While there are many solutions to the shortcomings of multicore processors, a solution that addresses one constraint often comes at the cost of another. In the past, resources such as memory, bandwidth, and speed were the limiting factors in computing performance. While these factors are still relevant today, energy efficiency and thermal cooling are emerging as the driving forces behind multicore processor design. A processor made of a complex core and many small cores may use the big core for faster sequential computation and the smaller cores for faster parallel execution. Application specific stresses can be biased to the particular type of core that can best handle the computation. The big(complex) core is the busiest, it is almost always turned on and usually must execute both serial and parallel phases of workload. Failure of big core immediately leads to failure of entire processor. If the processor has more than one big core than any one of them can be used for serial execution while the remainder are power gated until they are needed. Under a given area constraint, increasing the number of big cores on a die reduces the capacity for small cores. For power saving applications, it is best to use the big core on predominantly sequential executions. Energy efficiency can be effectively measured with metrics such as performance per joule or watt. While the heterogeneous processors is efficient in it’s ability to bias the workload it does require convoluted scheduling methods to manage it’s resources. There are many implementations of the heterogeneous processor and in order to develop solutions for the multicore processor we must consider each of the different configurations.

For the following situations all of the equations are composed of the following variables:

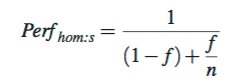
Variables:

* f: The fraction of instructions improved.
* n: The number of small cores in the processor.
* b: The number of big cores in the processor.
* r: How many times larger the area of a big core is compared to a small core.
* s: How many times better the performance of a big core is relative to a small core.
* P: How many times more power a large core consumes relative to a small core.

For each of the different instances of multi core processor we must consider the reliability and lifetime, in addition to the performance and energy efficiency, in order to achieve a design that is a holistic improvement on current technology. For the purpose of simplicity, we will focus on scaling factors and generally simplify other conditions. It is important to note that when as the fraction of accelerated instructions increases, overall speedup is dominated by parallel threads. For the majority of our analysis we will use performance per joule as our metric of energy efficiency. Lifetime reliability of a system is heavily correlated with temperature stability, with Hot Carrier Injection and Bias Temperature Instability accounting for the two most prominent critical-failure-mechanisms. The overall failure rate is calculated as the sum of all failure rates and we assume that failure rates are proportional to die area whilst stress conditions are equivalent. Based on the assumption of failure rates in proportion to die area, the failure of a big core is ‘r’ times greater than that of a small core. Not surprisingly, failure rates in serial phases are lower than in parallel executions and imbalanced failure rate distribution amongst the different core types limits overall lifetime. In cases where the (b/n) ratio is sufficiently large (ex. One complex core, many simple cores),the stresses are biased towards the complex core(s). When ‘n’ is suitably large, increasing ‘b’ alleviates stress on the big cores and improves the processor’s lifetime reliability. That being said, increasing (b/n) reduces maximum parallel throughput and the resulting extended execution time diminishes energy efficiency and reliability.

The simplest case is that of the Homogeneous Processor of Simple Cores. As it’s name implies, it is comprised of only of ‘n’ number of simple cores. The maximum speedup of said configuration is given by Eq.(1). Energy scaling derived using Amdahl’s law is Ehom:s=1 , assuming power consumption is normalized to one. Power scaling is the same as the performance model, (Whom:s)=(Perfhom:s). The power consumed in serial executions is equivalent to single- core power. In this configuration, in the case that parallel executions are sufficiently short, sequential throughput may limit the overall speedup. In the lifetime reliability analysis performed by Song, Mukhopadhyay, and Yalamanchili [3] the control case is based on the assumption of 100% parallel execution. During parallel executions, the performance improvement (f/n) is offset by the correspondingly greater total failure rate.

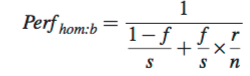
Eq.1)



Equation 1 is an optimistic estimation of performance because it does not account for thread parallelization or the overhead of data transfer.

While the above architecture represents the most fundamental configuration, the Homogeneous Processor of Complex Cores is similar in that it only possesses one core type. It is composed of ‘b’ number of big(complex) cores that have ‘s’ times faster performance compared to the simple core. In the same die area, there can only be n/r number of big cores, and each core is capable of executing serial threads independently. The performance speed up of this arrangement is given by Eq.(2). This speedup is achieved by speeding up serial executions by a factor of ‘s’ and parallelizing the ‘f’ fraction of throughput by n/r number of big cores. It is assumed that a big core consumes ‘p’ times the amount of power of a small core (normalized to one) and the total energy dissipation of the processor is given by Eq.(3). This calculation is simplified by the assumption that unused cores are power-gated. If the processor only requires one big core at a time to execute serial instructions, than the processor consumes ‘p’ power while executing said threads. If all cores in the processor are required for a task, than p(n/r) amount of power is dissipated by the processor. Complex cores have lower power density and therefore a lower failure rate per unit area, however, they posses a higher overall failure rate per core. While there are some perceived benefits of such a setup, the modest performance increase in parallel execution is offset by the failure rate of the big cores.

2)



In equation 2, (1-f) is the serial execution fraction of the workload, and is speed up by a factor of s, the ‘f’ or parallelized fraction is computed by (n/r) number of big cores.

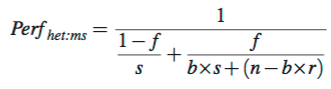
3)

Macintosh HD:Users:bryanguner:Desktop:Screen Shot 2016-11-13 at 3.35.20 PM.png

During serial execution, the processor consumes p amount of power because only a single core is in use, during parallel execution it consumes p(n/r) assuming all cores are active. The big core sped up is given by (b\*s) and the small by (n-b\*r).

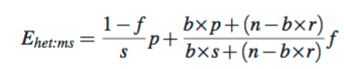
The progression of homogeneous architectures and their limitations, begs the question of weather or not a hybrid layout might leverage the best attributes of each core type. The most intuitive of such combinations is the Heterogeneous Processor with Maximum Scheduling. When performing only one application at a time, one complex core is adequate to carry out serial threads. Depending on the application of the processor, there may be a need to include numerous big cores at the expense of small ones to execute concurrent serial threads for maximum overall throughput. This case makes the assumptions of maximum scheduling (cores are fully utilized at execution time), which is unrealistic in the context of today’s hardware-software interfaces. The performance speedup is given by Eq.(4), during single-thread executions, the serial fraction of the workload (1-f) is accelerated by a complex core that is ‘s’ times faster than a small core. Both types of cores, depending on resource availability, parallelize the ‘f’ fraction of the workload. The power consumption of the Heterogeneous Processor is given by Eq.(5), during sequential operation the processor consumes ‘p’ power, while unused cores are assumed to be power-gated. In parallel phases, total power is calculated as the sum of ‘big core consumption’ and ‘small core consumption’ (b\*p)+(n-b\*r). This formulation is based on the normalization of small core consumption to one. The total power consumption is given by Eq.(5). In the analysis of lifetime reliability performed by Song, Mukhopadhyay, and Yalamanchili [3], it was found that when the number of small cores is relatively small, parallel executions lead to relatively long latency times. In their fault analysis, that the probability that a big core caused the processor to fail was simply calculated as the big core failure rate over the total failure rate.

4)



Operating under the assumption of maximum scheduling for the heterogeneous processor where the small cores occupy (n-b\*r) area.

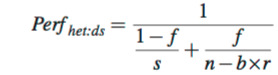
5)



The ‘f’ fraction of the workload is parallelized between both big cores with a speedup of (b\*s) and a small core speed up of (n-b\*r).

The Homogeneous Processor with Dynamic Scheduling is composed of both core types, however, in contrast with the Max Scheduling configuration, separates the use of distinct core types to the tasks that best suit them (big core for serial, small cores for parallel). This approach excels in power-constrained applications; running parallel threads exclusively on small cores implicitly solves scheduling issues created by the discrepancy in performance of the respective core types. The performance speedup is given by Eq.(6) and the energy consumption by Eq.(7). Power-gating unused cores improves scheduling protocol, delivering improved in reliability at the expense of performance degradation. The performance of this architecture is improved by an increase in the ratio of big cores to small cores, however, it’s lifetime is threatened by the bottleneck of the whole processor’s dependency on the big cores operation.

6)



The (1-f) fraction is accelerated by a big core that is ‘s’ times faster than it’s simple counterpart, ‘f’ is parallelized by (n-b\*r) small cores.

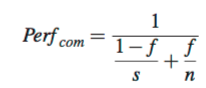
7)

Macintosh HD:Users:bryanguner:Desktop:Screen Shot 2016-11-13 at 3.36.12 PM.png

In parallel executions, threads are run exclusively on small cores and consume (n-b\*r) amount of power.

The Composed Processor of Simple Cores is comprised of ‘n’ number of small cores, where a subset of cores is dynamically combined to speed up serial execution. In this scenario, a set of ‘r’ small cores has the same performance as a complex core ‘s’. The aforementioned group consumes ‘r’ amount of power, which can potentially be greater than the power usage of a complex core ‘p’. The performance speedup of said configuration is given by Eq.(8) and the power consumption by Eq.(9). This architecture represents an ideal case where both serial and parallel threads are accelerated. This ideal design provides the greatest aggregate performance speedup over the whole spectrum of parameters considered in this analysis. The only difference between the Composed Processor and the Homogeneous Processor of Small Cores is that the Composed Processor is capable of grouping together small cores to emulate the performance of a large core. The reliability impact of this implementation is also ideal in the fact that a different group of simple processors can be used to run serial threads for each successive application so that no single small core is overworked.

8)



The Composed Processor represents and ideal case where both serial and parallel phases of workload are accelerated in which serial executions are accelerated by a factor of ‘s’ and the ‘f’ fraction of execution is parallelized by ‘n’ small cores.

9)

Macintosh HD:Users:bryanguner:Desktop:Screen Shot 2016-11-13 at 3.37.03 PM.png

The processor dissipates r amount of power during serial execution and n amount of power during parallel execution.