Motivation
Moore’s Law
The number of transistors on integrated circuits doubles approximately every two years. Continues to hold until today

Dennard’s Law
Power density of the transistor decreases with size. Came to an end in this decade

Implication on Hardware
- A computer with a single core cannot run faster.
- Chip manufacturers add multiple cores to enable the computer to run faster.
- Birth of heterogeneous processors.

Problem
- Existing programming languages used to develop computer applications are not able to tap into the full potential of the modern hardware.
- Computer software cannot run faster.
- Advancements realized due to faster computers will HALT!

Goal
- Enable existing software to run faster on the modern hardware.
- Make it feasible to develop new potentially life saving applications on faster computers.

Existing Approaches
- Use low level languages like OpenCL or CUDA.
- Unproductive, Unportable, Hard to achieve Performance.

Our Approach
- Develop high level languages.
- Habanero-C: Enhance productivity, portability and performance by extending familiar C language.

Habanero-C (HC) constructs
- finish{ Body } 
  - Ensures all tasks spawned inside the body are completed
- async copyin(args1) copyout(args2) at(device){ Body } 
  - Asynchronously copy data between the device and the host.
- forasync point (args) size (args) seq (args)[scratchpad(args)] 
  - Asynchronously execute the body of the loop on the device.

Compilation Framework

Empirical Evaluation

Portability
- Habanero-C code can be executed on various machines.
- Including latest computers from AMD, NVIDIA and Intel

Productivity
- Minimal (~2%) code rewrite required to port existing applications.
- Easy to write new applications in HC.
- e.g. Lattice Boltzmann Method (LBM) Simulation took 2 Weeks of hand coding low level OpenCL as opposed to 2 Hours of HC programming.

Performance
- Performance of HC is similar to hand-coded OpenCL

Summary
- High level languages necessary for portability, productivity and performance.
- HC provides simple extensions to the C language to program applications on newer hardware.
- Performance of HC is comparable to hand-coded lower-level implementations.
- In future, support for execution on multiple devices.

- Speedup obtained on a NVIDIA M2050 graphics card
- LBM: 300 x 300 x 60
- Input Size: 2MM: 2048 x 2048 x 2048
- Hand Coded OpenCL
- Habanero-C

Moore's Law: The number of transistors on integrated circuits doubles approximately every two years. Continues to hold until today.

Dennard's Law: Power density of the transistor decreases with size. Came to an end in this decade.

Implication on Hardware:
- A computer with a single core cannot run faster.
- Chip manufacturers add multiple cores to enable the computer to run faster.
- Birth of heterogeneous processors.

Problem:
- Existing programming languages used to develop computer applications are not able to tap into the full potential of the modern hardware.
- Computer software cannot run faster.
- Advancements realized due to faster computers will HALT!

Goal:
- Enable existing software to run faster on the modern hardware.
- Make it feasible to develop new potentially life saving applications on faster computers.

Existing Approaches:
- Use low level languages like OpenCL or CUDA.
- Unproductive, Unportable, Hard to achieve Performance.

Our Approach:
- Develop high level languages.
- Habanero-C: Enhance productivity, portability and performance by extending familiar C language.

Habanero-C (HC) constructs:
- finish{ Body } 
  - Ensures all tasks spawned inside the body are completed
- async copyin(args1) copyout(args2) at(device){ Body } 
  - Asynchronously copy data between the device and the host.
- forasync point (args) size (args) seq (args)[scratchpad(args)] 
  - Asynchronously execute the body of the loop on the device.

Compilation Framework:

Empirical Evaluation:

Portability:
- Habanero-C code can be executed on various machines.
- Including latest computers from AMD, NVIDIA and Intel

Productivity:
- Minimal (~2%) code rewrite required to port existing applications.
- Easy to write new applications in HC.
- e.g. Lattice Boltzmann Method (LBM) Simulation took 2 Weeks of hand coding low level OpenCL as opposed to 2 Hours of HC programming.

Performance:
- Performance of HC is similar to hand-coded OpenCL

Summary:
- High level languages necessary for portability, productivity and performance.
- HC provides simple extensions to the C language to program applications on newer hardware.
- Performance of HC is comparable to hand-coded lower-level implementations.
- In future, support for execution on multiple devices.

- Speedup obtained on a NVIDIA M2050 graphics card
- LBM: 300 x 300 x 60
- Input Size: 2MM: 2048 x 2048 x 2048
- Hand Coded OpenCL
- Habanero-C