HiMap: A Hierarchical Mapping Approach for Enhancing Lifetime Reliability of Dark Silicon Manycore Systems

Vijeta Rathore*, Vivek Chaturvedi*, Amit K. Singh†, Thambipillai Srikanthan*, Rohith R*, Siew-Kei Lam*, Muhammad Shafique‡

*School of Computer Science and Engineering, Nanyang Technological University (NTU), Singapore

†School of Computer Science and Electronic Engineering, University of Essex, UK

‡Institute of Computer Engineering, Vienna University of Technology (TU Wien), Austria

Corresponding authors: vijeta001@e.ntu.edu.sg, vchaturvedi@ntu.edu.sg, a.k.singh@essex.ac.uk, muhammad.shafique@tuwien.ac.at

Abstract—Technology scaling into the nano-scale CMOS regime has resulted in increased leakage and roadblock on voltage scaling, which has led to several issues like high power density and elevated on-chip temperature. This consequently aggravates device aging, compromising lifetime reliability of the manycore systems. This paper proposes *HiMap*, a dynamic hierarchical mapping approach to maximize lifetime reliability of manycore systems while satisfying performance, power, and thermal constraints. HiMap is process variation- and aging-aware. It comprises of two levels: (1) it identifies a region of cores suitable for mapping, and (2) it maps threads in the region and intersperses dark cores for thermal mitigation while considering the current health of the cores. Both the levels strive to reduce aging variance across the chip. We evaluated HiMap for 64-core and 256-core systems. Results demonstrate an improved system lifetime reliability by up to 2 years at the end of 3.25 years of use, as compared to the state-of-the-art.

***Index Terms*—lifetime, reliability, aging, mapping, manycore systems, dark silicon, process variation, optimization.**

I. Introduction and Related Work

Technology scaling has ushered the era of manycore systems, with Intel's 72-core Xeon Phi 7290F [1], Tilera's 144-core TILE-Gx [2] and a thousand core designs conceptualized [3], [4]. With failure of Dennard scaling, the voltage has scaled disproportionally to the feature size, thereby leading to increased power density and elevated chip temperature. To restrict the temperature, manycore systems function under the constraint of a maximum safe temperature or T_{safe}. On reaching T_{safe}, dynamic thermal management (DTM) is triggered which turns off the core(s) or reduces its(their) frequency, negatively impacting the performance. To avoid excessive DTM triggers and to avoid exceeding the cooling capacity, manycore systems are restricted by a power budget called the Thermal Design Power (TDP). Given the high power density, the TDP entails a significant fraction of the chip to remain off. It is termed as *dark silicon*; for 8 nm as much as 30% of the chip needs to be dark [5]. The elevated temperature accelerates the device aging mechanisms such as electromigation (EM), negative-bias temperature instability (NBTI), time-dependent dielectric breakdown (TDDB), etc. [6], which in turn jeopardize the lifetime reliability of the system. A 10°C rise in temperature can reduce the lifetime reliability by half [7]. Moreover, the on-chip process variation (PV) causes some cores to be slower than the rest (by as much as 30% [8]), making these more likely to getting slower due to the aging, further diminishing the lifetime reliability of the manycore system, especially when considering standard practices like guardbanding.

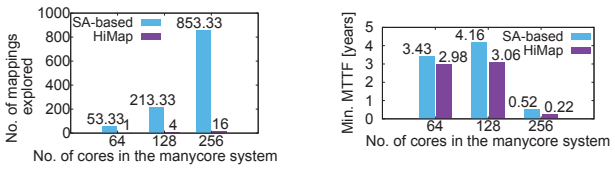
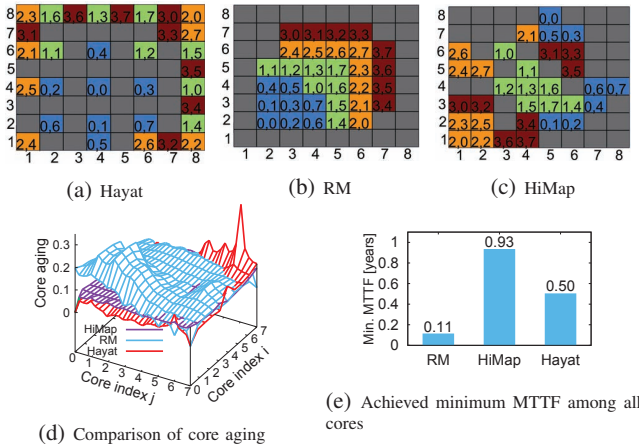


Fig. 1: **Left:** No. of mappings explored (normalized w.r.t. HiMap for 64-core case), **right:** minimum MTTF among all cores for the obtained mapping



(a) Hayat (b) RM (c) HiMap

(d) Comparison of core aging (e) Achieved minimum MTTF among all cores

Fig. 2: Mappings obtained from(different colors represent different applications): (a) Hayat [9], (b) RM [10], (c) proposed HiMap, along with a comparison of the core aging and achieved minimum MTTF among all cores

A. Related Work and Motivational Analysis

In the literature, there are several system-level approaches to improve lifetime reliability of multi-/manycore systems [6], [10], [11], [12]. However, most do not consider process variation, and dark silicon constraints and are computation intensive making them inadequate to match the scale of the manycore systems. For instance, Das et al. [6] devised a convex optimization-based mapping technique and Wang et al. [11] used sequential quadratic programming to find the optimal processor speeds while meeting the aggregate frequency constraints. Also, Huang et al. [12] have proposed a simulated annealing (SA) based mapping approach to improve lifetime reliability of multi-core systems.

We conducted a motivational study to compare the SA-based mapping approach [12] with our hierarchical mapping approach, HiMap. We selected SA-based approach for comparison since, being heuristic-based, it is the most scalable among the mentioned related works. We compared the number of explored mappings and achieved lifetime reliability for manycore systems, with 64, 128, and 256 cores. The workload comprised of 4, 6, and 9 benchmarks from SPLASH-2. As

shown in Fig. 1, SA-based approach explored a much larger number of mappings than HiMap, while improving the lifetime reliability by nearly a year. For instance, for a 128-core system, SA explored 52× more mappings (Fig. 1 (left)) with a lifetime (MTTF) improvement of 1.1 years (Fig. 1 (right)). We noted that for HiMap, the number of explored mappings is much smaller as the search space of mappings is greatly reduced (as explained later), while not heavily compromising the lifetime reliability, as seen in Fig. 1 (right). Thus, making it a more efficient mapping solution for manycore systems.

Haghbayan et al. [10] proposed a Reliability-aware Mapping (RM) approach for dark silicon manycore systems, however it does not explicitly considered PV. Gnad et al. [9] have proposed PV- and aging-aware mapping approach for dark silicon many-core systems, namely *Hayat*. We obtained mappings generated by Hayat, RM and our proposed HiMap, for a 256-core system considering the PV and a TDP constraint, as shown in Fig. 2. In this figure, each cell represents a core; the pair of integers on it indicates the application and thread assigned to the core. The gray cells are dark cores. It is seen that Hayat maps application threads in a non-localized manner (Fig. 2a), while RM maps the threads of an application in a contiguous region (Fig. 2b). Hayat improves aging by leveraging dark cores for thermal mitigation, albeit with an increased communication overhead as compared to the contiguous case. The contiguous mapping of RM on the other hand favors inter-thread communication, however it does not leverage dark cores for thermal mitigation. In HiMap, we strive to achieve thermal mitigation with the help of dark cores as well as contain the communication distance, as reflected by the mapping in Fig. 2c. Fig. 2d compared the core aging after one year of use. HiMap gives a more uniform aging as compared to both Hayat and RM. The minimum MTTF among all the cores after one year of dynamic mapping intervention is as shown in Fig. 2e. HiMap improved minimum MTTF by 0.43 year and 0.82 year, respectively, compared to Hayat and RM.

Wang et al. [13] interspersed dark cores in the region of application mapping, for thermal mitigation and performance optimization, however, they do not form the region based on blocks of cores. Carvalho et al. [14] created rectangular regions of cores or *clusters* to map applications, while HiMap forms clusters by selecting blocks, which gives it the flexibility to select cores from irregular locations as well. Kim et al. [15] proposed a learning-based dynamic voltage frequency scaling and dark core placement solution for dark silicon manycore to minimize energy consumption while meeting reliability, thermal and performance constraints. They considered aging of power grid networks due to EM. [16], [17] address soft-errors and aging holistically. [18], [19] investigate power-reliability trade-off for different V/f levels.

B. Target problem and Associated Scientific Challenges

Since workload assignment directly influences the power profile and chip temperature, judicious dynamic mapping solutions responsive to the prevailing aging can improve the lifetime reliability of cores in a manycore system. *This paper aims to find efficient and scalable dynamic mapping solution to improve the lifetime reliability of dark silicon manycore systems.* The solution needs to account for the workload characteristics and process variation, and assess core aging on-line. It can utilize dark cores for thermal mitigation, while containing the spread of the cluster for mapping an application. The time and power overhead of the approach should be low. Last, but importantly, it should be scalable with respect to (w.r.t.) the size of the manycore system and workload.

C. Our Novel Contributions and Key Features

The key contributions of our proposed dynamic PV- and aging-aware mapping approach are:

1) It determines the mapping and placement of dark cores to enhance the system lifetime reliability while meeting performance, thermal and power constraints, efficiently, through a hierarchical approach with two levels. The first level finds a cluster of cores to map an application. The second level ensures uniform aging within the cluster by placing threads causing more aging on relatively healthier cores, while also leveraging dark cores for thermal mitigation by interspersing within the cluster.
2) Through the first-level block-based selection, it prunes the design-space of possible mappings by a great extent, as discussed in the motivational analysis, making it scalable w.r.t. the manycore size.
3) It considers PV by incorporating variations in leakage and frequency. It also considers PV in the assessment of core aging and lifetime reliability.
4) It achieves improved lifetime reliability and more uniform aging as compared to the state-of-the-art techniques.

This is the first work, to the best of our knowledge, to present a **block-based** hierarchical mapping approach—for aging optimization and lifetime reliability enhancement.

II. System Model and Preliminaries

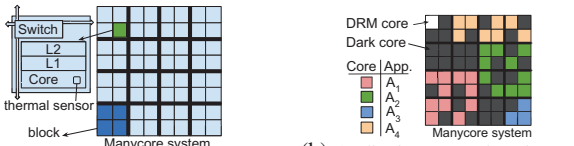
Manycore Processor Architecture and Process Variation Model: The manycore system is an $L_X \times L_Y$ grid of tiles. A tile consists of a core, a memory (private L1 and L2 caches) and a switch as shown in Fig. 3a. A set of cores is denoted as $C = \{C_{i,j}, \forall i \epsilon [1, L_X], j \epsilon [1, L_Y]\}$. The unused cores are power-gated, ensuring zero power for the sleeping cores. Each core has a thermal sensor for thermal monitoring. The on-chip network is of a mesh topology.

We use statistical process variation (PV) model presented in [20]. PV impacts the metal width and interconnect resistance [21], as given by:

$$W_{x,y} = \kappa_1 p_{x,y}\ (1a), H_{x,y} = \kappa_2 p_{x,y}\ (1b), Res_{x,y} = \gamma p_{x,y}\ (1c)$$

Maximum frequency of a core ($f_{i,j}$) is related to its nominal value f_{nom} as $f_{i,j} = \beta \min_{s,t \in S_{CP,i,j}} p_{s,t}$, where, $S_{CP,i,j}$ is the set of grid points containing critical paths. κ_1, κ_2, γ, and β is a technology specific constant. The leakage and dynamic power are also affected by PV as described in [20].

Application Model: We model workload as M periodic multi-threaded applications, $A = \{A_1, A_2, \ldots, A_M\}$ with periods given by $Q = \{Q_1, Q_2, \ldots, Q_M\}$. Application A_p has N_p threads, denoted by set $\{\tau_{p,1}, \tau_{p,2}, \ldots, \tau_{i,N_p}\}$. Thread $\tau_{p,q}$ has a performance requirement of $f_{req,p,q}$. Mapping of thread $\tau_{p,q}$ (of application A_p) to core $C_{i,j}$ is represented by the mapping function $m : \{\tau_{p,q}, p \epsilon [1, M], q \epsilon [1, N_p]\} \rightarrow \{C_{i,j}, i \epsilon [1, L_X], j \epsilon [1, L_Y]\}$



(a) Cores grouped in blocks (b) Applications mapped to clusters

Fig. 3: Illustration of blocks and application clusters

Aging and Lifetime Reliability Assessment: Without the loss of generality, we consider the aging due to EM, since it is one of the major aging phenomena affecting lifetime reliability of the interconnect of the manycore system. However, the proposed approach can work for other aging mechanisms as well by assessing the aging and lifetime reliability of the considered aging phenomena. EM affects power grid networks leading to increase in resistance and larger voltage drop [22]. As a result of which, the maximum operation frequency reduces. Similar to [22], we consider a core as faulty if its supply voltage drops by more than a threshold value of V_f. The lifetime reliability, measured as mean time to failure (MTTF), of a core is formulated as [22]:

$$MTTF = t_{growth,\Delta V=V_f} + t_{nuc} \tag{1}$$

where, $t_{growth,\Delta V=V_f}$ is the duration for the worst-case voltage drop to become V_f, and t_{nuc} is the nucleation time approximated as given by Eq. 2.

$$t_{nuc} \approx \tau^* e^{\frac{E_V}{kT}} e^{-\frac{f\Omega}{kT}(\sigma_{Res}+\sigma)} ln\left\{\frac{\sigma}{\sigma_{Res}+\sigma-\sigma_{crit}}\right\} \tag{2}$$

Eq. 3 approximates the growth of resistance:

$$\Delta Res(t) = v(t-t_{nuc})\left[\frac{\rho_{Ta}}{h_{Ta}(2H+W)} - \frac{\rho_{Cu}}{HW}\right] \tag{3}$$

The symbols involved are as defined in [22].

We consider the MTTF of the system as shortest MTTF among all the cores, as also considered in [6], [11], i.e.:

$$MTTF_{sys} = \min_{\forall(i,j)\in[1,L_X]X[1,L_Y]}\{MTTF_{i,j}\} \tag{4}$$

III. Problem Formulation

We formulate lifetime reliability enhancement as an optimization problem with the objective of maximizing the system MTTF assuming constraints of performance, temperature, and TDP. For the ease of illustration, we only explore mappings with up to one thread per core. Thus, HiMap aims at finding one-to-one mapping $m(\tau_{i,k}) = C_{p,q}$ to achieve:

$$\max \min_{\forall(i,j)\in[1,L_X]X[1,L_Y]} MTTF_{i,j} \tag{5}$$

Such that, the thermal constraint (Eq. 6a) and TDP (Eq. 6b) are satisfied. Also, for an application A_p's ($p\epsilon[1,M]$) thread $\tau_{p,q}$ ($q\epsilon[1,N_p]$), the performance constraint is satisfied, i.e., a thread is mapped to $C_{i,j}$, with a frequency greater than or equal to the thread's required frequency, $f_{req,p,q}$ (Eq. 6c).

$$T_{i,j} \leq T_{safe} \forall (i,j) \in [1,L_X]X[1,L_Y] \tag{6a}$$

$$\sum_{\forall(i,j)\in[1,L_X]X[1,L_Y]} P_{total,i,j} \leq TDP \tag{6b}$$

$$f_{i,j} \geq f_{req,p,q} \tag{6c}$$

IV. Proposed Hierarchical Mapping Approach

The concepts central to the proposed approach are:

- **Blocks**: To manage the complexity and scale of mapping exploration, we group cores into *blocks* of equal size, e.g. Fig. 3a shows a 64-core system with 2×2 sized blocks.
- **Clusters**: A cluster is a region of cores, formed by selecting some blocks, to which threads of an application are mapped. A cluster can have some dark cores as well, e.g., Fig. 3b shows 4 clusters with 4 applications mapped.
- **Inclusion of sleeping cores in the cluster:** To calculate the number of sleeping cores (n_{sleep}) and the number of blocks in the cluster, we solve Eq. 7, such that k and p are the smallest positive integers satisfying the equations. The ratio of running to sleeping cores in a cluster (R), the number of running cores (n_{run}), and block-size ($Size$) are known. k represents the number of blocks in the cluster, thus the cluster has $k * Size$ cores, which is the sum of running and sleeping cores in the cluster i.e. $n_{run}+n_{sleep}$. In case the cluster is of a size that is not a multiple of $Size$, we include p additional sleeping cores so that the entire block can be included in the cluster.

$$n_{run} + (n_{run}/R) + p = k * Size \tag{7a}$$

$$n_{sleep} = (n_{run}/R) + p \tag{7b}$$

$$k\epsilon\{1,2,3,\dots\}, p\epsilon\{0,1,\dots,Size-1\} \tag{7c}$$

- **Hyperperiod:** It is the least common multiple of all the application periods.
- **Epoch:** It is the period for our aging assessment and mapping intervention.
- **Health of a core:** A core's health is inversely related to the aging state, for EM we quantify it as the increment in the power grid resistance (ΔRes). We formulate health of a core as $Health = 1/(1+\Delta Res)$.

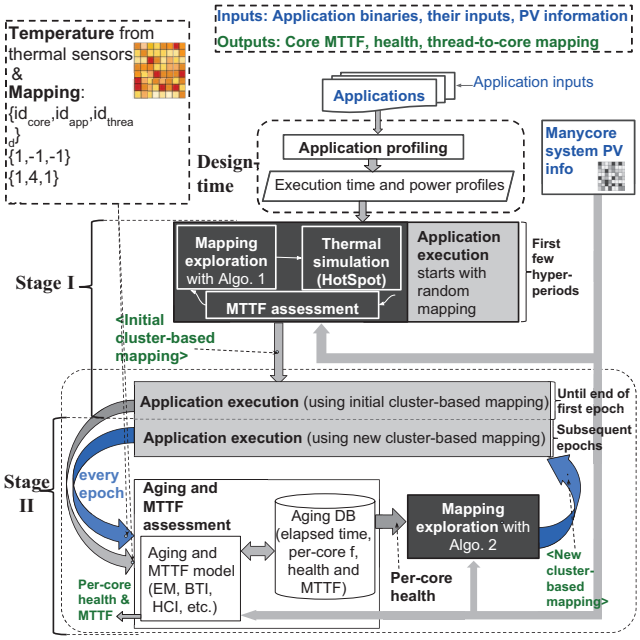


Fig. 4: Flow of the proposed HiMap approach

Fig. 4 shows the flow of HiMap. HiMap obtains execution time and average dynamic power profiles for all applications at design-time. It controls the mapping every epoch, and performs all mapping exploration at runtime, parallel to the application execution, on a dedicated core called dynamic lifetime reliability manager or *DRM core*. The mapping exploration is done in

two stages: initial cluster-based mapping exploration (*stage I*) and finding alternate cluster-based mapping to achieve uniform aging across the cores (*stage II*). In stage I, DRM core finds an initial cluster-based mapping to maximize $MTTF_{sys}$; it takes several hyperperiods. Concurrently, applications execute on cores as per a random mapping. Once the mapping exploration is over, applications are executed according to the obtained mapping until end of first epoch. Thereafter, stage II continues in which DRM finds a new mapping every epoch. As core-level aging profile varies over the epoch, it identifies the set of healthy cores and maps workload to these to ensure uniform aging across the cores.

Core Aging and MTTF assessment: To evaluate a mapping in terms of lifetime reliability, the DRM core performs per-core aging and MTTF assessment, for given PV, core temperatures and existing aging (if any). For stage I, it returns per-core MTTF corresponding to the explored mapping. During stage II, it assesses per-core aging for the duration of an epoch (as described in Sec. II for EM) and maintains an aging database with per-core aging, frequency, and health.

Stage I. Initial Cluster-based Mapping Exploration: In stage I, HiMap uses Algo. 1 to get the block-to-cluster assignment and thread-to-core mapping within the clusters. HiMap runs Algo. 1 for a set of block dimensions (formed by starting from the smallest i.e. 2×1 and 1×2, and incrementing in the X and Y directions by 1 in various combinations up to a medium block dimension of 3×3) and chooses the one that gives the best $MTTF_{sys}$, along with the corresponding blocks-to-cluster assignment and cluster-based mapping. The inputs of Algo 1 include set of applications A, their power (P_{dyn}) and execution time (E_{app}) profiles, number of blocks needed for applications ($num\text{-}blk$), PV information, T_{safe}, TDP, and average block frequency derived from the PV information. Algo. 1 takes parameters such as block assignment heuristic specified by the order to sort the blocks ($order_{blk}$) and the order to sort the applications ($order_{app}$), block dimension ($l_{blk} \times b_{blk}$), the maximum allowable distance $max\text{-}hop\text{-}count$ (it is the maximum number of blocks that a block being included to a cluster can be away from the center of the cluster yet formed), and a few parameters specific to SA including the cooling schedule (Th), number of rounds (Num_{rounds}), and number of steps (Num_{steps}). The outputs are the block-to-cluster assignment B and mapping m.

Block-to-cluster assignment is made in lines 1 to 5. We explored various block assignment schemes described in Tab. I for SPLASH-2 applications and the shown MTTF and maximum temperature. It indicates that assigning **H**igher **P**ower app. to **F**aster blocks (HPF) is the most beneficial in improving $MTTF_{sys}$, since faster blocks are healthier than the rest, making HPF our chosen block assignment scheme.

TABLE I: Comparison of block assignment schemes for SPLASH-2

Scheme	MTTF	T_{max} [K]
Higher **P**ower app. to **F**aster blocks (HPF)	1.07	351.60
Lower **P**ower app. to **F**aster blocks (LPF)	1.06	353.04
Longer **E**xecution time app. to **F**aster blocks (LEF)	1.03	356.15
Higher **P**ower app. to **S**lower blocks (HPS)	1.05	351.75
Higher **T**emp. app. to **S**lower blocks (HTS)	1.04	350.90
Higher **T**emp. app. to **F**aster blocks (HTF)	1.0	358.95

Algorithm 1: PV-aware block assignment and mapping exploration

Input: A, P_{dyn}, E_{app}, $num\text{-}blk_A$, F_{req}, C, PV, $F_{avg,blk}$, TDP, T_{safe}
Output: B, m
Parameters : $order_{blk}$, $order_{app}$, l_{blk}, b_{blk}, $max\text{-}hop\text{-}count$, Num_{rounds}, Num_{steps}, Th

1: Arrange blocks as per $order_{blk}$;
2: Arrange A as per $order_{app}$;
3: **for** All applications A_i in A **do**
4: Select first $num\text{-}blk_{A_i}$ available blocks, meeting $max\text{-}hop\text{-}count$, and assign to A_i and update block assignment B;
5: **end for**
6: Initialize mapping m with all cores sleeping;
7: **for** $j \leftarrow 1$ to $|A|$ **do**
8: In m, randomly map threads of A_i to cores in its cluster, s.t. F_{req,A_l}, TDP, T_{safe} are satisfied;
9: Intra-cluster-simulated-annealing(Num_{rounds},Num_{steps},Th,F_{req,A_j}, TDP, T_{safe},m);
10: **end for**
11: **return** B and m

Next, it conducts SA *within the cluster* for each application (function *Intra-cluster-simulated-annealing*, line 9) to obtain a Pareto mapping for enhancing $MTTF_{sys}$, instead of performing SA-based mapping over the entire chip as in [12]. The inputs to the function Intra-cluster-simulated-annealing include SA-specific parameters such as Th, Num_{rounds}, and Num_{steps}, along with each thread's frequency requirement ($f_{req,i,k}$), TDP, T_{safe}, and an initial random mapping through mapping structure m. It begins with a random thread-to-core mapping in the cluster and explores mappings obtained by assigning a randomly chosen thread to an earlier sleeping core. As shown in Fig. 4, it uses thermal simulation tool Hotspot [23] to get temperature for the explored mapping. Next, the MTTF assessment gives $MTTF_{sys}$. Finally, it returns the mapping with highest $MTTF_{sys}$, through m.

Stage II. Finding Alternate Cluster-based Mappings: It begins from the second epoch. At every epoch HiMap finds an alternate mapping using Algo. 2, right before end of the ongoing epoch to ensure uniform aging of the cores. Algo. 2 first performs the block to cluster assignment, and subsequently maps threads to cores of the assigned blocks. The inputs to Algo 2 include the application details, PV information, old mapping, old block to cluster assignment, updated core health values, and TDP and T_{safe} constraints.

We use heuristic HPF for block to cluster assignment as it was found to be the most beneficial among the explored heuristics during the generation of the initial cluster-based mapping, as shown in Tab. I (here T_max is the maximum chip temperature). Towards this, first, a list of all the blocks of the system is made and arranged in a decreasing order of average health of the cores (ln 2). Next, we sort applications in decreasing order of average power (ln 3). Starting with the highest average power application the following is done. The blocks of the cluster of application, A_j, are arranged in a decreasing order of number of running cores as $B_{cluster,j}$ (ln 5). Next, for each block b in $B_{cluster,j}$ (ln: 6), it sorts threads in decreasing order of average power as list $threads$ (ln: 7). For block assignment, it chooses the fastest available block (b') among B such that it is at worst $max\text{-}hop\text{-}count$ blocks away from the already chosen blocks (ln 8). Next, core of b' are arranged in decreasing order of health as list $cores\text{-}to$ (ln: 9, 10).

Algorithm 2: PV- and aging-aware mapping exploration

Input: A, N_r, N_s, $num\text{-}blk_A$, F_{req}, C, PV, $Health_{avg,blk}$, B, old mapping m, TDP, T_{safe}

Result: Alternate mapping m'

Parameters : l_{blk}, b_{lk}, $max\text{-}hop\text{-}count$

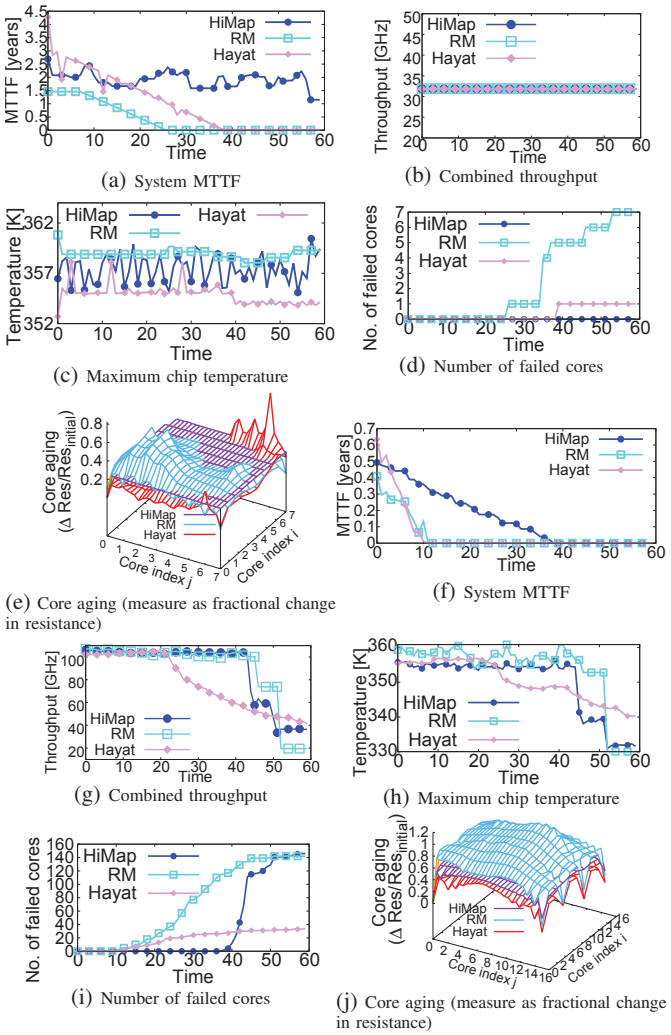
1: $m' \leftarrow \phi$;
2: Arrange B in decreasing order of $health_{avg,blk}$;
3: Arrange applications in decreasing order of avg. power as A';
4: **for** $j \leftarrow 1$ to $|A|$ **do**
5: Sort $B_{cluster,j}$ in decreasing order of number of running cores;
6: **for** each block b in $B_{cluster,j}$, starting from the first **do**
7: Sort $threads$ mapped to b in descending order of power;
8: Pick the fastest available block b', within max-hop-count limit blocks;
9: $cores\text{-}to \leftarrow$ cores cb';
10: Sort $cores\text{-}to$ in decreasing order of f_{max};
11: **if** Check if mapping $threads$ to $cores\text{-}to$ meets F_{req} **then**
12: map $threads$ to $cores\text{-}to$ leaving the rest sleeping, and update m';
13: **else**
14: SwapBlockWithHigherPowerApplication(F_{req},A_j,A);
15: **end if**
16: **while** !(m' meets TDP and T_{safe}) **do**
17: Try other mappings by remapping threads within the block;
18: **if** Do not have any unexplored combination and still not met TDP or T_{safe} **then**
19: Skip mapping A_j and log it;
20: **end if**
21: **end while**
22: **end for**
23: **end for**

While mapping $threads$ to $cores\text{-}to$, it checks performance constraint (F_{req}). If it is not met, it checks with an already mapped block of a higher power application, if it can satisfy the performance constraint. If so, it swaps workload between this block and b (using function $SwapBlockWithHigherPowerApplication$, ln 14). If no suitable mapping is found, it returns failure. Lastly, output mapping is checked if it meets TDP and T_{safe}. If any is not satisfied, it tries to find a mapping by exhaustively checking permutations within blocks such that all constraints are met. If unsuccessful, it logs failed mapping for the application.

Stage II ensures uniform aging of cores by mapping to healthier blocks and placing threads such that weaker cores sleep avoiding further stressing and healthier cores run higher power generating threads. For SPLASH-2, the application threads have nearly equal runtime. Hence, higher power generating threads lead to higher temperature and greater aging.

Mapping intervention overhead and scalability: During both the stages, there is no performance penalty owing mapping exploration since it is run on a separate core i.e. DRM core. During stage I, when the initial cluster-based mapping is found the scheduler maps threads as per the obtained mapping from the next hyperperiod onwards, for the rest of the epoch. For a number of block dimensions explored ($num\text{-}block\text{-}dim\text{-}options$), stage I has a runtime complexity of $O(num\text{-}block\text{-}dim\text{-}options * M * Num_{rounds} * Num_{steps})$, where, M is the number of applications, and Num_{rounds} and Num_{steps} are, respectively, the number of rounds and steps in SA. Thus, stage I is scalable w.r.t. number of cores and applications.

The runtime complexity of stage II is $O(M.k.logk)$, where M is the number of applications and k is the maximum number of threads among the applications. Thus, stage II is scalable w.r.t. number of applications and size of manycore system. As mentioned, during stage II Algo. 2 is run on DRM core to get the alternate mapping just before end of the epoch. The scheduler maps threads to cores as per the obtained mapping for the length of an entire epoch, starting with the next hyperperiod. The remapping is non-preemptive and there is no associated performance overhead. As the intervention is once an epoch, its power overhead is negligible.



(a) System MTTF (b) Combined throughput (c) Maximum chip temperature (d) Number of failed cores (e) Core aging (measure as fractional change in resistance) (f) System MTTF (g) Combined throughput (h) Maximum chip temperature (i) Number of failed cores (j) Core aging (measure as fractional change in resistance)

Fig. 5: Comparison of various metrics over time (measured in epochs) due to different mapping techniques for **64-core (a-e)** and **256-core (f-j)** systems

V. Experimental Setup and Results

Our experimental setup consists of the Snipersim [24] manycore simulator, interfaced with power simulator McPAT [25], thermal simulator Hotspot [23], and our aging and MTTF assessment. We have updated the leakage model in HotSpot with a PV-aware adaptation of temperature dependent leakage model from [26]. The technology node is 22 nm. Tile size is 0.7 mm $\times$ 0.8 mm with Nehalem core, L1 cache (256 kB), L2 cache (512 kB), nominal frequency of 1GHz. We conducted experiments for 64-core and 256-core systems with applications from SPLASH-2, considering T_{safe} of 90°C, and frequency to vary by 10% in 64-core system and 30% in 256-core system. We took epoch as 1 month and simulated for 5 years (60 epochs). For 64-core system, we took four 8-threaded applications. For 256-core system, we took three 8-threaded applications and six 16-threaded applications. We took R=2 and $max\text{-}hop\text{-}count$=2.

We compared HiMap with state-of-the-art aging-aware mapping approaches, namely Hayat [9] and RM [10]. For a fair comparison we compared lifetime reliability as well as performance. We performed HiMap's stage I with block dimensions: 2×1, 1×2, 2×2, 4×1 and 1×4. For the particular PV maps, we found the block dimension of 2×2 to be the most favorable for improving $MTTF_{sys}$, in both manycore systems.

As shown in Fig. 5f and 5a, for both the systems, HiMap achieves improved system MTTF in the long run compared to both Hayat and RM. Specifically, for 64-core HiMap improved system MTTF by 2 years at the end of 3.25 years, and for the 256-core system by 0.32 years at the end of 1 year. This is due to HiMap's noncontiguous block-based and aging-aware mapping which ensures uniform aging across the chip. Since RM's objective is to maximize frequency, it is relevant to compare the throughput, which is the sum of frequency of cores executing threads of all applications. As shown in Fig. 5g for 256-cores, for most of the time HiMap remained close to RM in terms of throughput and fared lower than RM for a short duration. However, at all times HiMap met the performance constraint. For 64-core system, all the three approaches give similar performance (Fig. 5b).

Fig. 5h and 5c compare maximum chip temperature. Hayat achieves lowest temperatures due to wider mapping spreads and higher thermal mitigation owing dark cores, as compared to HiMap. However, it results in early core failures as shown in Fig. 5i and 5d, since it aims at maximizing sum of the health of all the cores as opposed to maximizing $MTTF_{sys}$, as in HiMap. RM results in the highest temperatures due to contiguous mapping and lesser thermal mitigation.

Fig. 5j and 5e show the aging as fractional change in resistance ($\Delta Res/Res$) (assessed using Eq. 3) at end of 60 epochs. For 256-core case, it is evident from Fig. 5j that HiMap results in overall lesser aging than RM. The maximum fractional change in resistance were 1.3, 1.7, and 2.9, respectively for HiMap, Hayat, and RM. Thus, maximum aging caused by HiMap is lesser than Hayat. For 64-core case, Fig. 5e shows that both RM and Hayat lead to larger worst-case aging across the cores than HiMap. HiMap achieved a more uniform core aging. Fig. 5i and 5d compare the number of failed cores. For 256-core case (Fig. 5i) RM has the most failed number of cores almost throughout. Although Hayat has the least number of failed cores at the end of 60 epochs, it begins have failures earlier than HiMap. This is as per the expectation, since HiMap tries to save the weakest core. For 64-core case, HiMap led to no failures. RM led to more failures than Hayat. Thus, HiMap results in improved system MTTF and more uniform core aging in presence of PV, while meeting performance, TDP and thermal constraints.

VI. Conclusions

This paper presents HiMap, an efficient and scalable hierarchical mapping approach for improving lifetime reliability of dark silicon manycore systems while satisfying performance, power and temperature constraints. HiMap ensures uniform aging and defers core failure while containing inter-thread communication distances with aging- and PV-aware cluster-based mapping of threads to healthy cores keeping weaker cores as dark for thermal mitigation. Experimental results for 64- and 256-core systems validate HiMap's effectiveness and show significant improvement over state-of-the-art. HiMap exhibits that a hierarchical approach can enable efficient management of the complex space of mapping onto dark silicon manycore systems towards lifetime reliability enhancement.

VII. Acknowledgement

The coauthor Dr. Shafique's contributions in this work are supported in parts by the German Research Foundation (DFG) as part of the GetSURE project in the scope of SPP-1500 priority program "Dependable Embedded Systems".

References

[1] Intel® Xeon Phi™ Processor 7290F.
[2] Tilera and Maipu Introduce 512-Core Router.
[3] Brent Bohnenstiehl et al. Kilocore: A 32 nm 1000-processor array. In *Hot Chips*, 2016.
[4] Daniel Johnson et al. Rigel: A 1,024-core single-chip accelerator architecture. *IEEE Micro*, 31(4):30–41, 2011.
[5] Jörg Henkel et al. New trends in dark silicon. In *DAC*, page 119, 2015.
[6] Anup Das et al. Reliability-driven task mapping for lifetime extension of networks-on-chip based multiprocessor systems. In *DATE*, 2013.
[7] L.-T. Yeh and R. C. Chu. *Thermal Management of Microelectronic Equipment: Heat Transfer Theory, Analysis Methods, and Design Practices*. ASME Press, 2002.
[8] Keith A Bowman et al. Impact of die-to-die and within-die parameter fluctuations on the maximum clock frequency distribution for gigascale integration. *JSSC*, 37(2):183–190, 2002.
[9] Dennis Gnad et al. Hayat: Harnessing dark silicon and variability for aging deceleration and balancing. In *DAC*, pages 1–6, 2015.
[10] Mohammad-H. Haghbayan et al. A lifetime-aware runtime mapping approach for many-core systems in the dark silicon era. In *DATE*, 2016.
[11] Shengquan Wang et al. Thermal-aware lifetime reliability in multicore systems. In *ISQED*, pages 399–405, 2010.
[12] Lin Huang et al. On task allocation and scheduling for lifetime extension of platform-based mpsoc designs. *TPDS*, 22(12):2088–2099, 2011.
[13] Xiaohang Wang et al. Bubble budgeting: throughput optimization for dynamic workloads by exploiting dark cores in many core systems. *TC*, 2017.
[14] Ewerson Carvalho et al. Heuristics for dynamic task mapping in noc-based heterogeneous mpsocs. In *RSP*, pages 34–40, 2007.
[15] Taeyoung Kim et al. Learning-based dynamic reliability management for dark silicon processor considering em effects. In *DATE*, 2016.
[16] Florian Kriebel et al. ageopt-rmt: compiler-driven variation-aware aging optimization for redundant multithreading. In *DAC*, 2016.
[17] Semeen Rehman et al. dtune: Leveraging reliable code generation for adaptive dependability tuning under process variation and aging-induced effects. In *DAC*, pages 1–6, 2014.
[18] Mohammad Salehi et al. dsrelim: Power-constrained reliability management in dark-silicon many-core chips under process variations. In *CODES+ ISSS*, pages 75–82, 2015.
[19] Mohammad Salehi et al. Drvs: Power-efficient reliability management through dynamic redundancy and voltage scaling under variations. In *ISLPED*, pages 225–230, 2015.
[20] Bharathwaj Raghunathan et al. Cherry-picking: exploiting process variations in dark-silicon homogeneous chip multi-processors. In *DATE*, 2013.
[21] Vikas Mehrotra et al. A methodology for modeling the effects of systematic within-die interconnect and device variation on circuit performance. In *DAC*, pages 172–175, 2000.
[22] Xin Huang et al. Physics-based electromigration assessment for power grid networks. In *DAC*, pages 1–6, 2014.
[23] Wei Huang et al. Hotspot: A compact thermal modeling methodology for early-stage vlsi design. *TVLSI*, 14(5):501–513, 2006.
[24] Trevor E Carlson et al. Sniper: Scalable and accurate parallel multi-core simulation. *Intel European Exascale Labs*, page 22.
[25] Sheng Li et al. Mcpat: an integrated power, area, and timing modeling framework for multicore and manycore architectures. In *MICRO*, 2009.
[26] Vivek Chaturvedi et al. On the fundamentals of leakage aware real-time dvs scheduling for peak temperature minimization. *JSA*, 58(10), 2012.