Outline

• Mobile computing market -- what do users care about?
• Mobile computing landscape -- how does the platform look like?
• Applications and developers -- who is writing code for mobile?
• Browser wars revisited -- why does it matter?
• Parallel heterogeneous programming made easier using MARE
MOBILE DEVICE USAGE IS EXPLODING

By 2014 there will be more mobile-connected devices on Earth than people.

By 2014, there will be 7.3 billion mobile connected devices in the world.

More than 7.15 billion world population (est.)
Size of mobile market roughly doubling every two years

Source: IDC. HPC forecast Intersect360
What do users care about?

The phase in which buyers decide whether or not they need a new phone, and what improvements they will seek on what they already have.

**Primary Features Buyers Considered**

- Camera
- Battery Life
- Ease of Use
- Speed
- Email / Text
- Color / Style
- Plans
What about next morning?

EXPERIENCE

The phase in which buyers experience using and living with the phone they selected, discovering how the product compares to the expectations prior to purchase.

Reactions After Purchase

<table>
<thead>
<tr>
<th>Battery Life</th>
<th>No. of Negative</th>
<th>No. of Positive</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>10</td>
<td>4</td>
</tr>
</tbody>
</table>

McKinsey’s Consumer and Shopping Insights
The Internet of Everything

<table>
<thead>
<tr>
<th>Connectivity</th>
<th>Context</th>
<th>Control</th>
</tr>
</thead>
<tbody>
<tr>
<td>An estimated 25B devices connected by 2020, talking to each other, making choices, and taking decisions</td>
<td>Use context awareness to filter the data that reaches every person: time, location, taste</td>
<td>Allow users to control and customize experience through new, mobile centric interfaces</td>
</tr>
</tbody>
</table>

Enabling whole new classes of applications, where **power** is even more critical.
What’s in a smartphone?
Anatomy of a Mobile SoC: Qualcomm Snapdragon 800

- **Krait 400 CPU**
  - Features 28nm HPm process technology
  - Superior 2GHz+ performance

- **Adreno 330**
  - For advanced graphics

- **Hexagon QDSP6**
  - For ultra low power applications and custom programmability

- **Integrated 802.11ac, 3.0 USB, and BT 4.0**
  - Offers broad array of high speed connectivity

- **Multimedia**
  - Audio, Video and Gestures

- **Camera**
  - 21MP with dual ISP

- **Connectivity**
  - 4G LTE, WiFi, USB, BT, and FM

- **Support for up to 2560x2048 display**
- Miracast 1080p HD support
- IZat GNSS with support for three GPS constellations

Source: [www.qualcomm.com/snapdragonprocessors/800](http://www.qualcomm.com/snapdragonprocessors/800)
Mobile Constraints

Energy

Thermal
## Mobile Applications

<table>
<thead>
<tr>
<th>Connectivity</th>
<th>Gaming</th>
</tr>
</thead>
<tbody>
<tr>
<td>Social networking, sharing: Media rich, data heavy</td>
<td>On device and multiplayer games with realistic graphics</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Browsing</th>
<th>Imaging</th>
</tr>
</thead>
<tbody>
<tr>
<td>Information gateway</td>
<td>Computational photography, search and processing of images and video</td>
</tr>
</tbody>
</table>
Mobile Software Development

Developers
- Think in terms of services and “features”
- Portable apps, performance not the main concern
- Use JavaScript frameworks

Gap!

High Power Efficiency

Low Power Efficiency

Qualcomm Research
How can we easily write parallel applications that use all available cores in a battery-powered device?
Our Vision of a Mobile Software Stack

Portability: Zoomm Web App Engine
- Use of concurrency to optimize execution of Web Apps
- Hardware exploitation through the browser

Programmability: MARE
- Parallel, heterogeneous programming made easier
- Power and performance optimizations

Performance: Domain Specific libraries
- Exploit domain knowledge to provide composable libraries for all programmers
- Hide hardware complexity
Browsers: why do we care?
Web Browser Usage

Information access

89% Email
85% Browsing

Web Applications

Source: Illuminas, 2013
Insight: There is no one dominant component – Parallelization must address the entire browser structure!

ARM Cortex A9 Execution Time

Average over the top 30 Alexa sites (May 2010), WebKit browser on a 400MHz Cortex A9, Linux
Zoomm: Pervasive Concurrency

Cascaval et al.: Zoomm: a parallel web browser engine for multicore mobile devices, PPoPP 2013
Zoomm: Page Load Performance

- **WebKit**
  - CNN, BBC, Yahoo, Guardian
  - NYT, Facebook, Engadget, QQ

- **Zoomm**
  - Same categories as WebKit

HTC Jetstream, MSM8660, 2-core, 1.5GHz, Aug 2012
Qualcomm MARE

Multicore Asynchronous Runtime Environment
The Smartphone of 2010

- All applications run on a single-core Qualcomm Snapdragon processor at 1GHz
The Smartphone of 2013

- Multiple applications can simultaneously run on a quad-core Snapdragon at 2+GHz
What is Qualcomm MARE?

- MARE is a **programming model** and a **runtime system** that provides simple yet powerful abstractions for parallel, power-efficient software
  - Simple C++ API allows developers to express concurrency
  - User-level library that runs on any Android device, and on Linux, Mac OS X, and Windows platforms

- The goal of MARE is to reduce the effort required to write apps that fully utilize heterogeneous SoCs
MARE Workflow

Focus on your application and not on the hardware

- Understand algorithms
- Partition algorithms into independent units of work
- Setup task dependencies
- Link with MARE Runtime
Qualcomm MARE API Concepts

• **Tasks** are units of work that can be asynchronously executed
• **Groups** are sets of tasks that can be canceled or waited on
### Advantages of using Qualcomm MARE

<table>
<thead>
<tr>
<th>Simple</th>
<th>Productive</th>
<th>Efficient</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tasks are a natural way to express parallelism</td>
<td>Focus on application logic, not on thread management</td>
<td>Task dependences allow the MARE runtime to perform more intelligent scheduling decisions</td>
</tr>
</tbody>
</table>
Hello World!

```cpp
#include <stdio.h>
#include <mare/mare.h>

int main() {
    mare::runtime::init(); // Initialize MARE

    auto hello = mare::create_task([] { printf("Hello "); }); // Create task
    auto world = mare::create_task([] { printf("World!"); }); // Create task

    hello >> world; // Set dependency

    mare::launch(hello); // Launch hello task
    mare::launch(world); // Launch world task

    mare::wait_for(world); // Wait to complete

    runtime::shutdown(); // Shutdown MARE
    return 0;
}
```
Parallel Programming Patterns in MARE

• MARE offers several commonly used parallel patterns:
  – `mare::pfor_each` processes the elements of a collection
  – `mare::pscan` performs an in-place parallel prefix operation for the elements of a collection.
  – `mare::transform` performs a map operation on the elements of a collection

• Programmers can also create their own patterns using the MARE API

• Parallel data structures: concurrent queues, hash tables, etc.
Bullet Physics Parallelization using MARE

• Three major components: about 75% of total execution time

• Constraint Solver
  – Finds non-interacting groups of objects and solves as independent tasks
  – Coarse-grained task-parallelization

• Rendering
  – Very coarse-grained task-parallelization: offload to a separate thread to allow for continuous simulation

• Narrow Phase Collision Detection
  – Fine-grained data-parallelization

• About 2x FPS improvement on typical mobile devices
  – Dramatically improves the smoothness of rendering and overall user experience
Bullet Physics Parallelization

![Graph showing Frames Per Second vs. Frame Number with two lines for MARE and Serial]
How can You try MARE?

- Stay tuned!
- Planning preview availability on Qualcomm Developer Network
  - Fall 2013 Beta release for the multicore version
- We Want Your Feedback!
  - What scenarios will you use it for?
  - Usability and features of the library
Acknowledgments

• Manticore team
  – Pablo Montesinos Ortego, Michael Weber, Wayne Piekarski, Behnam Robatmili, Dario Suarez-Gracia, Vrajesh Bhavsar, Jimi Xenidis, Han Zhao, Kishore Puskuri

• Interns
  – Madhukar Kedlaya, Christian Delozier, Freark Van Der Berg, Christoph Kershbaumer

• Former members
  – Mehrdad Reshadi, Seth Fowler, Alex Shye, Dilma DaSilva

• Qualcomm
  – Nayeem Islam, Mark Bapst, Charles Bergan, Matt Grob, Ben Gaster

• MulticoreWare
  – Wen-Mei Hwu, Mark Allender, Kevin Wu
## Challenges

<table>
<thead>
<tr>
<th>Power and performance</th>
<th>Programmability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Custom hardware for power efficiency. Heterogeneity is the game.</td>
<td>Tools and languages to enable programmers to understand and express concurrency and application semantics</td>
</tr>
<tr>
<td>Expertly designed and optimized frameworks that hide hardware complexity</td>
<td>Composable building blocks</td>
</tr>
</tbody>
</table>

Mobile has exciting applications and a huge impact
Legal Disclaimers

All opinions expressed in this presentation are mine and may not necessarily reflect those of my employer.

Qualcomm and Snapdragon are trademarks of Qualcomm Incorporated, registered in the United States and other countries. All Qualcomm Incorporated trademarks are used with permission.

Other products and brand names may be trademarks or registered trademarks of their respective owners.