Overview
The goal of the Efficient Supercomputing Project is to significantly reduce the amount of energy consumed executing scientific code while providing programmers an API that allows for productive algorithm implementation. We do this by exposing locality to the programmer, minimize unnecessary network traffic, and reduce cache contention and meta-data overhead.

Goals
- Design a high-performance efficient architecture, that provides parallelism with minimal overhead over 100s-1000s of cores
- Enable faster, more efficient code through software configuration of cache hierarchies and active messages
- Provide programming system, allowing developers to productively implement algorithms that optimally use hardware

Motivation
TCO of a Data Center 55% due to power requirements

Application Energy Breakdown
Splash 2 Radix Sort Energy Consumption

Cache Hierarchy
The Kapok project is focused on reducing the amount of energy consumed in the data supply on chip. Optimizing the coherent cache hierarchy is an important means by which we can do this. Novel structures and programming interfaces must be developed to improve coherency scalability.

Exposure to Data Locality
- Data movement is 45% of the energy in a many-core system
- Cache coheren shared memory obfuscates this energy
- Improve energy-efficiency and performance in many-core processors
- Our research targets all types of energy consumption shown

Memory & Communication
Since communication and memory energy does not scale with computational energy, data movement will become a larger problem as devices scale. Active messaging, block transfers, and fast barriers are examples of efficient communication mechanisms provided by Kapok.

Remote Messaging & Communication
- The key to reducing the amount of energy consumed in cache coherency protocols is simple: do not miss
- Access highly contended variables/locks at their home node via active messages instead of invalidating loads and stores
- Configurable cache hierarchy allows programmers to take advantage of different forms of sharing

Active Messaging
Energy
- Software configuration of cache hierarchies improves performance
- Allow the user to configure cache domains
- Provide APIs to allow for pinning data to local storage
- Convert portions of SRAM to non-coherent, locally addressed scratchpad memory

Exposed Data Locality
- Allow programmers to control data location
- Less energy spent locating and moving the data on loads and stores
- Managed either via hardware or software

Profiling
- Memory address contention, use remote writes
- Profiling information used to suggest communication mechanisms
- Design compilers to automatically select communication mechanisms for programs