since this stage does not consider the wiring, and is solved exactly, these values are exact lower bounds for the number of QLUTs for any mapping of these circuits. The same holds for the primary inputs and outputs: the second stage of the mapping process always used the minimum number of quaternary input and output wires.

Considering the inserted projections and nets the mapping generates good, but not optimal solutions. The number of

Table II
Characteristics of the resulting quaternary circuits.

| Inst. | QLUTs | Proj. | Inp. | Net. | Out. | Tr. | Time [s] |
|---|---|---|---|---|---|---|---|
| fb4 | 3 | 0 | 4 | 0 | 3 | 864 | 0 |
| fb8 | 6 | 2 | 8 | 3 | 5 | 1824 | 0 |
| fb16 | 12 | 6 | 16 | 9 | 9 | 3744 | 0 |
| fb32 | 24 | 15 | 32 | 22 | 17 | 7632 | 8 |
| mb4 | 15 | 17 | 4 | 28 | 4 | 5136 | 1 |
| mb8 | 49 | 55 | 8 | 96 | 8 | 16752 | 150 |
| c432 | 7 | 5 | 9 | 9 | 3 | 2256 | 0 |
| c499 | 48 | 68 | 21 | 100 | 16 | 17088 | 7 |
| c880 | 47 | 69 | 28 | 104 | 12 | 16848 | 25 |
| c1355 | 51 | 76 | 21 | 111 | 16 | 18336 | 60 |
| c1908 | 50 | 71 | 18 | 108 | 13 | 17808 | 28 |
| c2670 | 83 | 108 | 81 | 159 | 32 | 29088 | 744 |
| fir44 | 18 | 1 | 2 | 1 | 18 | 5232 | 0 |
| fir84 | 33 | 24 | 4 | 30 | 27 | 10656 | 37 |

Table III
COMPARISON OF BINARY AND QUATERNARY CIRCUITS.

| Inst. | LUTs | Trans. [%] | Wires [%] |
|---|---|---|---|
| fb4 | 2.00 | -45,45 | -50.00 |
| fb8 | 2.00 | -42.42 | -42.86 |
| fb16 | 2.00 | -40.91 | -39.29 |
| fb32 | 2.00 | -39.77 | -36.61 |
| mb4 | 1.87 | -30.52 | 0.00 |
| mb8 | 1.94 | -33.21 | 0.90 |
| c432 | 1.43 | -14.55 | -25.00 |
| c499 | 1.29 | 1.47 | 23.20 |
| c880 | 1.55 | -14.07 | 10.85 |
| c1355 | 1.51 | -10.74 | 22.03 |
| c1908 | 1.50 | -10.06 | 28.70 |
| c2670 | 1.60 | -17.16 | -6.21 |
| fir44 | 2.00 | -44.95 | -47.50 |
| fir84 | 2.00 | -38.84 | -17.57 |

projections inserted is in average about 86% of the number of QLUTs, and reaches up to 150% for the combinatorial circuits, i.e., each QLUT needs in average 1.5 projections to provide all its quaternary inputs. Since projections are much cheaper to implement than QLUTs, this still leads to an overall reduction of the number of transistors, with the exception of instance c499.

Looking at the wires, for instances mb4 and mb8 the mapping is not able to reduce the number of interconnections, and for four combinational circuits the number increases. This can be explained by the chosen two-stage approach, which minimizes transistors (QLUTs and projections) in both stages.

The last column of Table II shows the running time for both stages (in seconds) the ILP solver took to map the instances. For all instances, more than 90% of the time has been spent in the first stage. This is expected, since the first stage problem is NP-complete. The execution time also depends on the characteristics of the circuits, since the number of restrictions of the formulation of the first stage grows with the number of joinable BLUTs. Consequently, the proposed approach is useful for studying gains in circuits of modest size, but to be able to map larger circuits, we have to substitute the present strategy by more efficient heuristic methods.

## V. QUATERNARY DEVICES IN FUTURE TECHNOLOGIES

Process variability and reduced noise margin are important challenges for the development of new multiple-valued devices. In comparison with binary circuits, voltage-mode multiple logic devices present closer voltage levels to represent logic values. The main goal is to avoid excessive power consumption, and for this reason, in theory they are more susceptible to errors.

However, a recent work [15] demonstrated by extensive Monte Carlo simulations that quaternary devices deal very well with process variability and a reduced noise margin for technologies with nodes ranging from 90nm to 32nm.

In addition, analog designers used to deal with process variability for many years. Important works have been proposed in this field where authors shown the efficacy of these analog devices even in adverse conditions for nanometer technologies.

Quaternary devices can be seen in a similar way as analog devices. The knowledge and experience acquired by designer applied to the development of analog devices in nanotechnologies may be very useful in an effort to develop new quaternary devices.

## VI. CONCLUSIONS AND FUTURE WORK

Our results indicate that a quaternary implementation of arithmetic circuits can reduce transistor and interconnections costs significantly. The proposed approach yields very good results for the circuits studied in this paper, with almost optimal results for binary full adders up to 32 bits. However, the method is limited to circuits of moderate size, since it is based on the exact solution of an NP-complete problem. Also, it obtains best results for circuits of not too complex interconnection structure, which require the insertion of fewer projections and additional wires. It is also worth noting, that the mapping could be, with little effort, generalized to map LUTs of any number of input and outputs.

One important factor on the implementation of circuits in quaternary FPGAs is related to the obtained results in terms of the expected circuit performance and power consumption. The reduced number of transistors and wires will have direct impact on the circuit area and power consumption. Indirectly, reduced wire length associated to smaller interconnection capacitances will be the key for the development of faster circuits.

## REFERENCES


[1] W. J. Dally, "Computer architecture is all about interconnect," in *8th International Symposium on High-Performance Computer Architecture*, Cambridge, Massachusetts, Feb. 2002.

[2] R. Pragasan, "Spartan FPGA – the gate array solution," Xilinx Application Notes, Aug. 2001.

[3] A. Singh and M. Marek-Sadowska, "Efficient circuit clustering for area and power reduction in FPGAs," in *International Symposium on Field Programmable Gate Arrays*, Feb. 2002.

[4] P. M. Kelly, T. M. McGinnity, L. P. Maguire, and L. McDaid, "Exploiting binary functionality in quaternary look-up tables for increased functional density in multiple-valued logic FPGAs," *Electronics Letters*, vol. 41, no. 6, pp. 300–302, Mar. 2005.

[5] A. Gonzalez and P. Mazumder, "Multiple-valued signed digit adder using negative differential resistance devices," *IEEE Transactions on Computers*, vol. 47, no. 9, pp. 947–959, Sep 1998.

[6] T. Hanyu and M. Kameyama, "A 200 MHz pipelined multiplier using 1.5 v-supply multiple-valued mos current-mode circuits with dual-rail source-coupled logic," *IEEE Journal of Solid-State Circuits*, vol. 30, no. 11, pp. 1239–1245, Nov 1995.

[7] Z. Zilic and Z. Vranesic, "Multiple-valued logic in FPGAs," in *Proceedings of the 36th Midwest Symposium on Circuits and Systems*, Aug 1993, pp. 1553–1556 vol.2.

[8] R. C. G. da Silva, C. Lazzari, H. Boudinov, and L. Carro, "CMOS voltage-mode quaternary look-up tables for multi-valued FPGAs," *Microelectronics Journal*, vol. 40, no. 10, pp. 1466–1470, Oct. 2009.

[9] C. Lazzari, P. Flores, J. C. Monteiro, and L. Carro., "A new quaternary fpga based on a voltage-mode multi-valued circuit." in *Proceedings of the Design, Automation & Test in Europe 2010*, March 2010.

[10] *The programmable logic databook 2003*, Xilinx Inc., San Jose, California, 2003.

[11] *Ciclone III Device Data Sheet*, Altera Inc., San Jose, California, 2010.

[12] J. Edmonds, "Paths, trees, and flowers," *Canad. J. Math*, vol. 17, pp. 449–467, 1965.

[13] M. Ritt, "Maximum matching with ordering constraints NP-complete," Departamento de Informática Teórica, Instituto de Informática, Universidade Federal do Rio Grande do Sul, Tech. Rep. RP 362, 2009, Available: http://www.inf.ufrgs.br/ mrpritt/rp362.pdf.

[14] IBM, *IBM ILOG CPLEX V12.1 User's Manual*, 2009.

[15] E. L. Rhod and L. Carro, "A low cost low power quaternary LUT cell for fault tolerant applications in future technologies," in *IEEE Computer Society Annual Symposium on VLSI*, 2009.