Rechnerarchitektur (RA)

Sommersemester 2016

Architecture-Aware Optimizations
- Hardware-Software co-Optimizations-

Jian-Jia Chen
Informatik 12
Jian-jia.chen@tu-..
Tel.: 0231 755 6078

2016/07/19
Outline

Heterogeneous System Architecture (HSA)
- Integrated CPU/GPU platforms
- Recent movement in chip designs

Architecture-aware software designs
- Energy-efficiency issues
- Darkroom
- Halide

Multicore revolutions
- Impact on the “safety-critical” industry sector
What is Heterogeneous Computing?

Use processor cores with various type/computing power to achieve better performance/power efficiency

Advantage of Heterogeneous Computing

CPU is ideal for scalar processing
- Out of order x86 cores with low latency memory access
- Optimized for sequential and branching algorithms
- Runs existing applications very well

GPU is ideal for parallel processing
- GPU shaders optimized for throughput computing
- Ready for emerging workloads
- Media processing, simulation, natural UI, etc.

Serial/Task-parallel workloads $\rightarrow$ CPU
Graphics/Data-parallel workloads $\rightarrow$ GPU

Heterogeneous Computing $\rightarrow$ Fusion, Norm Rubin, SAAHPC 2010
CPU/GPU Integration: CPU’s Advancement Meets GPU’s

Microprocessor Advancement
- Single-Thread Era
- Multi-Core Era
- Heterogeneous Systems Era

High Performance Task Parallel Execution

Heterogeneous Computing
- System-Level Programmable
- Power-efficient Data Parallel Execution
- Vertex/ Pixel Shader
- Graphics Driver-based programs

Throughput Performance

Programmability
- Unacceptable: Experts Only
- Mainstream: Mainstream

CPU/GPU Integration

GPU Advancement
Evolution of Heterogeneous Computing

Dedicated GPU
- GPU kernel is launched through the device driver
- Separate CPU/GPU address space
- Separate system/GPU memory
- Data copy between CPU/GPU via PCIe
Evolution of Heterogeneous Computing

Integrated GPU architecture

- GPU kernel is launched through the device driver
- Separate CPU/GPU address space
- Separate system/GPU memory
- Data copy between CPU/GPU *via memory bus*

- GPU kernel is launched through the device driver
- Separate CPU/GPU address space
- Separate system/GPU memory
- Data copy between CPU/GPU *via memory bus*
Evolution of Heterogeneous Computing

Integrated CPU/GPU architecture

- GPU kernel is launched through the device driver
- Unified CPU/GPU address space (managed by OS)
- Unified system/GPU memory
- No data copy - data can be retrieved by pointer passing
Utopia World of Heterogeneous Computing

Processors are architected to operate cooperatively
- Tasks in an application are executed on different types of core
- Unified coherent memory enables data sharing across all processors

Designed to enable the applications to run on different processors at different time
- Capability to translate from high-level language to target binary at run-time
- User-level task dispatch
- Decision making module

![Diagram of Utopia World of Heterogeneous Computing](image)
HSA Foundation

Founded in June 2012
Developing a new platform for heterogeneous systems
www.hsafoundation.com
Specifications under development in working groups to define the platform
Membership consists of 43 companies and 16 universities
Adding 1-2 new members each month
Diverse Partners Driving Future of Heterogeneous Computing

Founders

Promoters

Supporters

Contributors

© j. chen, informatik 12, 2016
HSA (Heterogeneous System Architecture) Hardware-Software Stack

Application

OpenCL

Task 1

Task 2

Task 3

OpenCL Compiler

HSA Intermediate Language (HSAIL)

Task 1

Task 2

Task 3

HSA Run-time

Agent Scheduler

CPU Finalizer

GPU Finalizer

Native machine codes

Core 1

Core 2

CU 1

CU 2

AC 1

AC 2

Shared virtual address space

User-level SW

HW
Intel Haswell

Haswell Processor Family Overview (Traditional)

22nm Process
Socket: G3 (947 pin) MB
H3 (1150 pin) DT
Fully Integrated VR
Power Aware Interrupt Routing for power / performance
Intel® Hyper-Threading Technology
Intel® AVX 2.0 extensions and AES-NI Instructions Improvements
Intel® Turbo Boost Technology

System Agent
Display
Core
Core
Core
Core
Core
LLC
LLC
LLC
Integrated Graphics

2ch DDR3L

Processor Graphics
3D Gfx/Media Arch/Perf.
Support for new APIs
3 Digital Displays
DP 1.2/HDMI 1.4/eDP

Desktop/All-In-One
DDR3/DDR3L-1600 2D/Ch
non-ECC UDIMM/SODIMM

Mobile
DDR3L (only)
POR for 1600 2D/Ch 6L rPGA
non-ECC SODIMM

DDR Power Gating
for lower idle power (MB)

PCIe 3.0 x16/2x8
3 controllers

Power Optimizer
(CPPM) Support (MB)

PCIe I/O

Display Ports

Last Level Cache (LLC) shared between CPU and Integrated Graphics
Hotham 1.0 Support
Overclocking Improvements

Haswell Offers Better Power & Performance with Improved I/O
“KAVERI” FEATURING UP TO 12 COMPUTE CORES (4 CPU+ 8 GPU)

CPU COMPUTE CORES

Up to four new multi-threaded AMD “Steamroller” CPU CORES

GPU COMPUTE CORES

Up to eight GCN GPU CORES powering parallel compute and next-gen gaming
NVIDIA Tegra K1

Kepler GPU (192 CUDA Cores)
Open GL 4.4, OpenGL ES3.0, DX11, CUDA 6

Quad Core Cortex A15 “r3"
With 5th Battery-Saver Core; 2MB L2 cache

Dual High Performance ISP
1.2 Gigapixel throughput, 100MP sensor

Lower Power
28hPM, Battery Saver Core

4K panel, 4K HDMI
DSI, eDP, LVDS, High Speed HDMI 1.4a
Qualcomm Snapdragon

- **Location**: GPS, GLONASS, Beidou, Galileo Satellites
- **Adreno 430 GPU**: OpenGL ES 2.0/3.1, OpenCL 1.2 Full, Content Security
- **Display Processing**: 4K, Miracast, picture enhancement
- **Modem**: 4th gen CAT 6 LTE, Up to 3x20MHz CA
- **Cortex-A57 & Cortex-A53 CPUs**
- **Memory**: LPDDR4
- **Hexagon DSP**: Ultra Low Power
- **Sensor Engine**
- **USB 3.0**
- **Multimedia Processing**: 4K Encode/Decode, Snapdragon Voice Activation, Gestures, Studio Access Security
- **Dual ISPs**: Camera, Up to 55MP, 1.2Gpixels/s bw, Camera SW
Outline

Heterogeneous System Architecture (HSA)
- Integrated CPU/GPU platforms
- Recent movement in chip designs

Architecture-aware software designs
- Energy-efficiency issues
- Darkroom
- Halide

Multicore revolutions
- Impact on the “safety-critical” industry sector
Energy Efficiency of different target platforms

© Hugo De Man, IMEC, Philips, 2007
Signal Processing ASICs

Markovic, EE292 Class, Stanford, 2013

Energy Efficiency (MOPS/mW)

- CPUs
- CPUs+GPUs
- GP DSPs
- Dedicated

~1000x
How about Memory?
Processor Energy with Corrected Cache Sizes

© Horowitz, DAC 2016
Processor Energy Breakdown

- 8 cores
- L1/reg/TLB
- L2
- L3

© Horowitz, DAC 2016
Data Center Energy Specs

© Malladi, ISCA 2012
What Is Going On Here?

Markovic, EE292 Class, Stanford, 2013

The graph shows the energy efficiency (MOPS/mW) over time. The x-axis represents time, and the y-axis represents energy efficiency. The graph compares different types of processors:

- CPUs
- CPUs+GPUs
- GP DSPs
- Dedicated

The graph indicates a significant improvement, with an approximate 1000x increase in energy efficiency compared to CPUs.
Energy Consumption (Approximate, 45nm)

<table>
<thead>
<tr>
<th>Integer</th>
<th></th>
<th>FP</th>
<th></th>
<th>Memory</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td></td>
<td>FAdd</td>
<td></td>
<td>Cache</td>
<td>(64bit)</td>
</tr>
<tr>
<td>8 bit</td>
<td>0.03pJ</td>
<td>16 bit</td>
<td>0.4pJ</td>
<td>8KB</td>
<td>10pJ</td>
</tr>
<tr>
<td>32 bit</td>
<td>0.1pJ</td>
<td>32 bit</td>
<td>0.9pJ</td>
<td>32KB</td>
<td>20pJ</td>
</tr>
<tr>
<td>Mult</td>
<td></td>
<td>FMult</td>
<td></td>
<td>1MB</td>
<td>100pJ</td>
</tr>
<tr>
<td>8 bit</td>
<td>0.2pJ</td>
<td>16 bit</td>
<td>1pJ</td>
<td>DRAM</td>
<td>1.3-2.6nJ</td>
</tr>
<tr>
<td>32 bit</td>
<td>3 pJ</td>
<td>32 bit</td>
<td>4pJ</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Instruction Energy Breakdown

- 25pJ I-Cache Access
- 6pJ Register File Access
- 70pJ Add

© Horowitz, DAC 2016
It’s more about the algorithm than the hardware
The efficiency cannot be achieved unless the algorithm is right!!

(a) Algorithms

GPU Alg.
Locality, Locality, and Locality!!!

© Hegarty et al., SIGGraph 2014
Darkroom (Stanford/MIT)

```
bx = im(x,y)
   (I(x-1,y) + I(x,y) + I(x+1,y))/3
end
by = im(x,y)
   (bx(x,y-1) + bx(x,y) + bx(x,y+1))/3
end
sharpened = im(x,y)
   I(x,y) + 0.1*(I(x,y) - by(x,y))
end
```

© Hegarty et al., SIGGraph 2014
Locality versus Parallelism

Halide Programming Language:
- http://halide-lang.org/

Performance needs a lot of tradeoffs
- Locality
- Parallelism
- Redundant recomputation
Outline

Heterogeneous System Architecture (HSA)
- Integrated CPU/GPU platforms
- Recent movement in chip designs

Architecture-aware software designs
- Energy-efficiency issues
- Darkroom
- Halide

Multicore revolutions
- Impact on the “safety-critical” industry sector
Automotive Software

Task

• Activation Pattern:
  • Periodic: 1 to 1000 ms
  • Angle synchronous
  • Sporadic

• Scheduled by the OS
  • Fixed Priorities
  • Preemptively or cooperatively

OSEK task states

© Hamann, Kramer, Ziegenbein, Lukasiewycz (Bosch), 2016
Assessment of Multi-Core Worst-Case Execution Behavior

Communication between runnables is realized with reading and writing of labels

<table>
<thead>
<tr>
<th>Communication</th>
<th>Share</th>
</tr>
</thead>
<tbody>
<tr>
<td>Forward</td>
<td>25 %</td>
</tr>
<tr>
<td>Backward</td>
<td>35 %</td>
</tr>
<tr>
<td>InterTask</td>
<td>40 %</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Size</th>
<th>Share</th>
</tr>
</thead>
<tbody>
<tr>
<td>Atomic (1-4 bytes)</td>
<td>97 %</td>
</tr>
<tr>
<td>Structs / Arrays</td>
<td>3 %</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Access type</th>
<th>Share</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read-only</td>
<td>40 %</td>
</tr>
<tr>
<td>Write-only</td>
<td>10 %</td>
</tr>
<tr>
<td>Read-Write</td>
<td>50 %</td>
</tr>
</tbody>
</table>

© Hamann, Kramer, Ziepenbein, Lukasiewycz (Bosch), 2016
Multi-Core Memory Access Models

Access time to data in different memories (local & global)

© Hamann, Kramer, Ziegenbein, Lukasiewycz (Bosch), 2016
An Industrial Challenge (FMTV 2016)

Precise analysis of worst-case end-to-end latencies

- mainly due to different involved periods and time domains
- What is the effect on memory layout and interconnect on the execution times?
- Automatic optimized application and data mapping
- Evaluation of digital (multi-core) execution platforms
- Evaluation of software growth scenarios

© Hamann, Kramer, Ziegenbein, Lukasiewycz (Bosch), 2016
MPPA-256 Processor Architecture (Kalray)

- Manycore Processor:
  - 16 compute clusters
  - 2 I/O clusters each with quad-core CPUs, DDR3, 4 Ethernet 10G and 8 PCIe Gen3
  - Data and control networks-on-chip
  - Distributed memory architecture
  - 634/317 GFLOPS SP/DP @ 600MHz

- Compute Cluster:
  - 16 user cores + 1 system core
  - NoC Tx and Rx interfaces
  - Debug & Support Unit (DSU)
  - 2 MB multi-banked shared memory
  - 77GB/s Shared Memory BW
  - 16 cores SMP System

- VLIW Core:
  - 32-bit or 64-bit addresses
  - 5-issue VLIW architecture
  - MMU + I&D cache (8KB+8KB)
  - 32-bit/64-bit IEEE 754-2008 FMA FPU
  - Tightly coupled crypto co-processor
  - 2.4 GFLOPS SP per core @600Mhz

Kalray, 2016
MPPA-256 NoC

- Dual 2D-torus NoC
  - D-NoC: high-bandwidth RDMA
  - C-NoC: low-latency mailboxes
  - 4B/cycle per link direction per NoC
  - Nx10Gb/s NoC extensions for connection to FPGA or other MPPA®

- Predictability
  - Data NoC is configured by selecting routes and injection parameters
  - Injection parameters are the \((\sigma, \rho)\) or (burst, rate) of Cruz network calculus
  - Guaranteed services rely on same methods as in AFDX Ethernet
Safety-Critical Systems with Multicore Platforms

**Goal:** deploy multi-core processors for safety-critical real-time applications (avionics, automotive,...)

**Problem:** concurrent use of shared resources (e.g. interconnect, main memory)
- unknown access latency for a concrete resource access
- complicated timing analysis
- hardware platforms may not be predictable
- Many features are designed by computer architects for **average cases** only

**Solution?**
- Maybe it is up to you.
- Did you see the above challenges?