Sustainable Dual-Level DVFS-enabled NoC with On-chip Wireless Links

Jacob Murray, Rajath Hegde, Teng Lu, Partha Pratim Pande, Behrooz Shirazi
School of Electrical Engineering and Computer Science
Washington State University, Pullman, WA USA
E-mail: {jmurray, tlu, rhegde, pande, shirazi}@eecs.wsu.edu

Abstract

Wireless Network-on-Chip (WiNoC) has emerged as an enabling technology to design low power and high bandwidth massive multi-core chips. The performance advantages mainly stem from using the wireless links as long-range shortcuts between far apart cores. This performance gain can be enhanced further if the characteristics of the wireline links and the processing cores of the WiNoC are optimized according to the traffic patterns and workloads. In this work, we demonstrate that by incorporating both processor- and network-level dynamic voltage and frequency scaling (DVFS) in a WiNoC, the power and thermal profiles can be enhanced without a significant impact on the overall execution time. We also show that depending on the benchmark applications, temperature hotspots can be formed either in the processing core or in the network infrastructure. The proposed dual-level DVFS is capable of addressing both.

Keywords
WiNoC, DVFS, Power, Thermal, Latency

1. Introduction

Continuing progress and integration levels in silicon technologies make possible complete end-user systems on a single chip. This massive level of integration makes modern multi-core chips widely adoptable in multiple domains. In the design of high-performance massively multi-core chips, power and heat are dominant constraints. The increasing power consumption is of growing concern due to several reasons, e.g., cost, performance, reliability, scalability, and environmental impact. Increased power consumption can raise chip temperature, which in turn can decrease chip reliability and performance and increase cooling costs. Wireless Network-on-Chip (WiNoC) is an enabling technology to integrate very high numbers of cores in a single chip [1]. The on-chip wireless links help in designing single-hop shortcuts among distant nodes giving rise to the small-world network architecture [1]. By reducing the hop count between largely separated communicating cores, wireless shortcuts have been shown to carry a significant amount of the overall traffic within the network. The amount of traffic detoured in this way is substantial and the low power wireless links enable energy savings [2]. The overall energy dissipation of the WiNoC can be improved even further if the characteristics of the wireline links and the processing cores are optimized according to the traffic patterns and workloads respectively. Dynamic voltage and frequency scaling (DVFS) is a popular methodology to optimize the power usage/heat dissipation of electronic systems without significantly compromising overall system performance [3]. In this work our aim is to demonstrate how the power and thermal efficiencies of multi-core chips designed with on-chip wireless links can improve even further by incorporating suitable DVFS schemes in the wireline links and processing cores.

2. Related Work

DVFS can be applied to multi-core processors either to all cores or to individual cores independently [4]. Multi-core chips implemented with multiple Voltage Frequency island (VFI) design style is another promising alternative. VFI is shown to be effective in reducing on-chip power dissipation [5][6]. Various research groups have addressed design of appropriate DVFS control algorithms for VFI systems [7]. Some researchers have also recently discussed the practical aspects of implementing DVFS control on a chip, such as tradeoffs between on-chip versus off-chip DC-DC converters [4], the number of allowed discrete voltage levels [8], and centralized versus distributed control techniques [9]. Thermal-aware techniques are principally related to power-aware design methodologies using DVFS [10]. It is shown that distributed DVFS provides considerable performance improvement under thermal duress [10].

Most of the existing works principally addresses power and thermal management strategies for the processing cores only. Networks consume a significant part of the chip’s power budget; greatly affecting overall temperature. But, there is little research on how they contribute to thermal issues [11]. Thermal Herd, proposed in [11], provides a distributed runtime scheme for thermal management that allows routers to collaboratively regulate the network temperature profile and work to avert thermal emergencies while minimizing performance impact. For the first time, [12] addressed the problem of simultaneous dynamic voltage scaling of processors and communication links for real-time distributed systems. Intel’s recent multi-core-based single chip cloud computers (SCC) incorporate DVFS both in the core and the network-levels. However, all of the above-mentioned works principally consider standard multi-hop interconnection networks for the multi-core chips, the limitations of which are well known. Also, the principal emphasis has been on the design of DVFS algorithms, voltage regulators, and scheduling and flow control mechanisms. It is already shown that small-world network architectures with long-range wireless shortcuts can significantly improve the energy consumption and achievable data rate of massive multi-core computing platforms [1]. Here, we complement that effort by simultaneously addressing the power and thermal management of WiNoC-based multi-core processing
platforms by incorporating processor- and network-level DVFS.

3. Wireless Mesh Architecture

Traditionally, a mesh is the most popular NoC architecture due to its simplicity and the regularity of grid structure. However, one of the most important limitations of this architecture is the multi-hop communications between far apart cores, which gives rise to significant latency and energy overheads. To alleviate these shortcomings, long-range and single-hop wireless links are inserted as shortcuts on top of a mesh. It is shown that insertion of long-range wireless shortcuts in a conventional wireline NoC has the potential for bringing significant improvements in performance and energy dissipation [1] [13]. Inserting the long-range links in a conventional wireline mesh reduces the average hop count, and increases the overall connectivity of the NoC. In this wireless mesh (WiMesh), by careful placement of wireless links depending on inter-core distance and traffic patterns we enable savings in latency, energy, and heat dissipation.

3.1 Physical Layer Design

Suitable on-chip antennas are necessary to establish the wireless links. It has already been shown that wireless NoCs designed using carbon nanotube (CNT) antennas can outperform conventional wireline counterparts significantly [1]. Antenna characteristics of carbon nanotubes (CNTs) in the THz frequency range have been investigated both theoretically and experimentally [14]. Moreover, these antennas can achieve a bandwidth of around 500 GHz. Thus, antennas operating in the THz/optical frequency range can support very high data rates. Radiation characteristics of multi-walled carbon nanotube (MWCNT) antennas are observed to be in excellent quantitative agreement with traditional radio antenna theory [14], although at much higher frequencies of hundreds of THz. Such nanotube antennas are good candidates for establishing on-chip wireless links and are considered here.

3.2 Wireless Link Placement

Using CNT antennas, different frequency channels can be assigned to pairs of communicating source and destination nodes [1]. This will require using antenna elements tuned to different frequencies for each pair, thus creating a form of frequency division multiplexing (FDM), creating dedicated channels between a source and destination pair. This is possible by using CNTs of different lengths, which are multiples of the wavelengths of the respective carrier frequencies. High directional gains of these antennas, demonstrated in [14], aid in creating directed channels between source and destination pairs. In [15], 24 continuous wave laser sources of different frequencies are used. Thus, these 24 different frequencies can be assigned to multiple wireless links in the WiMesh in such a way that a single frequency channel is used only once to avoid signal interference on the same frequencies. This enables concurrent use of multi-band channels over the chip. Hence, in this work we assume 24 wireless links each with a single channel for the WiMesh architecture. The placement of the links is dependent upon three main parameters, the number of cores, the number of long-range links to be placed, and the traffic distribution. The aim of the wireless link placement is to minimize the hop count of the network. As discussed in [1], we optimize the average hop count weighted by the probability of traffic interactions among the cores, which can be seen by (1).

$$\mu = \sum_{i,j} h_{ij} f_{ij}$$

Where, $\mu$ is the optimization metric, $h_{ij}$ is the number of hops between routers $i$ and $j$, and $f_{ij}$ is the total number of flits from router $i$ to router $j$. In this way equal importance is attached to both inter-core distance and frequency of communication. Such a distribution is chosen because in the presence of a wireless link, the distance between the pair becomes a single hop and hence it reduces the original distance between the communicating routers in the network. Thus, highly communicating routers have a higher need for a wireless link. A single hop in this work is defined as the path length between a source and destination pair that can be traversed in one clock cycle.

Wireless link placement is crucial for optimum performance gain as it establishes high-speed, low-energy interconnects on the network. It is shown in [1] that for placement of wireless links in a NoC, the Simulated Annealing (SA)-based methodology converges to the optimal configuration much faster than the exhaustive search technique. Hence, we adopt a SA based optimization technique for placement of the wireless links in this work to get maximum benefits of using the wireless shortcuts. We also need to ensure that the introduction of wireless shortcuts does not give rise to deadlocks in data exchange. The routing adopted here is a combination of dimension order (X-Y) routing for the nodes without wireless links and South-East routing algorithm for the nodes with wireless shortcuts. This routing algorithm is proven to be deadlock free in [16]. Between a source and destination pair, the wireless links are only chosen if the wireless path reduces the total path length compared to the wireline path.

4. Dual-Level DVFS

The execution flow of a program running on a multi-core NoC generally contains periods of heavy computation followed by periods of inter-core data exchange. During periods of high computation, network usage may be at a minimum, allowing the voltage and frequency of links and routers to be tuned down in order to save energy and hence improve the thermal profile while not incurring a significant penalty to network latency. For the processing cores, DVFS can be implemented by exploiting the slack due to asynchronous memory events. Scaling down the frequency of the processor slows down CPU-bound operations (busy cycles), but does not affect the time taken by memory-bound operations (idle cycles) [4]. The presence of such memory-bound intervals gives rise to idle cycles in the processing cores that can be exploited to reduce voltage and frequency. The idle cycles present opportunity to save power while running an application. By creating a dual-level DVFS scheme, significant energy savings and subsequent thermal
improvements can be achieved to enhance the sustainability of the NoC. The DVFS schemes implemented among the cores and the network should be run independently of each other. The following subsections describe the processor-level and network-level DVFS implementations.

4.1. Processor-Level DVFS

The processor-level DVFS follows a profiled, window based methodology. The behavior pattern of the benchmark is collected in order to properly fit the known processor utilizations to voltage/frequency pairs. As the workload behaviors are dynamic and multi-threaded, the threads are assigned to particular cores in order to replicate the workloads and correctly assign voltage/frequency pairs. The pattern is then divided into windows. The processor utilization is determined within the given window, and based on the current voltage/frequency pair and the processor utilization, the voltage and frequency are tuned according to Table 1. A multi-threshold jump between voltage/frequency pairs will result in a high switching delay. Thus, only single threshold transitions are allowed. The switching penalty is a result of having to tune the voltage. We conservatively allow the voltage to transition between states without performing instructions during that time. Hence, during a switch between states, the processor incurs a 100 ns penalty [4].

4.2. Network-Level DVFS

The network-level DVFS follows a history based DVFS similar to [17]. History based DVFS implemented on a small-world based network with wireless shortcuts can be seen in [2]. In the proposed network, each router in the network predicts future traffic based on previous traffic seen by the router. Tuning a given link’s voltage and frequency is determined by the link utilization, $U$. Both short-term and long-term link utilizations are required to predict the future link utilization, and are characterized by (2) and (3).

$$U_{\text{Short}} = \frac{1}{H} \sum_{i=1}^{H} f_i$$

(2)

$$U_{\text{Predicted}} = \frac{C \times U_{\text{Short}} + U_{\text{Predicted}}}{C + 1}$$

(3)

Where, $H$ is the history window, $f_i$ is 1 if a flit traversed the link on the $i^{th}$ cycle in $H$, and 0 otherwise, and $C$ is the weight given to the short-term utilization over the long-term utilization. After every $T$ cycles, where $I/T$ is the maximum allowable switching rate, the router determines whether a given link’s predicted utilization meets one of the thresholds described in Table 1. Depending on which threshold was crossed, if any, the router then determines whether or not to tune the voltage and frequency of the link. In order to prevent multi-threshold jumps, just as in the processor-level DVFS, the voltage is allowed to step up once, step down once, or remain unchanged during one voltage/frequency transition.

After every $T$ cycles have elapsed, the energy savings and latency penalty due to the DVFS performed on those $T$ cycles is determined. The energy is determined by the number of flits that traverse the link over the switching interval, and the voltage level that the link is currently at. The latency penalty in the network is caused by two main factors. Similar to the processors, a 100 ns switching penalty occurs as the voltage is tuned, and the links do not transmit data during this time. The second penalty is due to mispredictions of the utilization. A misprediction penalty occurs when the adjusted voltage/frequency pair did not meet the bandwidth requirements of the traffic over the given switching interval. The bandwidth requirements of the link were obtained by viewing the current link utilization over a smaller window whose size was determined as the average latency of a flit in the non-DVFS network [2]. This penalty can be considered worst-case, as it assumes every flit is time-critical, and the processor receiving the data may be able to progress after the flit arrives.

<table>
<thead>
<tr>
<th>Voltage (V)</th>
<th>Frequency (GHz)</th>
<th>Utilization Threshold</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.5</td>
<td>≥ 0.9</td>
</tr>
<tr>
<td>0.9</td>
<td>2.25</td>
<td>≥ 0.8</td>
</tr>
<tr>
<td>0.8</td>
<td>2.0</td>
<td>≥ 0.7</td>
</tr>
<tr>
<td>0.7</td>
<td>1.75</td>
<td>≥ 0.6</td>
</tr>
<tr>
<td>0.6</td>
<td>1.5</td>
<td>≥ 0.5</td>
</tr>
<tr>
<td>0.5</td>
<td>1.25</td>
<td>&lt; 0.5</td>
</tr>
</tbody>
</table>

5. Performance Evaluation

In this section we characterize the performance of the WiMesh architecture by incorporating the dual-level DVFS through detailed full system simulations. We use the GEM5 platform, a full system simulator to obtain detailed processor and network-level information [18]. The WiMesh interconnection network was modified in GEM5 through Garnet. Wormhole routing where the packet is divided into fixed length flow control units (flits) is adopted here. We consider a system of 64 alpha cores running Linux within the GEM5 platform. Three SPLASH-2 benchmarks, FFT, RADIX, LU [19], and the PARSEC benchmark, CANNEAL [20] are considered to study the latency, energy dissipation, and thermal characteristics of the dual-level DVFS enabled WiMesh over traditional non-DVFS mesh architecture. Within GEM5, the benchmarks are run from start to completion, and statistics are obtained as outputs from GEM5.

Processor-level DVFS is implemented within the GEM5 simulator to obtain accurate execution time penalties and performance statistics when different parts of the benchmark are run with various voltage/frequency pairs depending on the busy/idle cycle requirements. The statistics generated by GEM5 simulations are incorporated into McPAT (Multi-core Power, Area, and Timing) [21], which is an integrated power, area, and timing-modeling framework for multithreaded, multi-core, and many-core architectures to determine the processor-level power statistics. Detailed data traversal through the network information of the benchmarks are obtained as GEM5 output and put into a network simulator. The simulator uses NoC routers synthesized from an RTL level design using 65nm standard cell libraries from CMP (http://cmp.imag.fr), using Synopsys™ Design Vision. Energy dissipation of the
network routers were obtained from the synthesized netlist by running Synopsys™ Prime Power, while the energy dissipated by wireline links was obtained through HSPICE simulations taking into consideration length of the wireline links. Each wireless link can sustain a data rate of 10Gbps and has an energy dissipation of 0.33pJ/bit, which is significantly less than even most efficient metal wires [1]. Network-level DVFS is performed within the simulator to determine network-level latency penalty and power.

After obtaining processor and network power values, the processors and the network routers and links are arranged on a 20mm x 20mm die. This floor plan along with the power values are input into HotSpot [22] to obtain steady state thermal profiles. The flow of the overall simulation is shown in Fig. 1. As temperature is closely related to the energy dissipation of the IC, the thermal profile depends on the energy dissipation of the cores and the NoC, which is quantified in Section 5.3. To compare the characteristics of the thermal profiles of a particular region in the chip, the average communication density and computation density are defined by (4) and (5),

$$\rho_{\text{comm}} = \frac{1}{t_c} \left( \sum_i \frac{N_{\text{flit}}}{N_{\text{router}}} + \sum_j \frac{N_{\text{flit}}}{N_{\text{link}}} \right)$$  \hspace{1cm} (4)

$$\rho_{\text{comp}} = \frac{1}{t_c} \sum_i \frac{N_{\text{ic}}}{N_{\text{core}}}$$  \hspace{1cm} (5)

where $\rho_{\text{comm}}$ is the average communication density in flits per cycle (FPC) over the region of interest, and $\rho_{\text{comp}}$ is the average computation density in instructions per cycle (IPC) over the region of interest. $N_{\text{router}}$ is the number of routers in the region of interest, $N_{\text{link}}$ is the number of links in the region of interest, $N_{\text{core}}$ is the number of cores in the region of interest, $N_{\text{fr}}$, $N_{\text{fl}}$, and $N_{\text{ic}}$ are the number of flits traveled on each router and link in that region, and the number of instructions performed on each core in that region respectively. The total number of cycles executed is $t_c$. If the average communication and computation densities of a given area on the chip are high, the temperature in that region will also be correspondingly high.

### 5.1. Execution Time Penalty

In this subsection, we evaluate the proposed dual-level DVFS WiMesh architecture in terms of its effects on the overall execution time. The execution time directly relates to the latency penalty. Fig. 2 shows the average flit latency within only the network. Here, it can be seen that the improvements created due to the wireless shortcuts are essentially balanced with the penalty introduced by the network-level DVFS. In fact, there is a slight improvement to the latency compared with the non-DVFS mesh by 4.46%, 6.07%, 5.18%, and 0.81% for the FFT, RADIX, LU, and CANNEAL traffics, respectively. Consequently, the overall execution time penalty will arise as a worst-case, by considering the execution time penalty introduced by the processor-level DVFS algorithm. Fig. 3 shows the execution time of the dual-level DVFS enabled WiMesh compared with the non-DVFS standard mesh architecture. The
execution time penalty due to DVFS in presence of FFT, RADIX, LU, and CANNEAL traffics are 7.24%, 7.07%, 7.68%, and 28.64%, respectively over the original non-DVFS mesh. As mentioned above, this penalty is due to the DVFS algorithm implemented on the processing cores. The penalties that arise from the SPLASH-2 benchmarks are relatively similar, however, in the CANNEAL benchmark there is a drastic increase in execution time penalty.

As mentioned above in section 4, implementation of DVFS in the processor level depends on the busy and idle cycles. The overall computation densities of the benchmarks are 0.76 IPC, 0.85 IPC, 0.80 IPC, and 0.48 IPC for FFT, RADIX, LU, and CANNEAL respectively. Intuitively, the assumption would be that since the computation density of CANNEAL is drastically lower, there is more opportunity to save energy without a significant impact to execution time due to the fact that there are periods of idle cycles, caused by memory-bound instructions. In the CANNEAL benchmark however, the computational trend, which transitions rapidly between busy and idle periods causes the voltage/frequency to switch often. This phenomenon can be seen in Fig. 4. Here, the number of switches that a single core performs from one voltage/frequency pair to another is, on average, slightly over 10,000 times when performing processor-level DVFS on CANNEAL traffic, which is a direct result of many busy-idle-busy segments within the processing cores. The SPLASH-2 benchmarks have much lower average switches per-core at 54, 208, and 346 for FFT, RADIX, and LU, respectively. Due to this frequent switching, the switching penalty observed in the presence of CANNEAL traffic becomes a dominant factor in the runtime of the benchmark.

5.2. Energy Dissipation

In this subsection, we evaluate the energy dissipation characteristics of the proposed dual-level DVFS WiMesh architecture. Fig. 5 describes the relative energy savings of the dual-level DVFS WiMesh architecture over the non-DVFS mesh architecture. From Fig. 5, a clear trend can be seen. The overall energy savings is decreasing from the FFT (33.85%), RADIX (21.93%), LU (20.59%), and CANNEAL (16.72%) benchmarks, respectively. By viewing the energy savings of only the network, between non-DVFS mesh and DVFS WiMesh, we can see a relatively constant amount of savings across the benchmarks. This is due to the fact that the benchmarks do not highly utilize the network, so we are effectively at a low injection rate within the network. Moreover, inserting long-range wireless links helps to improve the energy dissipation. There is a direct correlation with the network and processor savings and the decreasing overall savings, which requires further explanation.

Fig. 6 describes the problem clearly. In Fig. 6, the
energy contributions from the cores and network of the benchmarks are compared. As the energy contribution from the network is reduced, the lower savings of the more dominant core energy masks the significant network savings. Thus, as the energy contribution from the network reduces from 44% to 11% among the benchmarks, as seen in Fig. 6, the energy savings will mainly be seen as a result of the implementation of the processor-level DVFS. Realistically, as there are scenarios of high network energy and low computational energy and vice versa, this demonstrates the need to implement DVFS within both the processors and the network in order to maximize potential energy savings.

5.3. Thermal Profile

In this subsection, we present the overall thermal profile of the dual-level DVFS WiMesh architecture. By implementing dual-level DVFS, the thermal profile of the chip can be significantly improved. From Fig. 7, it can be seen that chip temperatures in the standard non-DVFS Mesh NoC are extremely high. The chip will not be able to sustain these high temperatures and reliable operation will be severely compromised. From [23], if the chip exceeds 115°C, the emergency temperature threshold will have been breached, jeopardizing the dependability of the chip. In the non-DVFS mesh architecture running CANNEAL, the network exceeds this emergency threshold, with a maximum temperature of 116.53°C. By implementing the dual-level DVFS on the WiMesh architecture, we have enhanced the reliability of the chip by drastically reducing the temperature much below the emergency threshold.

As mentioned above, there are several different thermal scenarios occurring within the benchmarks considered. In the FFT benchmark, the overall temperatures are generally low; however, a specific processor is dominating the thermal behavior, which has a much higher temperature than the average temperature of the chip. The computation density of the specific core (core number 50 in this case), \( \rho_{\text{comp50}} = 0.91 \) IPC. The average computation density of FFT is, \( \rho_{\text{compavg}} = 0.76 \) IPC. As the computational density of core 50 is much higher than the average, the core's energy is significantly higher than any other core that is not directly along the chip edge. This results in a hotspot formed at the single core. By enabling dual-level DVFS on the WiMesh, we have reduced the temperature of the hotspot location by 5.73°C.

In the RADIX benchmark the overall chip temperatures are still low; however, the work performed is uniformly distributed among the cores. The computation density among all the cores has an average, \( \rho_{\text{compavg}} = 0.84 \) IPC, and standard deviation, \( \rho_{\text{compsd}} = 0.056 \) IPC, hence, the hotspot is fully-distributed among the cores. As all of the cores are quite active, the energy dissipations among them are comparable. The hottest chip temperature was reduced by 5.75°C, which was very similar to the thermal savings in the FFT case.

The LU benchmark raised the overall thermal profile of the chip compared to the other SPLASH-2 benchmarks. Similar to the RADIX scenario, no single core is dominating the energy dissipated in the NoC, and hence, several hotspot regions are forming within the cores. As the computation density of the spots are high (\( \rho_{\text{comspot}} = 0.90 \) IPC) in comparison to the average computation density (\( \rho_{\text{compavg}} = 0.80 \) IPC), the chip is forming several distributed hotspots over the regions of higher computation density. The implementation of the dual-level DVFS WiMesh architecture has introduced a 7.51°C reduction to the hottest chip temperature.

The CANNEAL benchmark has introduced another interesting scenario among the thermal profiles. From Fig. 7, the thermal hotspot is forming within the network as opposed to the processors. As we saw in Fig. 6, the contribution of overall energy by the network running CANNEAL was only 11%. Thus, we need to investigate the reason behind the location of thermal problems in the network. Even though the energy contribution of the network has been reduced, the energy dissipation in the network has increased by an order of magnitude compared with the energy dissipated by the network in presence of the SPLASH-2 traffics. The communication density within the hotspot region of the network, \( \rho_{\text{comspot}} = 1.16 \) FPC, which is much higher than the average communication density, \( \rho_{\text{commavg}} = 0.71 \) FPC over the entire network. Hence, the area of the network forming the hotspot is due to the increased traffic density traversing the network. As the computation density of the cores is only 0.48 IPC, the network becomes the dominating thermal constraint for the chip. As the thermal issues arise in the network, a significant reduction in energy within the network (61% from Fig. 5) will ultimately result in a significant improvement to the thermal profile. Hence, we can observe that in case of CANNEAL, the hottest chip temperature is reduced by a significant 31.63°C and has averted the thermal emergency that was forming in the non-DVFS mesh architecture. By viewing the thermal profile of the dual-level DVFS WiMesh in presence of CANNEAL traffic, we notice the hotspot has effectively dispersed from the network. Now, however, a single router within the network has become the hotspot of the dual-level DVFS WiMesh as the amount of traffic that is traversing the router is an order of magnitude higher than any of the other routers. Consequently, we plan to investigate temperature-aware routing in addition to DVFS where any “hot” router can be bypassed in data exchange so that it does not fail due to excessive temperature issues.

6. Conclusions and Future Work

In this work, we have demonstrated how a dual-level DVFS mesh with wireless shortcuts improves both the energy and thermal profile of a multi-core chip at the cost of execution time. By adopting a small-world interconnection infrastructure, where long distance communications will be predominantly achieved through high performance specialized single-hop wireless links, communications can be made significantly more energy efficient. To further extend the energy savings dual-level DVFS on the wireline links and among the processors was implemented. We were able to save more energy and improve the thermal profile significantly.
Figure 7: Thermal Profiles of Non-DVFS Mesh vs. Dual-Level DVFS WiMesh
Top to Bottom: FFT, RADIX, LU, CANNEAL

Non-DVFS Mesh

- Single-Core Hotspot
  - $\rho_{\text{comp}_\text{So}} = 0.91\ \text{IPC}
  - $\rho_{\text{comp}_\text{avg}} = 0.76\ \text{IPC}$
  - $\rho_{\text{comp}_\text{So}} \gg \rho_{\text{comp}_\text{avg}}$
  - $\therefore T_{\text{comp}_\text{So}} \gg T_{\text{comp}_\text{avg}}$

- Fully-Distributed Hotspot
  - $\rho_{\text{comp}_\text{avg}} = 0.84\ \text{IPC}$
  - $\rho_{\text{comp}_\text{avg}} = 0.056\ \text{IPC}$
  - $\rho_{\text{comp}_\text{So}} \approx \rho_{\text{comp}_\text{avg}}$
  - $\therefore T_{\text{comp}_\text{So}} \approx T_{\text{comp}_\text{avg}}$

- Partially-Distributed Hotspot
  - $\rho_{\text{comp}_\text{spot}} = 0.90\ \text{IPC}$
  - $\rho_{\text{comp}_\text{avg}} = 0.80\ \text{IPC}$
  - $\rho_{\text{comp}_\text{spot}} > \rho_{\text{comp}_\text{avg}}$
  - $\therefore T_{\text{comp}_\text{spot}} > T_{\text{comp}_\text{avg}}$

- Network-Induced Hotspot
  - $\rho_{\text{comp}_\text{spot}} = 1.16\ \text{FPC}$
  - $\rho_{\text{comp}_\text{avg}} = 0.71\ \text{FPC}$
  - $\rho_{\text{comp}_\text{spot}} \gg \rho_{\text{comp}_\text{avg}}$
  - $\therefore T_{\text{comp}_\text{spot}} \gg T_{\text{comp}_\text{avg}}$

Dual-Level DVFS WiMesh

- Single-Core Hotspot
  - $\rho_{\text{comp}_\text{So}} = 0.91\ \text{IPC}$
  - $\rho_{\text{comp}_\text{avg}} = 0.76\ \text{IPC}$
  - $\rho_{\text{comp}_\text{So}} \gg \rho_{\text{comp}_\text{avg}}$
  - $\therefore T_{\text{comp}_\text{So}} \gg T_{\text{comp}_\text{avg}}$

- Fully-Distributed Hotspot
  - $\rho_{\text{comp}_\text{avg}} = 0.84\ \text{IPC}$
  - $\rho_{\text{comp}_\text{avg}} = 0.056\ \text{IPC}$
  - $\rho_{\text{comp}_\text{So}} \approx \rho_{\text{comp}_\text{avg}}$
  - $\therefore T_{\text{comp}_\text{So}} \approx T_{\text{comp}_\text{avg}}$

- Partially-Distributed Hotspot
  - $\rho_{\text{comp}_\text{spot}} = 0.90\ \text{IPC}$
  - $\rho_{\text{comp}_\text{avg}} = 0.80\ \text{IPC}$
  - $\rho_{\text{comp}_\text{spot}} > \rho_{\text{comp}_\text{avg}}$
  - $\therefore T_{\text{comp}_\text{spot}} > T_{\text{comp}_\text{avg}}$

- Network-Induced Hotspot
  - $\rho_{\text{comp}_\text{spot}} = 1.16\ \text{FPC}$
  - $\rho_{\text{comp}_\text{avg}} = 0.71\ \text{FPC}$
  - $\rho_{\text{comp}_\text{spot}} \gg \rho_{\text{comp}_\text{avg}}$
  - $\therefore T_{\text{comp}_\text{spot}} \gg T_{\text{comp}_\text{avg}}$
7. Acknowledgment

This work was supported in part by the US National Science Foundation (NSF) grants CCF-0845504, CNS-1059289, and CCF-1162202.

8. References


