ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2014

Design and Implementation of an On-Chip Permutation Network for Multiprocessor System-On-Chip

Manjunath E[1], Dhana Selvi D[2]

M.Tech Student [DE], Dept. of ECE, CMRIT, AECS Layout, Bangalore, Karnataka, India[1]

Professor, Dept. of ECE, CMRIT AECS Layout, Bangalore, Karnataka, India [2]

ABSTRACT: The novel on-chip network in silicon proven design to support guaranteed traffic permutation in multiprocessor SOC applications. A pipelined circuit-switching Employed in the proposed network with FIFO approach combined with a multistage network topology in dynamic path-setup scheme. The runtime path arrangement enabled by dynamic path-setup scheme for arbitrary traffic permutations along with the Error Correction Block (ECB). The circuit-switching approach offers the permuted data and its compact overhead enables the benefit of stacking multiple networks in system on chip. A CMOS test-chip with 0.13m validates the feasibility and efficiency of the proposed design. The shown experimental result in the proposed on-chip network achieves 1.9x to 8.2x reduction of silicon overhead compared to other design approaches.

KEYWORDS: Traffic permutation Guaranteed throughput, network-on-chip, permutation network, pipelined circuit-switching, multistage interconnection network.

I.INTRODUCTION

The multiprocessor system-on-chip (MP SoC) design being interconnected with on-chip networks is currently emerging for applications of parallel processing, scientific computing, and so on Permutation traffic, On-chip multiprocessing applications exhibited with a traffic pattern in which each input sends traffic to exactly one output and each output receives traffic from exactly one input, is one of the important traffic classes. The standard traffic occurs in general-purpose MP SoCs , for example, and fast Fourier transform (FFT) computations, polynomial, sorting cause shuffled permutation, whereas transpose permutation is exhibited in matrix transposes or corner-turn operations. Recently, application specific MP SoCs targeting flexible Turbo/LDPC decoding have been developed, and they exhibit arbitrary and concurrent traffic permutations due to multi-mode and multi standard feature. In addition, many of the MP SoC applications (e.g., Turbo/ LDPC decoding) compute in real-time, therefore, guaranteeing throughput (i.e., data lossless, guaranteed bandwidth, predictable latency, and in-order delivery) is critical for such permutation traffics

Regarding the topology, regular direct topologies, such as mesh and torus, are intuitively feasible in a 2-D chip with physical layout. On the contrary, the large router radix and the high wiring irregularity of indirect topologies such as Butterfly or Benes pose a challenge for physical implementation. However, throughput degradation is lead in an arbitrary permutation pattern with its intensive load on individual source destination pairs stresses the regular topologies. Indirect multistage topologies are preferred for on-chip traffic permutation intensive applications.

II. REVIEW OF LITERATURE

A dynamic PA Common switch architecture-setup scheme supports a runtime path arrangement when the permutation is changed. Each path setup, Based on dynamic probing mechanism which starts from an input to find a path leading to its corresponding output. The probing is introduced in works, in which a probe (or setup flit) is dynamically sent under a routing algorithm in order to establish a path towards the destination. The Exhausted profitable backtracking (EPB) use to route the probe in the network work. A path arrangement with full permutation consists of sixteen path setups, whereas a path arrangement with partial permutation may consist of a subset of 16 path setups.

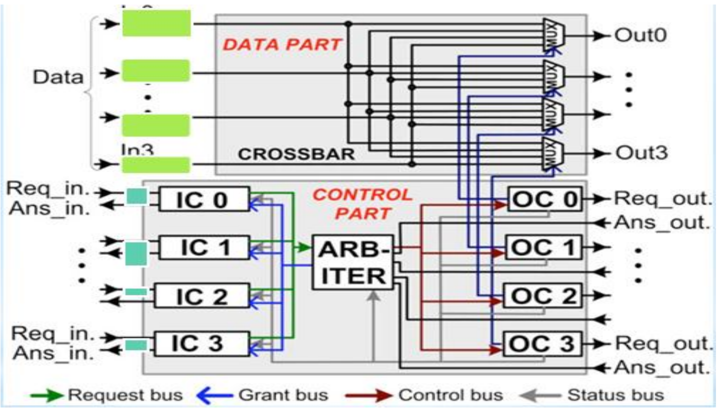


Fig.1 Common switch architecture with ECC and Input FIFO.

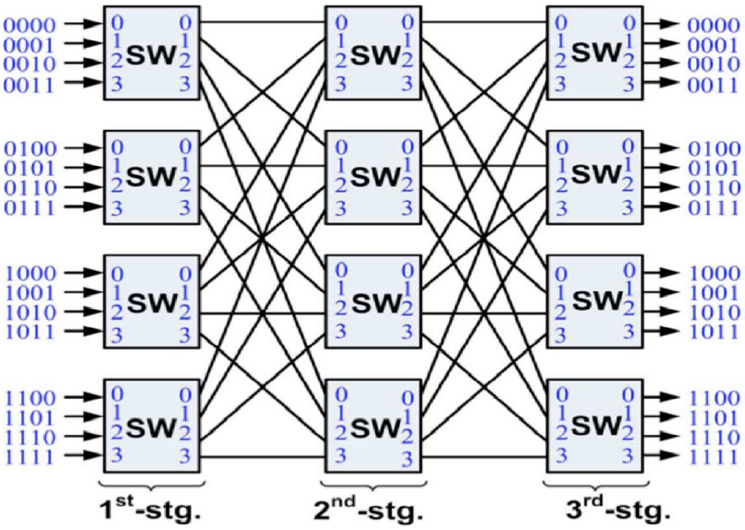


Fig. 2. Proposed on-chip network topology with port addressing scheme.

Regarding the switching technique, packet switching requires an excessive amount of on-chip power and area for the queuing buffers (FIFOs) with pre-computed queuing depth at the switching nodes and/or network interfaces. Regarding the routing algorithm, the deflection routing is not energy-efficient due to the extra hops needed for deflected data transfer, compared to a minimal routing. Moreover, the deflection makes packet latency less predictable; hence, it is hard to guarantee the latency and the in-order delivery of data. Unlike conventional packet-switching approaches, our on-chip network employs a circuit-switching mechanism with a dynamic path-setup scheme under a multistage network topology. The dynamic path setup tackles the challenge of runtime path arrangement for conflict-free permuted data. The pre-configured data paths enable a throughput guarantee. By removing the excessive overhead of queuing buffers, a compact implementation is achieved and stacking multiple networks to support concurrent permutations in runtimes feasible.

III. ON-CHIP NETWORK TOPOLOGY

Clos network, a family of multistage networks, is applied to build scalable commercial multi core processors with thousands of nodes in macro systems. A typical three-stage Clos network is defined as C (n, m, p), where n represents the number of inputs in each of p first stage switches and m is the number of second stage switches. In order to support a parallelism degree of 16 as in most practical MP SoCs, we proposed to use C (4, 4, 4) as a topology for the designed network. This network has a rearrange able property that can realize all possible permutations between its input and outputs. The choice of the three stage Clos network with a modest number of middle stage switches is to minimize implementation cost, whereas it still enables a rearrange able property for the networks. A pipelined circuit switching scheme is designed for use with the proposed network. This scheme has three phases: the setup, the transfer, and the release. A dynamic path setup scheme supporting the runtime path arrangement occurs in the setup phase. In order to support this circuit switching scheme, a switch by switch interconnection with its handshake signals is proposed. The bit format of the handshake includes a 1 bit Request (Req) and a 2 bit Answer (Ans). Req = 1 is used when a switch requests an idle link leading to the corresponding downwstream switch in the setup phase. The Req = 1 is also kept during data transfer along the set up path. A Req = 0 denotes that the switch releases the occupied link. This code is used in both the setup and the release phases. An Ans = 01(Ack) means that the destination is ready to receive data from the source. When the Ans = 01 propagates back to the source, it denotes that the path is set up, then a data transfer can be started immediately. An Ans = 11(nAck) is reserved for end to end flow control when the receiving circuit is not ready to receive data due to being busy with other tasks, or overflow at the receiving buffer, etc. An Ans = 10 (Back) means that the link is blocked. This Back code is used for a back pressure flow control of the dynamic path setup scheme.

IV. DYNAMIC PATH SETUP TO SUPPORT PATH ARRANGEMENT

A dynamic path setup scheme of the proposed design to support a runtime path arrangement when the permutation is changed. In each path setup, which starts from an input to find a path leading to its corresponding output, is based on a dynamic probing mechanism. The concept of probing is introduced, in which a probe (or setup flit) is dynamically sent under a routing algorithm in order to establish a path towards the destination. The exhausted profitable backtracking (EPB) is proposed to use to route the probe in the network work. A path arrangement with full permutation consists of 16 path setups, whereas a path arrangement with partial permutation may consist of a subset of 16 path setups. A question is that can the proposed EPB based path setups used with the Clos C (4, 4, 4) realize all possible full permutations between its inputs and outputs? As proofed in works, the three stage Clos network C(n, m, p) is rearrange able if m>n In the proposed network of C(4, 4, 4) m = n = 4 so it is rearrange able. There will always exist an available path from an idle input leading to an idle output. By the Exhaustive Property of EPB as proofed in work, the EPB based path setup completely searches all the possible paths within the set of path diversity between an idle input and idle output. Directly applying the Exhaustive Property of the search into rearrange able C (4, 4, 4) shows that the EPB based path setup can always find an available path within the set of four possible paths between the input and the idle output. Based on this EPB based path-setup scheme, it is obvious that the path arrangement for full permutation can always be realized in the proposed network with C (4, 4, 4) topology.

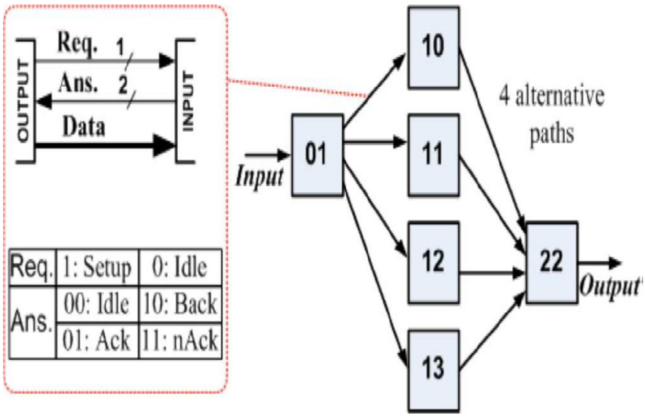


Fig. 3. Switch by switch interconnection and path diversity capacity.

As designed in this network, each input sends a probe containing a 4 bit output address to find an available path leading to the target output. During the search, when the probe moves forwards it will find a free link and moves backwards it will face a blocked link. In non-repetitive movement, the probe finds an available path between the idle input and its corresponding idle output. The EBP based path setup scheme is designed with a set of probe routing algorithms as mentioned later. The following examples describe how the path setup works to find an available path by using the set of path diversity. It is assumed that a probe from a source (e.g., an input of switch 01) is trying to set up a path to a target destination (e.g., an available output of switch 22). First, the probe will non-repetitively try paths through the second stage switches in the order of 10»11»12»13 Assuming that the link 01-10 is available, the probe first tries this link Req = 1 and then arrives at switch 10. If link 10-22 is available, the probe arrives at switch 22 and meets the target output. If Ans = Ack then propagates back to the input to trigger the transfer phase. If link 10-22 is blocked, the probe will move back to switch 01 (Ans = back) and link 01-10 is released Req = 0. From switch 01, the probe can then try the rest of idle links leading to the second stage switches in the same manner. By means of moving back when facing blocked links and trying others, the probe can dynamically set up the path in runtime in a conflict avoidance manner.

V. SIMULATION RESULTS AND DISCUSSION

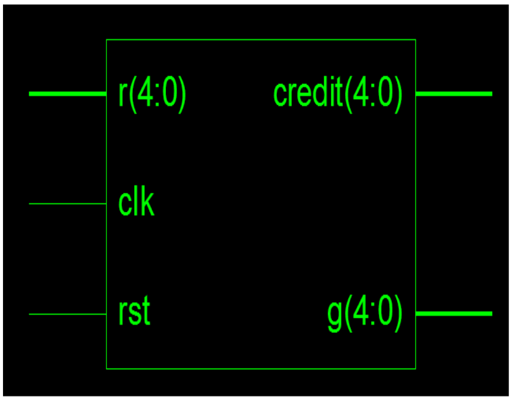


Fig.4 RTL of Arbiter

The RTL schematic of the simulated of arbiter is as shown above in fig 4 of R(4:0) shows the input for arbiter which consists of controls signals from input circuit. Were credit and G(4:0) indicates the credit signals and grant signals

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2014

respectively along with these control signals there are some deciding input signals for the arbiter which are reset and clock.

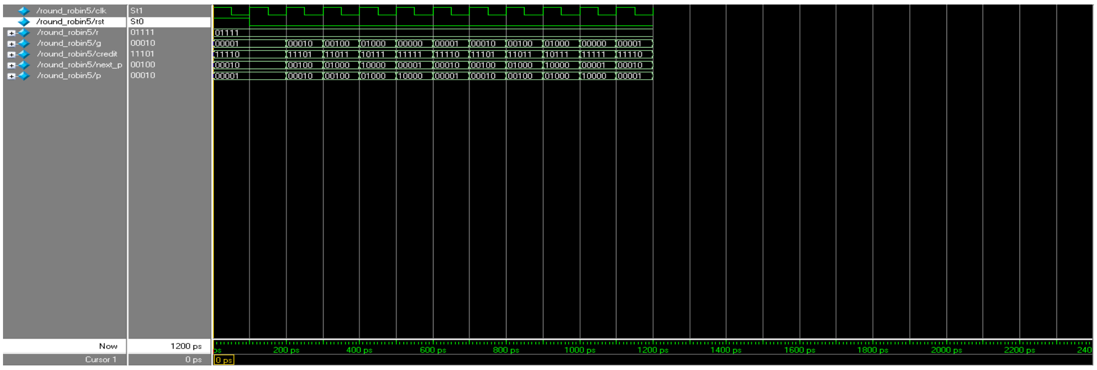
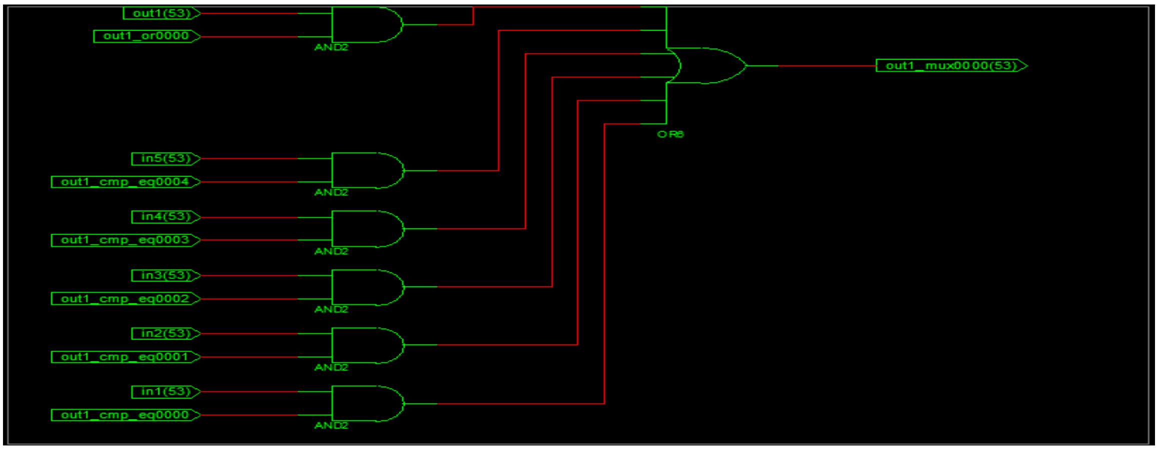


Fig.5 Simulation Results of Arbiter

The simulation waveform of the arbiter is as shown above in fig 5 of R(4:0) shows the input for arbiter which consists of controls signals from input circuit. Were credit and G(4:0) indicates the credit signals and grant signals respectively along with these control signals there are some deciding input signals for the arbiter which are reset and clock with the next-p signal for the next registers.



.

Fig.6 RTL Schematic of Cross Bar

The RTL schematic of the internal circuit of cross bar is as shown above in fig 6 which consists of and gates were the input of these and gates are given from manual data or from any module in turn the output is decided by one the multiplexer. The circuit shown is for the one input cross bar.

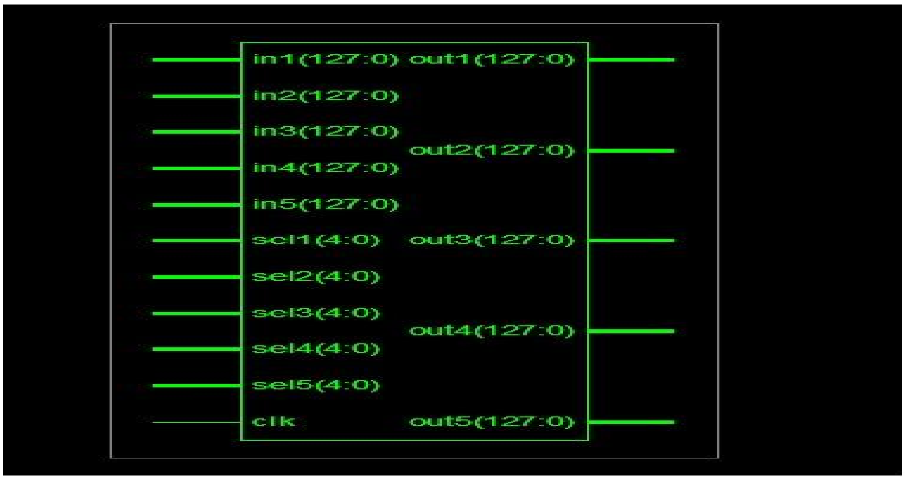


Fig 7 Block Diagram of Cross Bar

The RTL schematic of the simulated of cross bar is as shown above in fig 7. The complete cross bar is shown in the schematic were the 4x4 input and output lines and 4 select lines.

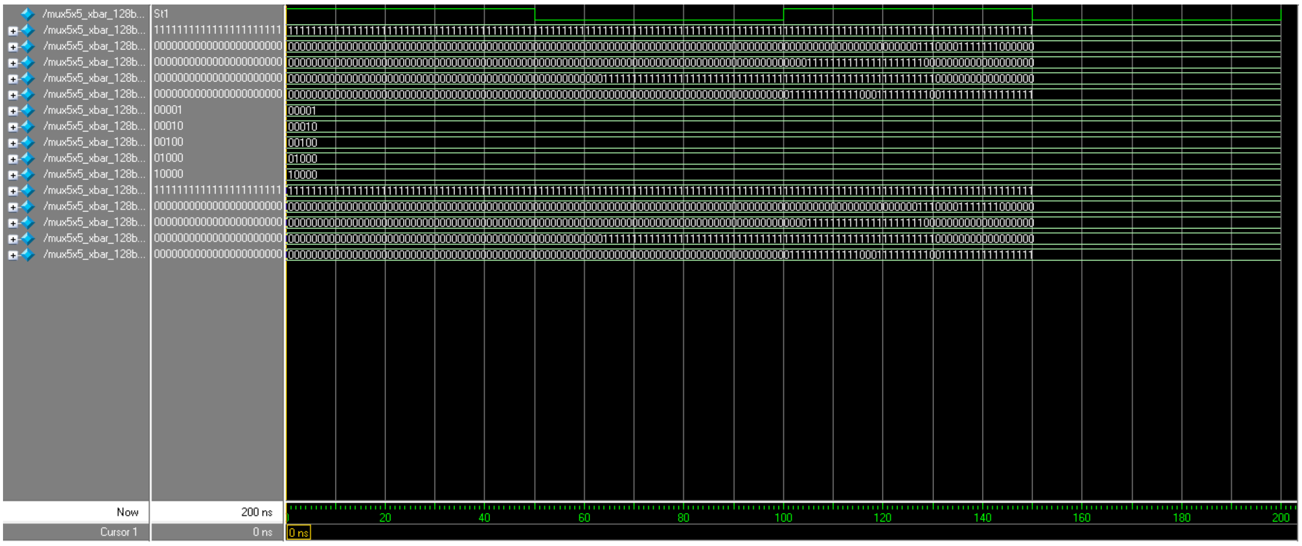

/mux5x5_xbar_128b... St1
/mux5x5_xbar_128b... 11111111111111111111
/mux5x5_xbar_128b... 0000000000000000000000
/mux5x5_xbar_128b... 0000000000000000000000
/mux5x5_xbar_128b... 0000000000000000000000
/mux5x5_xbar_128b... 0000000000000000000000
/mux5x5_xbar_128b... 00001
/mux5x5_xbar_128b... 00010
/mux5x5_xbar_128b... 00100
/mux5x5_xbar_128b... 01000
/mux5x5_xbar_128b... 10000
/mux5x5_xbar_128b... 11111111111111111111
/mux5x5_xbar_128b... 0000000000000000000000
/mux5x5_xbar_128b... 0000000000000000000000
/mux5x5_xbar_128b... 0000000000000000000000
/mux5x5_xbar_128b... 0000000000000000000000
Now 200 ns
Cursor 1 0 ns
0 ns
20
40
60
80
100
120
140
160
180
200

Fig.8 Simulation results of Cross Bar

The simulation waveform of the cross bar is as shown above in fig 8. Where the 4x4 input and output lines and 4 select lines.

VI. CONCLUSION

On chip network design supporting traffic permutations in MPSoC applications. By using a circuit switching approach combined with dynamic path setup scheme under a Clos network topology, the proposed design offers arbitrary traffic permutation in runtime with compact implementation overhead. A silicon proven test chip validates the proposed design and suggests availability for use as an on-chip infrastructure-IP supporting traffic permutation in future MPSoC researches

REFERENCES

[1] S. Borkar, “Thousand core chips—A technology perspective,” in Proc. ACM/IEEE Design Autom. Conf. (DAC), 2007, pp. 746–749.

[2] P.-H. Pham, P. Mau, and C. Kim, “A 64-PE folded-torus intra-chip communication fabric for guaranteed throughput in network-on-chip based applications,” in Proc. IEEE Custom Integr. Circuits Conf. (CICC), 2009, pp. 645–648.

[3] C. Neeb, M. J. Thul, and N.Wehn, “Network-on-chip-centric approach to interleaving in high throughput channel decoders,” in Proc. IEEE Int. Symp. Circuits Syst. (ISCAS), 2005, pp. 1766–1769.

[4] H. Moussa, A. Baghdadi, and M. Jezequel, “Binary de Bruijn on-chip network for a flexible multiprocessor LDPC decoder,” in Proc. ACM/ IEEE Design Autom. Conf. (DAC), 2008, pp. 429–434.

[5] H. Moussa, O. Muller, A. Baghdadi, and M. Jezequel, “Butterfly and Benes-based on-chip communication networks for multiprocessor turbo decoding,” in Proc. Design, Autom. Test in Euro. (DATE), 2007, pp. 654–659.

[6] S. R. Vangal, J. Howard, G. Ruhl, S. Dighe, H. Wilson, J. Tschanz, D. Finan, A. Singh, T. Jacob, S. Jain, V. Erraguntla, C. Roberts, Y. Hoskote, N. Borkar, and S. Borkar, “An 80-tile sub-100-w TeraFLOPS processor in 65-nm CMOS,” IEEE J. Solid-State Circuits, vol. 43, no. 1, pp. 29–41, Jan. 2008.

[7] W. J. Dally and B. Towles, Principles and Practices of Interconnection Networks:. San Francisco, CA: Morgan Kaufmann, 2004.

[8] N. Michael, M. Nikolov, A. Tang, G. E. Suh, and C. Batten, “Analysis of application-aware on-chip routing under traffic uncertainty,” in Proc. IEEE/ACM Int. Symp. Netw. Chip (NoCS), 2011, pp. 9–16.

[9] P.-H. Pham, J. Park, P. Mau, and C. Kim, “Design and implementation of backtrackingwave-pipeline switch to support guaranteed throughput in network-on-chip,” IEEE Trans. Very Large Scale Integr. (VLSI) Syst., 10.1109/TVLSI.2010.2096520.

[10] D. Ludovici, F. Gilabert, S. Medardoni, C. Gomez, M. E. Gomez, P. Lopez, G. N. Gaydadjiev, and D. Bertozzi, “Assessing fat-tree topologies for regular network-on-chip design under nanoscale technology constraints,” in Proc. Design, Autom. Test Euro. Conf. Exhib. (DATE), 2009, pp. 562–565.