A Temperature-Aware Placement and Routing Algorithm Targeting 3D FPGAs

Kostas Siozios, Dimitrios Soudris

A Temperature-Aware Placement and Routing Algorithm Targeting 3D FPGAs

Kostas Siozios and Dimitrios Soudris

National Technical University of Athens (NTUA),
School of Electrical & Computer Engineering,
9 Heroon Polytechneiou, Zographou Campus, 157 80 Athens, Greece
{ksiop, dsoudris}@microlab.ntua.gr

Abstract. In current reconfigurable architectures, the interconnect structures increasingly contribute to the delay and power consumption budget. The demand for increased clock frequencies and logic availability (smaller area foot print) makes the problem even more important, leading among others to rapid elevation in power density. Three-dimensional (3D) architectures are able to alleviate this problem by accommodating a number of functional layers, each of which might be fabricated in different technology. Since power consumption is a critical challenge for implementing applications onto reconfigurable hardware, a novel temperature-aware placement and routing (P&R) algorithm targeting 3D FPGAs, is introduced. The proposed algorithm achieves to redistribute the switched capacitance over identical hardware resources in a rather "balanced" profile, reducing among others the number of *hotspot* regions, the maximal values of power sources at *hotspots*, as well as the percentage of device area that consumes high power. For evaluation purposes, the proposed approach is realized as a new CAD tool, named 3DPRO (3D-Placement-and-Routing-Optimization), which is part of the complete framework, named 3D MEANDER. Comparing to alternative solutions, the proposed one reduces the percentage of silicon area that operates under high power by 63%, while it leads to energy savings (about 9%), with an almost negligible penalty in application's delay ranging from 1% up to 5%.

1. Introduction

For decades, semiconductor manufacturers have been shrinking transistor size in ICs to achieve the yearly increases in speed and performance described by Moore's Law, which exists only because the RC delay was negligible in comparison with signal propagation delay [1]. For submicron technology, however, the RC delay becomes a dominant factor. This has generated many discussions concerning the end of device scaling as we know it, and has hastened the search for solutions beyond the perceived limits of current 2D devices.

One emerging solution to this problem is the 3D integration, which replaces a large number of long interconnects needed in 2D structures with shorter ones. Such

architectures mitigate many of the limitations that the 2D devices exhibit. Among others, they provide: (*i*) higher logic density in the same foot print area, (*ii*) shorter interconnections among the logic blocks, (*iii*) reduced signal propagation delay, (*iv*) greater versatility and resource utilization, and (*v*) lower power consumption.

One of the most critical challenges for efficient application implementation in 3D FPGAs is the power management, and hence the thermal problem, which has already been studied for 2D architectures [6, 7, 17]. This problem is exacerbated in the 3D devices for two reasons: (*i*) the vertically stacked layers cause a rapid increase of power density [9], and (*ii*) the thermal conductivity of the dielectric layers inserted between device layers for insulation is very low compared to silicon and metal.

Moreover, an obvious consequence of this trend is the increased power consumption per area unit. In recent years, power density in 2D FPGAs has doubled every three years [1], and this rate is expected to increase as feature sizes, frequencies and technologies scale faster than operating voltages. As the power density will continue increasing in future technologies (according to "*A-power*" law), the power consumption is regarded as a limiting factor to the increasing scales of integration predicted by Moore's law [1].

Thermal management of Field-Programmable Gate Array (FPGA) devices is more critical compared to ASIC solutions, as they dissipate more power, while their operating temperatures usually exceed the critical one. Also, the leakage current increases exponentially with temperature, causing a positive feedback loop between leakage power and temperature.

Eliminating and managing power consumption for reconfigurable architecture requires appropriate algorithm support. Realizing applications on 2D FPGAs is a well studied problem; however, there are only a few solutions regarding 3D architectures [8, 13, 14, 16].

In [13] a P&R approach for 3D ICs is presented, having as criterion to minimize the total wire-length, the applications delay, and the on-chip temperature. Even though the framework supports reconfigurable architectures, however, the thermal feature is available sorely for ASIC designs.

A similar approach is shown in [14], where the P&R algorithm optimizes the energy consumption and the thermal profile of a 3D standard-cell device under the supplied timing constraint. The employed algorithm focuses on the energy consumption of interconnect-related components. Unfortunately, the software implementation is not publically available, in order to evaluate this approach against to our proposed solution.

In [16] a thermal-driven 3D floor-planning algorithm that provides a trade-off between runtime and quality is presented. The algorithm tries to reduce the total wire-length, as well as the maximum on-chip temperature, compared to a non thermal-driven approach.

In [8] a P&R algorithm and its software implementation targeting to explore alternative interconnection schemes for 3D FPGAs are introduced. The employed cost functions pay effort to minimize the application delay, the power/energy consumption, as well as the total wire length, ignoring about their distribution. This tool is part from an open-source CAD framework, named *3D MEANDER*, for mapping applications onto 3D FPGAs.

All these approaches realize digital applications on 3D devices having as goal to minimize the total power/energy consumption of the design, ignoring about the spatial distribution of its sources. Moreover, none of them is aware during the P&R procedure about spatial distribution of parameters that affect the power/energy consumption (*i.e.*, switched capacitance). This results both to increased power/energy consumption, as well as to significant variations of on-chip temperature values across the 3D device. Among others, this non-uniformity in power consumption leads to increased cooling costs, as the IC packaging has to be designed for the worst case scenario.

The rest paper is organized as follows. In Section 2, we formulate the temperature-aware P&R problem, while the employed methodology in order to derive the temperature model is described in Section 3. The algorithmic steps of the proposed temperature-aware P&R algorithms are introduced in Section 4. Section 5 evaluates the efficiency of applying such a temperature-aware approach against to other implementations for application mapping, while conclusions are summarized in Section 6.

2. Problem Formulation

Power consumption of FPGAs is generally grouped into three categories: (*i*) dynamic power, (*ii*) static power, and (*iii*) interface (I/O) power. These components are governed by the process technology and traditionally maintain constant percentages of the device's total power. The dynamic part of power consumption (formulated in the Equation (1)), occurring due to signal transition as the load capacitance is charged (or discharged), still dominates the total power consumption. In this equation, f represents the clock frequency of the signal, V_{dd} is the supply voltage, while Cap_i and $Activity_i$ are the capacitance and switching activity, respectively, of element i.

$$P_{switching} = 0.5 \cdot f \cdot V_{dd}^2 \cdot \sum_{i=1}^{Nets} \{Cap_i \cdot Activity_i\} \qquad (1)$$

When a lower bound on the supply voltage is set by external constraints (as often happens in real-world designs), or when the performance degradation due to lowering of the supply voltage is intolerable, then the only means of reducing power consumption is by lowering the effective capacitance and/or the switching activity (*i.e.*, switched capacitance). Throughout this paper, we discuss an algorithm for managing the spatial distribution of this product ($Cap_i \cdot Activity_i$) over the 3-D FPGA device, leading to a more "*uniform*" temperature profile.

Definition: Application Hypergraph

We consider as application hypergraph a directed hypergraph $AppG(L, N)$, where each vertex $l_i \in L$ represents a logic functionality of the target application, while the directed hyperedge $n_{i,j} \in N$ encodes the communication between logic functionalities l_i and l_j. The weight associated to hyperedge $n_{i,j}$, denoted as

$communication\_weight_{i,j}$, represents the communication load/bandwidth from vertex l_i to l_j.

Definition: Platform Graph
We consider as platform graph a directed graph $PlatG(C,W)$ where each vertex $c_i \in C$ represents an element of the target architecture (*e.g.*, logic block, processor, memory, etc.), while the directed edge $w_{i,j} \in W$ denotes a communication path between hardware elements c_i and c_j. The weight of the edge $w_{i,j}$, denoted as $interconnection\_weight_{i,j}$, encodes the fabricated interconnection hardware resources among these logic blocks.

3D Temperature-Aware Placement and Routing Problem
Given the architecture graph $PlatG(C,W)$ consisted by a set of V ($V = i \times j \times k$) slices (S), where $S = \{S_1(x_1, y_1, z_1), \cdots, S_v(x_i, y_j, z_k)\}$ find a placement ($Place: L \rightarrow C$) and a routing ($Route: N \rightarrow W$) of the application hypergraph on the available hardware resources (platform graph), in order each logic function (L) and the appropriate communication (N) to occupy uniquely a logic resource (C) and the available routing fabric (W), respectively. The derived P&R solution is accepted if the following conditions are satisfied:

(i) $S_i \cap S_j = 0$ for all the $i, j \in PlatG$.
(ii) The interconnection of each layer is accomplished with the minimum routing resources.
(iii) Distribute uniformly the switched capacitance over the 3D device.
(iv) Meeting timing/power/area constraints of the application.
(v) Employ an acceptable number of vertical links (in terms of the selected 3D bonding technology).

The proposed temperature-aware P&R solution was evaluated against to existing (*i.e.*, non temperature-aware) P&R algorithms with the usage of the 20 biggest MCNC benchmarks [15]. During this experimental setup, the P&R algorithms were applied to identical (*i.e.*, with same amount of logic resources and interconnection fabric) 3D FPGAs. The results show significant reduction (about 63%) on area percentage that operates under high temperatures, while we also achieve energy savings about 9%.

3. Methodology for deriving the temperature model

The proposed methodology for deriving the employed temperature model that calibrates our P&R algorithms, is depicted in Figure 1. For this reason, a representative number of benchmarks from [15] were implemented (with the same P&R algorithms) onto 3D FPGAs. Then we visualize the variation of switched capacitance over each layer, in order to determine the number, as well as the spatial distribution of *hotspot* regions. As a *hotspot* we refer to the device region where the temperature is higher than 70% of the maximum temperature of the 3D FPGA. This

step was presented in a previous work regarding the 2D architectures [7]. Then, the application is P&R on the target 3D architecture with the derived temperature-aware algorithm. This step is described in more detail in upcoming sections.

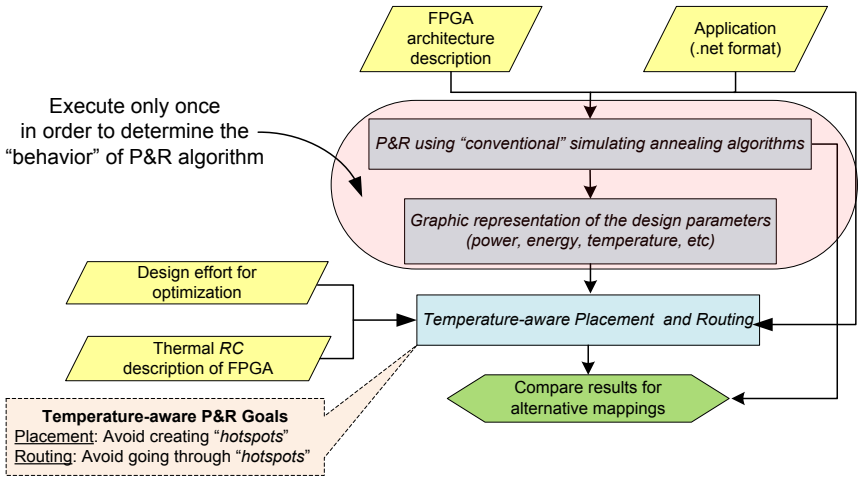

Figure 1: Proposed methodology for temperature-aware P&R

Our interest lies to control the temperature sources across the 3D device. In order to model this, each of the functional layers is divided into a grid with dimensions $X \times Y$, where every point (x, y, z) is assumed to be small enough in order its temperature to be constant. In general, the steady-state temperature of each location across the FPGA is a function of the total power consumption regarding all the on-chip heat sources. Equation (2) gives the parameters of the thermal RC circuit [7] that models the temperature across the device.

$$T_{xyz} = \sum_{x=1}^{X} \left\{ \sum_{y=1}^{Y} \left\{ \sum_{z=1}^{Z} \{ R_{xyz} \times P_{xyz} \} \right\} \right\} \quad (2)$$

In this equation, the value of R_{xyz} refers to the transfer thermal resistance, while the P_{xyz} represents the power consumption of the slice placed at spatial location (x, y, z). The on-chip temperature for this point is represented as T_{xyz}.

4. Proposed P&R Algorithm Targeting 3D FPGA devices

Fundamentally, the problem of 3D P&R is related to topological arrangements of the application's functionality to slices (*i.e.*, logic blocks) of the 3D FPGA, while satisfying the design timing, power and area constraints. The proposed temperature-

aware P&R algorithm pays effort to minimize the on-chip temperature gradient, obtaining an even uniformly spatial distribution of switched capacitance, in respect to the application's timing constraints. This approach can be though as a power management strategy. The result is an application mapping with fewer *hotspot* regions, as compared to a conventional (*i.e.*, timing-aware) approach.

Figure 2 shows the tool flow, named *3D MEANDER*, for realizing applications on 3D FPGAs. This flow adopts some existing CAD tools from the 2D toolset [5], which do not need to be aware of the 3D FPGA topology (*i.e.*, technology platform independent). To the best of our knowledge, this toolset is the first complete framework in academia for mapping applications on 3D reconfigurable devices starting from hardware description language up to configuration file generation.

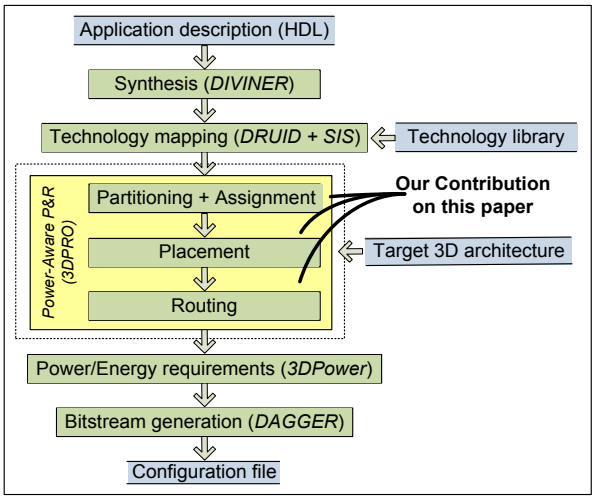

Figure 2: The 3D MEANDER Framework

The proposed temperature-aware algorithm (shown in Algorithm 1) is implemented within the *3DPRO* tool [8]. The calculation of application's delay is based on Elmore model [12]. Regarding the algorithmic complexity of this tool, it is similar to existing solutions [10, 13]. Regarding the calculation of temperature variations across the 3D FPGA, we employ models introduced in [2], appropriately extended in order to be aware about the third dimension. These models were integrated in the *3DPower* tool.

```
Function TEMPERATURE-AWARE P&R FOR 3D FPGAs()
     Input netlist: technology mapped application
     Input target architecture: target 3D FPGA device
     Partition(netlist, target architecture);
     Place(netlist, target architecture);
     Route(netlist, target architecture);
     Calculate statistics(netlist);
End Function
```

Algorithm 1: The proposed temperature-aware P&R

Since our proposed P&R approach is temperature-aware, it poses new challenges to application implementation on 3D devices. Detail description of each algorithmic step (*i.e.*, partitioning, placement, routing) will be given in the upcoming sections.

4.1 Application Partitioning and Layer Assignment

The first step of the proposed temperature-aware P&R algorithm deals with the application partitioning into Z balanced sections. This number (Z) is equal to the 3D device layers. The employed partitioning algorithm (shown in Algorithm 2) is based on [11], as it tries to minimize the interlayer communication, however its cost function was appropriately extended to spread as much as possible the spatial distribution of application's switched capacitance across the 3D device, without affecting the total power/energy consumption, the application's delay or the area requirements.

This procedure is done by recursive bi-partitioning of the application hypergraph $AppG$, such that to minimize the value of the employed cost function (depicted in Equation (3)). Since the net length is tightly firmed to its resistance and capacitance values, we can manage the power consumption sources by weighting each net according to its switched capacitance.

```
Function PARTITION(netlist, number of layers)
     while accept (partition=True) do
          Subgraphs ← split(netlist, number of layers);
          C ← calculate(connections among subgraphs);
          If (C > crititcal) then
               try to repartition the netlist;
          else
               accept partitions ← True
          end if
     end while
end Function
```

Algorithm 2: The proposed temperature-aware partitioning

We associate the switched capacitance criticality of a logic element as weight to the corresponding vertex in the hypergraph, while the timing criticality is shown as weight to the corresponding hyperedge. These weights encourage the partitioning algorithm to split the application in a way that balances both of these factors. The criticalities of the graph (*i.e.*, weights of vertexes and hyperedges) are updated at each partitioning level, while the partitioning process stops when both the switched capacitance distribution and the timing constraints are met.

Next, the algorithm assigns the derived application segments on the device layers by taking into consideration a number of design parameters. More specifically, the algorithm tries not to assign segments that consume high power close to each other, or on the middle of the 3D stack, as it is more difficult to dissipate heat. This task is

accomplished in conjunction to the effort for minimizing the interlayer communication or other design constraints (*i.e.*, delay, power/energy consumption, etc).

Additionally, since our algorithm can be used for architecture-level exploration, rather than providing only an output, we calculate the Pareto-based space of alternative application partitions. These solutions balance the area occupied by active hardware resources, the number of interlayer connections and the variation of power consumption (*i.e.*, power sources) among layers. In order to quantify each of the derived application partitioning, we employ cost function shown in Equation (3).

$$cost_{function} = (k \times power_{variation}) + (1-k) \times \{(p \times interlayer_{connections}) + ((1-p) \times area_{balance})\} \quad (3)$$

where $power_{variation}$ denotes the variation of power sources over the 3D FPGA, the $interlayer_{connections}$ is equal to the total amount of hyperedge-cut, while the $area_{balance}$ corresponds to the variation of area occupied by active hardware resources among the device layers. The employed factors k and p provide higher flexibility to the cost function, as they can be used to tune the algorithm for further optimizing the partitioning result. Finally, we have to mention that both the cost function, as well as the criticalities of the hypergraph (*i.e.*, weights of vertexes and edges), are updated after each iteration, while the partitioning stops when both the distribution of switched capacitance and the timing constraints are met.

4.2 Application Placement

After the partitioning step, the placement algorithm assigns the application's logic functionalities (L) to available hardware modules (C). As the majority of applications realized onto FPGAs utilize only a subset of the available hardware resources, this non-uniformity leads to high variation of power consumption across the device [8]. This problem gets even worst in 3D devices due to high power/temperature variation among layers.

The proposed temperature-aware placement algorithm tries to place the logic functionalities (L) in a way that minimizes the maximal switched capacitance values (referred as *hotspots*), as well as to distribute it across the whole 3D FPGA. As the switching activity depends on the functionality implemented inside the logic blocks, while the capacitance is proportional to the interconnection length and the number of hardware modules that form each network, the proposed algorithm pays effort to handle in an efficient way their product (*i.e.*, switched capacitance).

More specifically, by placing on adjacent spatial locations logic functionalities connected through nets with high switching activity, these nets probably will be shorter (exhibit smaller capacitance), leading to reduced power consumption. Unfortunately, it is not always possible to place close all these blocks, as this might lead to increased application delay (*i.e.*, delay of the slowest path). Also, the placement of functionalities with high bandwidth requirements should be assigned

onto the same functional layer, since there is plethora of routing resources, as compared to the reduced connectivity of vertical connectivity.

The proposed placement approach (shown in Algorithm 3) is based on simulated annealing. During the placement pairs of logic blocks are selected and swapped randomly, until either the resulted placement is good enough, or the maximum number of iterations is reached. The efficiency of a placement is characterized by calculating its cost function, shown in Equation (4), where:

$$\Delta Cost = \alpha \times \frac{\Delta Temperature_{cost}}{Previous\ Temperature_{cost}} + (1-\alpha) \times \left[\beta \times \frac{\Delta Wire_{cost}}{Previous\ Wire_{cost}} + (1-\beta) \times \frac{\Delta Time_{cost}}{Previous\ Time_{cost}}\right] \quad (4)$$

where

$$Temperature_{cost} = T\left[\sum_{i=1}^{Total\ l_{Nets}} \left\{Activity(i) \times \left[q(i) \times \left[\frac{bb_x(i)}{C^{\varepsilon}_{av,x}(i)} + \frac{bb_y(i)}{C^{\varepsilon}_{av,y}(i)} + r \times \frac{bb_z(i)}{C^{\lambda}_{av,z}}\right]\right]\right\}\right]$$

$$Wire_{cost} = \sum_{i=1}^{Total\_Nets} \left\{q(i) \times \left[\frac{bb_x(i)}{C^{\varepsilon}_{av,x}(i)} + \frac{bb_y(i)}{C^{\varepsilon}_{av,y}(i)} + \frac{bb_z(i)}{C^{\lambda}_{av,z}}\right]\right\}$$

$$Time_{cost} = \sum_{\forall i,j\ \in application} \{Delay(i,j) \times criticality(i)^{const}\}$$

In this cost function, factors α of cost function balance the effort for reducing either the total wire length or the delay. However, in both cases, the algorithm tries to reduce the switching activity. The $bb_x(i)$, $bb_y(i)$ and $bb_z(i)$ parameters denote the dimensions of the 3D bounding box for network i, while the $q(i)$ is a scaling factor of the bounding box, used to make more accurate estimations about the wire-length for nets with more than 3 terminals [10]. The $delay(i,j)$ denotes the delay between a source-sink path of a network, the factor *const* is a constant, while the $criticality(i)$ gives the importance, in terms of how close to the critical path, is the network i. Finally, the $activity(i)$ represents the switching activity value for the network i. In order to calculate this parameter, the transition density for all the hardware elements of network i has to be summarized.

The $C_{av,x}(i)$, $C_{av,y}(i)$ and $C_{av,z}(i)$ parameters represent the average width of routing tracks across the x, y and z direction, respectively, for the bounding box of network *i*, while they are used in order to be taken into consideration the available routing resources during the placement. Their values depend solely on the fabricated interconnection resources, while they are constant during the placement. The values of ε and λ control the relative cost of employing narrower and wider routing channels.

More specifically, when their values are 0, then the cost function results to the conventional bounding box approach. Otherwise, as higher the values of these parameters are, then more and more tracks from narrowest routing channels have increased cost value, compared to the wider channels. We employ a different relative cost (λ) for the vertical interconnections, as the placement algorithm has to pay effort to not waste this kind of connections. Finally, by using an additional factor, denoted as r, we discourage the placer to put functions that exchange data in different layers.

```
Function PLACE(netlist, target architecture)
P ← Make an initial Placement();
T ← Initial Temperature;
R_limit ← Initial R_limit;
While (Exit_Criterion() not TRUE)  // outer loop
   {
   While (loop_criterion() not TRUE)  //inner loop
      {
      P_new ← Random swap placements(P, P_limit);
      ΔCost ← Cost(P_new) – Cost(P);
      r ← random value(0,1);
      if (r<e^-ΔCT) P ← P_new;  // accept movement
      }
   R_limit ← Update(R_limit);
   T ← Update(Temperature)
   }
End Function
```

Algorithm 3: The proposed temperature-aware placement algorithm

Even though it is true that such an approach can reach arbitrary close to the global minimum, if the cooling schedule is slow enough, it suffers from long run times for large circuits. In contrast to most of the existing approaches that start from a random initial placement, our solution employs a more "*sophisticated*" assignment of logic blocks, leading to shorter runtimes. This is achieved by taking into consideration during the initial placement apart from the timing and the wire-length constraints, the minimization of switched capacitance variation. Such info is available from the partitioning step (shown in previous section).

4.3 Application Routing

By defining the placement on the 3D FPGA, the routing algorithm forms the appropriate connections among the utilized logic blocks (C) through the available interconnection fabric (W). As the vertical interconnections are limited, compared to horizontal tracks, the routing algorithm sets their weight to a higher value, in order to discourage the unnecessary bends between horizontal and vertical wires. Also, this

penalty forces the router not to connect logic blocks placed on one layer by using interconnection fabric from different layers.

The proposed routing algorithm (shown in Algorithm 4) is based on Pathfinder negotiated congestion [4]. Initially, a number of networks are allowed to share the same routing fabric, which is gradually prohibited, until to the final routing where every network employs dedicated routing fabric. Such an approach finds the narrowest horizontal and vertical channels for which the application is fully routable.

```
Function ROUTE(netlist, target architecture)
horizontal ← initial horizontal channel width;
vertical ← initial vertical channel width;
while (routing ← optimal connection of logic blocks) do
route netlist();
   if (succeed routing) then
      optimal routing ← find narrowest channels;
      do
         T ← meet timing constraints();
         if(T ← True) distribute switched capacitance();
      while(T ← True)
   else increase horizontal/vertical channel widths;
end while
end Function
```

Algorithm 4: The proposed temperature-aware routing

By discouraging routing algorithm to form connections that cross *hotspot* regions, it is possible to spread the switched capacitance over the 3D device, while it also achieve the timing and total power/energy constraints. However, this is not always feasible, as it might increase the application's delay or its power/energy consumption. The efficiency of a derived application routing, is quantified with the cost function. The mathematical expression regarding this function is shown in Equation (5).

$$\begin{aligned}\Delta Cost(n) = {} & Criticality(i) \times Delay(i,j) + \big(1 - Criticality(i)\big) \\ & \times [\gamma \times switching(i) \times capacitance(n) + (1 - switching(i)) \\ & \times b(n) \times h(n) \times p(n)]\end{aligned} \tag{5}$$

In this expression, the factor γ defines the importance of temperature control during the routing procedure. The parameters $b(n)$, $h(n)$ and $p(n)$ represent the base cost, the historical congestion cost and the present congestion cost for the hardware element n, respectively. In order to come to acceptable solutions the value of $p(n)$ increases with the execution time, in order to avoid the overuse of routing resources. The factor $capacitance(n)$ corresponds to the normalized capacitance of resource n, while the $switching(i)$ refers to the importance of the switching activity for the network i. Equation (6) gives the mathematic expression of this parameter.

$$switching(i) = \min\left\{Max\_switching, \frac{switching(i)}{maximum\_switching}\right\} \tag{6}$$

Here the $Max\_switching$ corresponds to the maximum allowed value of switching activity regarding the network i, while the ratio $\frac{switching\ (i)}{maximum\ \_switching}$ gives the normalized switching activity over all the application's interconnection networks. As the value of the $Max\_switching$ parameter closes to 1 (*e.g.*, 0.99), then more and more interconnection networks with high switching activity will be taken in consideration during the routing congestion.

Comparing the proposed cost function with existing from literature [8, 10, 13], it has identical timing parameter. So, whenever a routing connection is timing critical, the routing algorithm pays effort to reduce the delay of the network. However, the second part of the cost function tries to handle the spatial distribution of switched capacitance. Whenever a connection exhibits high switching activity, the proposed algorithm tries to form the required connections through paths that exhibit reduced capacitance (in order to eliminate the temperature values). On the other hand, when the networks do not exhibit increased switching activity, the routing algorithm reduces the routing congestion and increases the application's operation frequency.

5. Experimental Results

We implement the proposed temperature-aware P&R algorithm in C++, as part to an existing open-source tool for 3D FPGAs, named 3DPRO [8]. The experimental results were retrieved using the 3DPRO tool, without and with the power-aware P&R feature. This section provides comparisons among the proposed temperature-aware P&R algorithm and the alternative solutions found in relevant literature, considering the 20 biggest MCNC benchmarks. The average complexity of the employed benchmark circuits, as listed in Table 1, is about 3,410 4-input LUTs, while each of the layers contains an array of 56×56 slices. In terms of the target 3-D devices, the average percentage of utilized logic resources is almost 97%.

The target 3D PFGA platforms (were each of the benchmarks is mapped) is inspired by the one proposed in [8]. Such a 3D device is constructed by stacking a number of identical 2D FPGAs on individual functional layers, providing appropriate communication among them by interlayer vias. These connections are realized inside vertically adjacent 3D Switch Boxes (SBs). The employed 3D architecture has a vias distribution with smaller fabrication costs compared to conventional 3D FPGAs, without any degradation in application performance, or increment of total power/energy consumption. The features of this architecture are summarized as follows:

- It consists of four functional layers ($Z = 4$).
- The percentage of vertical interconnections (i.e. vias) per functional layer is 30% (as derived in [8]).
- The spatial location (x, y) of each vertical interconnection per layer remains invariant.
- The vertical interconnection fabric was modeled based on the approach shown in [3].
- There are 4 bit connections between layers for each 3D SB.

- The hardware resources (both logic and interconnection) among layers are identical.
- Each application is P&R onto the smallest 3D FPGA.
- The employed 3D devices for the alternative mappings have identical hardware resources.

Table 1: Complexity of the employed benchmark applications, and utilization of the logic resources of the target 3-D FPGA.

Benchmark	# of 4-LUTs	2D FPGA		3D FPGA	
		FPGA array	% utilized logic resources	FPGA array	% utilized logic resources
alu4	1544	40×40	96.50%	20×20×4	96.50%
apex2	1920	44×44	99.17%	22×22×4	99.17%
apex4	1290	36×36	99.54%	18×18×4	99.54%
bigkey	2391	49×49	99.58%	25×25×4	95.64%
clma	8879	95×95	98.38%	48×48×4	96.34%
des	2092	46×46	98.87%	23×23×4	98.87%
diffeq	1974	45×45	97.48%	23×23×4	93.29%
dsip	2020	45×45	99.75%	23×23×4	95.46%
elliptic	4969	71×71	98.57%	36×36×4	95.85%
ex1010	4618	68×68	99.87%	34×34×4	99.87%
ex5p	1135	34×34	98.18%	17×17×4	98.18%
frisc	4561	68×68	98.64%	34×34×4	98.64%
misex3	1425	38×38	98.68%	19×19×4	98.68%
pdc	4631	69×69	97.27%	35×35×4	94.51%
s298	1948	45×45	96.20%	23×23×4	92.06%
s38417	7694	88×88	99.35%	44×44×4	99.35%
s38584	7884	89×89	99.53%	45×45×4	97.33%
seq	1826	43×43	98.76%	22×22×4	94.32%
spla	3752	62×62	97.61%	31×31×4	97.61%
Tseng	1605	41×41	95.48%	21×21×4	90.99%
Average:	**3407.9**	**56×56**	**98.37%**	**29×29×4**	**96.61%**

Figure 3 depicts the variation of switched capacitance over the layers of the 3D FPGA, with a timing-aware mapping. The employed application, named *alu4*, is one of the 20 biggest MCNC benchmarks, consisted of 1522 4-input LUTs. These logic modules are assigned to four equal sized layers, each of which occupies an array of 20×20 slices. The picture of Figure 3 is a very useful instrument to architecture designers, in order to specify the spatial distribution of *hotspot* regions over the device.

From this figure we conclude that the switched capacitance vary a lot, even for hardware resources assigned to adjacent spatial locations onto the same layer. Also, it is possible to locate regions on the layers with excessive high values of switched capacitance. In order to understand the thermal characteristics and prevent circuit failure, it is important to detect such *hotspots* regions. By specifying their spatial

distribution, the designer can concentrate his/her efforts to control the switched capacitance on certain regions only, but not on the whole device, reducing among others the design/fabrication cost.

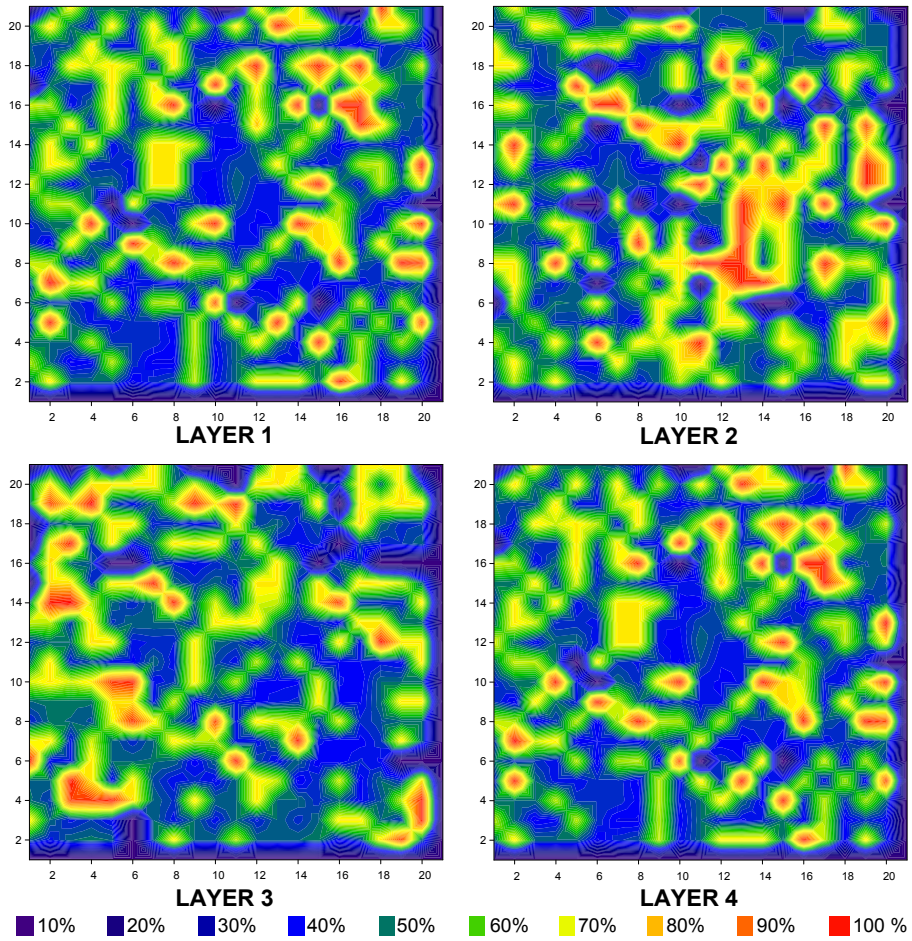

Figure 3: The variation of switched capacitance for *alu4* benchmark with a conventional P&R algorithm

The proposed temperature-aware P&R algorithm can assist to provide a solution to this problem, as it is aware about the distribution of switched capacitance across the 3D FPGA. Figure 4 plots the corresponding variation of switched capacitance regarding the same application and 3D device, for the proposed algorithm. In contrast to the conventional approach (shown in Figure 3), the proposed one exhibits more balanced variation of switched capacitance, and hence for power consumption and for on-chip temperature.

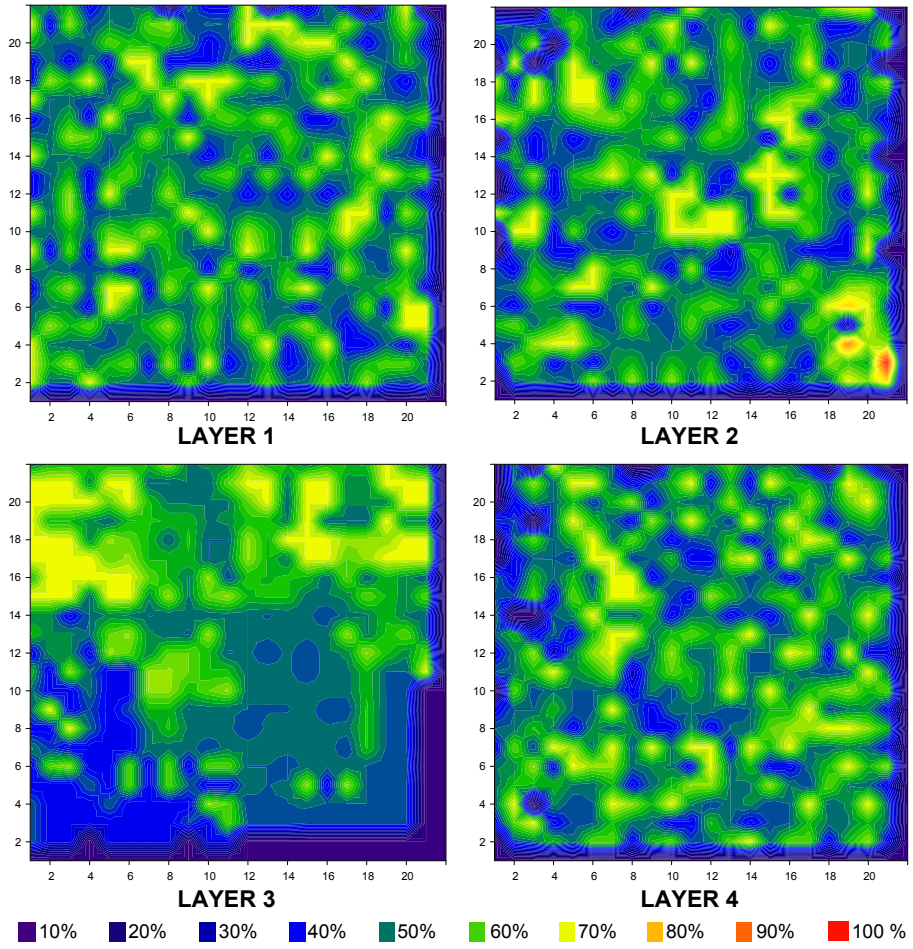

Figure 4: The variation of switched capacitance for *alu4* benchmark with the proposed temperature-aware P&R

Additionally, the maximal values of switched capacitance are lower, leading to cheaper and more reliable devices. One more conclusion might be derived from these two graphs. More specifically, the proposed approach distributes more uniformly the switched capacitance for the layers placed on the middle of the 3D stack. This feature is critical for the thermal efficiency of the target 3D architecture, as it is more difficult to dissipate heat from these layers.

For shake of completeness we employ the proposed temperature-aware P&R algorithm for two setups. Both of them were realized by appropriately tuning the parameters of the cost functions. More specifically, the first of them affects an approach where the importance of application's delay is thought to be similar to the temperature distribution, while in the second experimental setup, the employed cost functions are tuned to achieve even more uniform temperature distribution on the 3D FPGA.

Figure 5 compares the average (over the 20 biggest MCNC benchmarks) area percentage of the 3D FPGA that operates under high power sources for the two flavors of the proposed approach (non-aggressive and aggressive) against to conventional P&R. As we may conclude, the proposed solution achieves to reduce the percentage of area that operates under high power values (*i.e.*, belonging to *hotspot* regions), while it spreads these power sources on the rest device in a more uniformly manner. This is especially critical for designing reliable and cheaper devices, as there is no need for expensive packaging solutions. Moreover, by transferring power consumption from *hotspot* regions to the rest architecture, we increase the device reliability and reduce its fabrication cost.

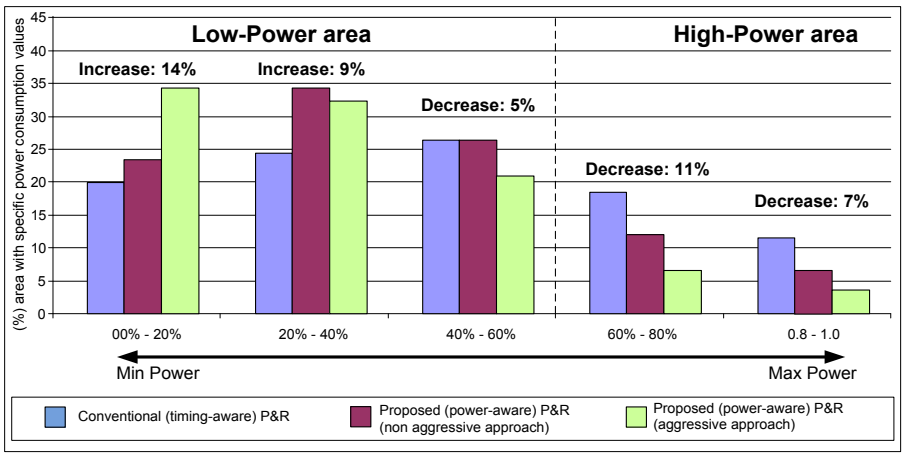

Figure 5: Variation of area percentage that operates under specific power consumption for alternative P&R algorithms

Apart from distributing uniformly the on-chip temperature; the proposed P&R algorithm pays effort not to increase either the application's delay or its total power/energy consumption. The upcoming Tables summarize the evaluation results of applying the proposed strategy to the 20 biggest MCNC benchmarks. The target 3D FPGA device was described in the beginning of this section, while the array dimensions for each benchmark is derived from Table I.

As we have already mentioned, the interconnection network contributes to the temperature of target 3D architectures. Table 2 compares the total wire-length for application mapping onto 2D and 3D FPGAs. Based on the results, the proposed temperature-aware approach leads to increased wire-length (between 6% and 22%), as compared to a timing-aware P&R. However, both of temperature-aware flavors results to smaller wire-lengths than the implementation targeting 2D architectures [8] about 19%. As we will prove later, the increased values of this parameter cannot outperform the advantages of realizing applications with the proposed power-aware P&R algorithm.

Table 2: Comparison in terms of total wire-length ($\times 10^3$m) for alternative P&R algorithms

Benchmark	2D FPGA		3D FPGA		
	Timing-aware P&R	Temperature-aware P&R [18]	Timing-aware P&R	Proposed (Temperature-aware) P&R	
				non- Aggressive	Aggressive
alu4	50.23	54.46	37.09	35.65	37.81
apex2	67.67	65.63	53.16	56.06	58.99
apex4	62.39	65.69	37.92	38.16	37.76
bigkey	60.01	68.15	43.68	48.57	32.71
clma	301.42	309.4	280.5	294.6	430.44
des	57.46	62.09	43.82	45.44	52.7
diffeq	46.77	54.48	31.08	36.31	34.88
dsip	45.81	49.56	33.69	35.46	29.9
elliptic	114.74	121.38	94.91	92.15	102.24
ex1010	41.34	46.55	31.47	33.78	36.9
ex5p	163.43	169.93	146.42	167	129.65
Frisk	108.35	114.56	91.26	99.83	174.05
misex3	50.56	56.91	37.88	38.26	39.31
pdc	174.33	183.36	160.37	173.32	238.78
s298	59.68	65.44	42.3	44.65	55.85
s38417	170.63	182.39	155.97	169.52	172.01
s38584	147.55	160.01	136.39	152.62	129.14
seq	63.46	68.55	48.74	54.49	52.8
spla	139.45	148.62	125.03	113.47	148.19
tseng	35.45	41.09	21.07	23.85	25.89
Average:	**98.04**	**104.41**	**82.64**	**87.66**	**101.00**

Table 3 gives the delay for each of the 20 biggest MCNC benchmark with the usage of alternative P&R algorithms for 2D and 3D FPGAs. Based on the results, the proposed temperature-aware P&R algorithm increases slightly the application's delay, ranging from 1% up to 5%, while the performance improvement compared to solution targeting 2D FPGAs [7] is up to 27%. The almost negligible performance degradation (due to the extra constraints for forming connections) is acceptable, as it does not lead to significant variation of the application's functionality.

Table 4 gives the energy requirements for each of the 20 biggest MCNC benchmarks. As we may conclude from the results, the proposed temperature-aware P&R algorithms (non-aggressive and aggressive) lead to energy savings, compared to conventional (*i.e.*, timing-aware) P&R that range between 4% and 9%, respectively. Additionally, the energy savings compared to corresponding solution from literature [7] is up to 29%.

Table 3: Comparison in terms of delay (10^{-9}sec) for alternative P&R algorithms

Benchmark	2D FPGA		3D FPGA		
	Timing-aware P&R	Temperature-aware P&R [18]	Timing-aware P&R	Proposed (Temperature-aware) P&R	
				non-Aggressive	Aggressive
alu4	9.77	8.69	7.32	7.51	7.71
apex2	9.37	9.65	7.87	8.04	8.12
apex4	8.86	9.1	6.78	6.96	7.29
bigkey	6.03	9.71	5.14	4.96	5.22
clma	10.1	9.23	8.84	9.95	10.47
des	7.76	11.1	6.78	6.65	7.34
diffeq	6.15	6.32	7.71	7.67	7.77
dsip	7.99	10.9	5.14	4.99	5.13
elliptic	10.9	11.7	7.34	7.63	7.69
ex1010	18.3	18.3	7.87	8.29	8.52
ex5p	9.26	7.17	5.95	5.96	6.28
Frisk	16.1	13.4	6.51	6.62	6.87
misex3	11.7	8.34	7.32	7.68	7.79
pdc	20.4	18.7	8.41	8.37	8.52
s298	13.4	13.4	11.69	11.54	11.69
s38417	9.85	9.79	10.95	11.35	11.43
s38584	7.45	7.85	10.47	10.12	10.64
seq	9.52	8.17	7.32	7.41	7.72
spla	15.6	18	7.87	7.62	8.14
tseng	5.55	5.53	7.61	7.48	7.88
Average:	**10.70**	**10.75**	**7.74**	**7.84**	**8.11**

Finally, we study the percentage of device area that operates under high power. This part of device area is mentioned as *hotspot* region, while its reduction is the main goal of the developed research. The two flavors of the proposed power-aware P&R algorithm achieve to reduce this percentage, compared to conventional (*i.e.*, timing-aware) P&R for the same 3D FPGA ranging from 30% up to 63%. Moreover, the reduction of area coverage for *hotspot* regions compared to existing approaches for 2D FPGAs [7] is about 58%.

The results presented in this section prove that the proposed temperature-aware P&R achieves to reduce the percentage of silicon area that operates under high power/temperatures (*hotspot* regions), which is the main goal of our research, without impact on other critical design parameters, even though there is an increase in total wire-length. This occurs due to the better application partitioning, partition to layer assignment, placement and routing.

Table 4: Comparison in terms of energy dissipation (10^{-9} Joule) for alternative P&R algorithms

Benchmark	2D FPGA		3D FPGA		
	Timing-aware P&R	Temperature-aware P&R [18]	Timing-aware P&R	Proposed (Temperature-aware) P&R	
				non- Aggressive	Aggressive
alu4	5.83	5.82	4.45	4.21	4.38
apex2	6.75	6.81	6.39	5.28	4.85
apex4	4.17	4.23	3.68	3.26	2.96
bigkey	7.95	8.27	7.10	8.49	7.76
clma	75.6	79.9	32.49	33.44	33.69
des	10.5	11.4	10.14	9.76	9.22
diffeq	3.51	3.5	9.28	8.35	8.27
dsip	7.82	8.01	5.58	5.77	5.10
elliptic	12.4	12.8	9.71	9.87	10.17
ex1010	16.2	16.3	9.91	8.31	8.68
ex5p	4.32	4.1	3.19	3.10	2.97
Frisk	12.1	11.2	19.01	17.02	15.70
misex3	5.55	5.25	4.12	3.79	4.01
pdc	22.3	21.4	14.02	15.83	13.70
s298	6.88	6.95	7.45	7.49	6.83
s38417	22.9	23.1	34.48	29.10	26.99
s38584	43.2	35.6	23.95	24.08	21.28
seq	6.32	6.08	5.72	4.70	5.02
spla	13.1	13.9	9.58	9.04	8.55
tseng	3.18	3.2	15.05	16.09	14.30
Average:	**15.13**	**14.98**	**11.76**	**11.35**	**10.72**

The gains of employing the proposed temperature-aware P&R approach targeting 3D FPGAs can be summarized as follows: (*i*) it spreads the power/temperature sources across the 3D FPGA in a way that it is more easy to dissipate heat, (*ii*) it reduces the peak values of power/temperature sources leading to cheaper fabrication cost for cooling, (*iii*) it reduces the total energy consumption increasing among other the battery life and the system's reliability, and (*iv*) it reduces significantly the percentage of silicon area that operates under high power/temperature consumption values (i.e. *hotspot* regions), which can be thought as a power/temperature management approach.

Table 5: Comparison in terms of area percentage that operates under high temperature values (*i.e.*, hotspot regions) for alternative P&R algorithms

Benchmark	2D FPGA		3D FPGA		
	Timing-aware P&R	Temperature-aware P&R [18]	Timing-aware P&R	Proposed (Temperature-aware) P&R	
				non- Aggressive	Aggressive
alu4	34%	17%	26%	14%	9%
apex2	41%	23%	30%	18%	9%
apex4	42%	29%	31%	23%	14%
bigkey	34%	19%	24%	15%	8%
clma	25%	12%	18%	10%	6%
des	39%	26%	30%	21%	9%
diffeq	37%	25%	28%	20%	9%
dsip	39%	27%	30%	22%	11%
elliptic	42%	30%	30%	26%	15%
ex1010	26%	15%	19%	12%	5%
ex5p	27%	17%	20%	14%	8%
Frisk	33%	24%	25%	19%	9%
misex3	31%	23%	23%	18%	11%
pdc	31%	26%	23%	17%	10%
s298	31%	24%	23%	14%	9%
s38417	27%	13%	20%	10%	7%
s38584	45%	30%	34%	25%	12%
seq	40%	31%	30%	24%	13%
spla	32%	23%	23%	18%	10%
tseng	33%	23%	25%	19%	8%
Average:	**34%**	**23%**	**26%**	**18%**	**9.6%**

6. Conclusions

A novel temperature-aware P&R algorithm targeting to 3D FPGAs, as well as its software implementation at 3DPRO tool, was presented. This approach could also be used as a power management strategy, since it achieves to re-distribute the power budget over identical hardware resources in a way that the produced heat is easily to be dissipated. More specifically, the proposed P&R algorithm reduces about 63%, in average, the percentage of device area that operates under high temperature by appropriately controlling the switched capacitance. In addition to that, we achieve energy savings about 9% (in average), with an almost negligible penalty (ranging from 1% up to 5%) in application's delay.

Acknowledgment

This paper is part of the 03ED593 research project, implemented within the framework of the "Reinforcement Program of Human Research Manpower" (PENED) and co-financed by National and Community Funds (75% from E.U.-European Social Fund and 25% from the Greek Ministry of Development-General Secretariat of Research and Technology).

References

[1] "International Technology Roadmap for Semiconductors", (http://www.intel.com/technology/silicon/itroadmap.htm)
[2] K. Poon, A. Yan and S. Wilton, "A Flexible Power Model for FPGAs", 12th International Conference on Field-Programmable Logic and Applications, pp. 312 – 321, 2002
[3] S. Gupta, M. Hilbert, S. Hong and R. Patti**,** "Techniques for producing 3D ICs with High-Density Interconnect", VLSI Multi-Level Interconnection Conference, 2004
[4] L. McMurchie, C. Ebeling, "Pathfinder: A Negotiation-Based Performance-Driven Router for FPGAs", ACM/SIGDA Int. Sym. on Field-Programmable Gate Arrays, pp.111-117, 1995.
[5] http://proteas.microlab.ntua.gr
[6] A. Telikepalli, "Designing for Power Budgets and Effective Thermal Management," In Xcell Journal, Issue 56, 2006.
[7] K. Siozios and D. Soudris, "A Novel Methodology for Temperature-Aware Placement and Routing of FPGAs", *IEEE Computer Society Annual Symposium on VLSI, pp. 55-60, 2007*
[8] K. Siozios, et.al., "Exploring Alternative 3D FPGA Architectures: Design Methodology and CAD Tool Support", 17th Int. Conference on Field-Programmable Logic and Applications, pp. 652-655, 2007
[9] T.Y. Chiang, et.al., "Thermal Analysis of Heterogeneous 3D ICs with Various Integration Scenarios", International Electron Devices Meeting (IEDM), pp. 31.2.1-31.2.4, 2001
[10] V. Betz, et. al., "Architecture and CAD for Deep-Submicron FPGAs", Kluwer Academic Publishers, 1999.
[11] N. Selvakkumaran and G. Karypis, "Multiobjective Hypergraph-Partitioning Algorithms for Cut and Maximum Subdomain-Degree Minimization", IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems, Vol. 25, No. 3, pp. 504-517, March 2006
[12] T. Okamoto and J. Cong, "Buffered Steiner Tree Construction with Wire Sizing for Interconnect Layout Optimization", Int. Conference on Computer Aided Design, pp. 44-49, 1996.
[13] C. Ababei, et.al., "Placement and Routing in 3D Integrated Circuits", IEEE Design & Test of Computers, Vol. 22, No. 6, pp. 520-531, 2005.
[14] S. Das, et.la., "Timing, Energy, and Thermal Performance of Three Dimensional Integrated Circuits", 14th ACM Great Lakes symposium on VLSI, pp. 338-343, 2004
[15] S. Yang, "Logic Synthesis and Optimization Benchmarks, Version 3",Microelectronics Centre of North Carolina,1991.
[16] J. Cong, J. Wei and Y. Zhang, "A Thermal-Driven Floorplanning Algorithm for 3D ICs", International Conference on Computer Aided Design, pp. 306-313, 2004.
[17] Lesea, et. al., , "Powering Xilinx FPGAs", 2002