18-646 - Advanced Parallel Hardware Architectures

Assignment Project Exam Help

https://powcoder.com

Carnegie Mellon University

Add WeChat powcoder

Course Information

- **Lectures:**
  - Tuesday and Thursday 6:00pm-7:20pm ET
- **Office Hours:**
  - Instructor Office Hours: Wednesdays 4:30pm-5:30pm ET
  - TA Office Hours: TBD
- **Course Links:**
  - **Canvas:** https://canvas.cmu.edu/courses/21510/pages/course-schedule
  - **Piazza:** https://piazza.com/class/ktmp02yw9zh5q8
  - **Gradescope:** https://www.gradescope.com/courses/241050

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

https://canvas.cmu.edu/courses/21510/pages/course-schedule

canvas.cmu.edu/courses/21510/pages/course-schedule

Carnegie Mellon University

Account

Dashboard

Courses

Calendar

Inbox

History

Commons

Help

Introduction and Multicore Programming		
Tue, Feb 2	Introduction to How to Write Fast Code II	
Thurs, Feb 4	Advanced Parallel Hardware Architectures	
Tue, Feb 9	Concurrency Opportunity Recognition	**Homework 1 Released**
Thur, Feb 11	Application Design for Multicore Programming	
Tue, Feb 16	Application Design for Manycore	**Mini-Project 1 Released**
Thur, Feb 18	Overview of Mini-Project 1	
Tue, Feb 23	Performance Analysis: Roofline Model	
Thur, Feb 25	No Class	**Homework 1 Due**
Manycore Programming		
Tue, Mar 2	Introduction to CUDA	**Homework 2 Released**
Thurs, Mar 4	Application Design for Manycore Programming	
Mon, Mar 8		**Mini-Project 1 Due**
Tue, Mar 9	Manycore Advanced Optimizations	**Project Proposals Due**
Thur, Mar 11	Mini-Project 1 Review	
Tue, Mar 16	No Class	**Mini-Project 2 Released**
Thur, Mar 18	Overview of Mini-Project 2	**Homework 2 Due**

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Course Goal

- **Course Goal**
  - When your research/application needs to be fast, you will be able to: **FLIRT**
    1. **F**eel comfortable hacking up a solution
    2. **L**everage existing software building blocks
    3. **I**ndicate which platform is the best one to use
    4. **R**eason about why a piece of existing code is slow
    5. **T**ake care of potential performance bottlenecks

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

How to Write Fast Code?

Fast Platforms

- **Multicore platforms**
- **Manycore platforms**
- **Cloud platforms**

+

Good Techniques

- **Data structures**
- **Algorithms**
- **Software Architecture**

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

- We focus on the fast hardware in this lecture
- Combines with **good techniques** to produce fast code…

  …in order to solve a problem or improve an outcome

Outline

- **Landscape of Computing Platforms**
- Hardware Architectures
  - Multicore vs Manycore
  - Instruction level parallelism
  - SIMD
  - Simultaneous multithreading
  - Memory hierarchy
  - System hierarchy
- How to Write Fast Code?

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Landscape of Computing Platforms

- This is what you see most often

  The **exponential increase** in transistor integration, and the **flattening** of clock speed, power consumption, and performance per clock

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Landscape of Computing Platforms

- This is triggering a flurry of technology innovations!

- Let's look at the innovations in three axis:
  - Instruction Set
  - Design Philosophy
  - Power Consumption

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Instruction Set
- **x86**
- **ARM**
- **Power/Cell**
- **SPARC**

Computing Technology Innovations

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Outline

- Landscape of Computing Platforms
- **Hardware Architectures**
  - **Multicore vs Manycore**
  - Instruction level parallelism
  - SIMD
  - Simultaneous multithreading
  - Memory hierarchy
  - System hierarchy
- How to Write Fast Code?

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

(1) Multicore vs Manycore Processors

- What's the difference between Multicore vs Manycore?

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Multicore

Manycore

- Multicore: yoke of oxen
  - Each core optimized for executing a single thread
- Manycore: flock of chickens
  - Cores optimized for aggregate throughput, deemphasizing individual performance

Multicore and Manycore Parallelism

- Similar in scaling trends:
  - Increasing vector unit width
  - Increasing numbers of cores per die
  - Increasing bandwidth to off-chip memory
- Different in optimization points

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Significant Architectural Difference

SIMD single instruction multiple data.

Specifications	Core i7 2600K	GTX580	
Processing Elements	4 cores, 8 way SIMD @2.5-3.4 GHz	16 cores, 16 way SIMD, dual issue @1.55 GHz	0.46x 0.62x
Resident Threads (max)	4 cores, 2 threads, 8 width SIMD 64 strands	16 cores, 48 SIMD vectors, 32 width SIMD 24,576 strands	384x
SP GFLOP/s	160-218	1587	7.3x - 9.9x
Memory Bandwidth	21.4GB/s – 42.7GB/s	192.4 GB/s	4.5x - 9.0x
Die info	995M Transistors 32nm process 216mm^2	3B Transistors 40nm process 520mm^2	

Intel Core i7-2600K

NVIDIA GTX580

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Outline

- Landscape of Computing Platforms
- **Hardware Architectures**
  - Multicore vs Manycore
  - **Instruction level parallelism**
  - SIMD
  - Simultaneous multithreading
  - Memory hierarchy
  - System hierarchy
- How to Write Fast Code?

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Load-store Architecture: MIPS

- Data and instructions are loaded from memory for processing
  - Follows the Von Neuman Architecture – separation of CPU and memory

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

(2) Instruction Level Parallelism (ILP)

- Instructions in a sequence that can be computed at the same time

- Example:
  - Euclidean distance between two points

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$

```
ld r1, x1
ld r2, y1
ld r3, z1
ld r4, x2
ld r5, y2
ld r6, z2
sub r1, r1, r4
sub r2, r2, r5
sub r3, r3, r6
mul r1, r1, r1
mul r2, r2, r2
mul r3, r3, r3
add r1, r1, r2
add r1, r1, r3
sqrt r1, r1
st d, r1
```

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Multiple Valid Instruction Sequence

- Compilers may produce valid instruction sequences that have less ILP when executed in order

Assignment Project Exam Help

```
ld r1, x1
ld r4, x2
sub r1, r4, r1
mul r1, r1, r1
ld r2, y1
ld r5, y2
sub r2, r5, r2
mul r2, r2, r2
add r1, r1, r2
ld r3, z1
ld r6, z2
sub r3, r6, r3
mul r3, r3, r3
add r1, r1, r3
sqrt r1, r1
st d, r1
```

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$

https://powcoder.com

Add WeChat powcoder

```
ld r1, x1
ld r2, y1
ld r3, z1
ld r4, x2
ld r5, y2
ld r6, z2
sub r1, r1, r4
sub r2, r2, r5
sub r3, r3, r6
mul r1, r1, r1
mul r2, r2, r2
mul r3, r3, r3
add r1, r1, r2
add r1, r1, r3
sqrt r1, r1
st d, r1
```

Traditional In-order Pipeline

- An **in-order processor pipeline** could run into issues
  - Reduced ILP and Read/Write operand dependency

Assignment Project Exam Help

```
ld r1, x1
ld r4, x2
sub r1, r4, r1
mul r1, r1, r1
ld r2, y1
ld r5, y2
sub r2, r5, r2
mul r2, r2, r2
add r1, r1, r2
ld r3, z1
ld r6, z2
sub r3, r6, r3
mul r3, r3, r3
add r1, r1, r3
sqrt r1, r1
st d, r1
```

https://powcoder.com

Add WeChat powcoder

http://www.lighterra.com/papers/modernmicroprocessors/

Later Out-of-order Pipelines

- A **out-of-order processor pipeline**:
  - Allows instruction re-ordering
  - Uses register-renaming

```
ld r1, x1
ld r4, x2
sub r1, r4, r1
mul r1, r1, r1
ld r2, y1
ld r5, y2
sub r2, r5, r2
mul r2, r2, r2
add r1, r1, r2
ld r3, z1
ld r6, z2
sub r3, r6, r3
mul r3, r3, r3
add r1, r1, r3
sqrt r1, r1
st d, r1
```

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Exploiting ILP

- Motivation:
  - Given a single, sequential stream of instructions, how to execute it as fast as possible?

Assignment Project Exam Help

- Advantages:
  - No changes in sequential software necessary

https://powcoder.com

Add WeChat powcoder

- Disadvantages:
  - Significantly more complex processor architecture
  - Longer to design the processor
  - Longer to verify the correctness of the processor design
  - Consumes more energy than simple in-order processor

Outline

- Landscape of Computing Platforms
- **Hardware Architectures**
  - Multicore vs Manycore
  - Instruction level parallelism
  - **SIMD**
  - Simultaneous multithreading
  - Memory hierarchy
  - System hierarchy
- How to Write Fast Code?

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

(3) SIMD – Single Inst, Multiple Data

- Taxonomy proposed by Michael J Flynn in 1966
  - **SISD**: No parallelism
  - **SIMD**: Exploits data parallelism
  - **MISD**: Redundancy (used in Space Shuttle flight control)
  - **MIMD**: Distributed computing

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

- SIMD:
  - Can be area and power efficient: Amortize control overhead over SIMD width
  - Parallelism exposed to programmer & compiler

Processor Power Consumption

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

A Brief History of x86 SIMD

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

What to do with SIMD?

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

4 way SIMD (SSE) 16 way SIMD (LRB)

- Neglecting SIMD is expensive
  - AVX: 8 way SIMD, Larrabee: 16 way SIMD
  - NVIDIA: 32 way SIMD, ATI: 64 way SIMD

How to Utilize SIMD Capabilities?

- **Compilers:**
  - Option available in GCC, ICC, and others
  - GCC:
    - Enable by options such as:
      - -ftree-vectorize
      - -msse2
      - -ffast-math
      - -fassociative-math
    - Also enabled by default with "-O3"
  - Examples at:
    - `http://gcc.gnu.org/projects/tree-ssa/vectorization.html#using`

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

- **Hand optimization:**
  - Optimization using intrinsics

Example: Exploiting SIMD Parallelism

- Applied to finding distance between two points:

Assignment Project Exam Help

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$

https://powcoder.com

Add WeChat powcoder

```
V1 = _MM_LOAD(&X1[0])
V2 = _MM_LOAD(&Y1[0])
V1 = _MM_SUB(V2, V1)
V1 = _MM_MUL(V1, V1)
_MM_STORE(&res[0], V1)
add r1, &res[0], &res[1]
add r1, r1, &res[2]
sqrt r1, r1
st d, r1
```

```
ld r1, x1
ld r2, x2
ld r3, x3
ld r4, y1
ld r5, y2
ld r6, y3
sub r1, r4, r1
sub r2, r5, r2
sub r3, r6, r3
mul r1, r1, r1
mul r2, r2, r2
mul r3, r3, r3
add r1, r1, r2
add r1, r1, r3
sqrt r1, r1
st d, r1
```

Example…

```
#include <stdio.h>
#include <pmmintrin.h>  //  header for SSE3
#define k 32

int main(){
  float x[k]; float y[k];          // vectors of length k
  __m128 X, Y;                     // 128-bit values
  __m128 acc = _mm_setzero_ps(); // set to (0, 0, 0, 0)
  float inner_prod, temp[4];
  int i, j;

  for (j=0; j<k; j++){ x[j] = 1.0; y[j] = 1.0;}

  for(i = 0; i < k - 4; i += 4) {
    X = _mm_load_ps(&x[i]);      // load chunk of 4 floats
    Y = _mm_load_ps(y + i);      // alternate way, pointer arithmetic
    acc = _mm_add_ps(acc, _mm_mul_ps(X, Y));
  }
  _mm_store_ps(&temp[0], acc);   // store acc into an array of floats
  inner_prod = temp[0] + temp[1] + temp[2] + temp[3];

  // add the remaining values
  for(; i < k; i++)
    inner_prod += x[i] * y[i];

  printf("Inner product is: %f\n", inner_prod);

  return 0;
}
```

Compile using:

```
gcc -lm -msse3 -O2 test.c
```

Run with:

```
./a.out
```

Result:

```
Inner product is: 32.000
```

SSE Version	gcc header
SSE	xmmintrin.h
SSE2	emmintrin.h
SSE3	Pmmintrin.h

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Common SSE Intrinsics

```
__m128 _mm_setzero_ps(void)// set to (0, 0, 0, 0)

__m128 _mm_set_ps1(float f)// set to (f, f, f, f)

__m128 _mm_set_ps(float w, float x, float y , float z) // set to (z,y,x,w)

__m128 _mm_setr_ps(float w, float x, float y , float z) // set to (w,x,y,z)

__m128 _mm_add_ps(__m128, __m128)

__m128 _mm_sub_ps(__m128, __m128)

__m128 _mm_mul_ps(__m128, __m128)

__m128 _mm_load_ps(float *base_addr) // load 4 floats from base_addr

__m128 _mm_loadr_ps(float *base_addr) // load in reverse order void

_mm_store_ps(float *addr, __m128 val) // write val into location addr
```

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Exploiting SIMD Parallelism

- Motivation:
  - Reduce processor complexity by explicitly representing instruction-level parallelism in vector form

Assignment Project Exam Help

- Advantages:
  - Power-efficient way to improve instruction throughput
  - Exploitable in many compute-intensive applications

https://powcoder.com

Add WeChat powcoder

- Disadvantages:
  - Explicit representation in vector instructions
  - Software requires re-compilation to take advantage of new SIMD capabilities
  - May require hand-tuning to exploit full benefit

Outline

- Landscape of Computing Platforms
- **Hardware Architectures**
  - Multicore vs Manycore
  - Instruction level parallelism
  - SIMD
  - **Simultaneous multithreading**
  - Memory hierarchy
  - System hierarchy
- How to Write Fast Code?

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

https://canvas.cmu.edu/courses/21510/quizzes/55266

Question 1 0 pts

What are the differences between multicore and manycore processors?

Assignment Project Exam Help

Question 2 0 pts

What is instruction-level parallelism?

https://powcoder.com

Question 3 0 pts

Add WeChat powcoder

What is SIMD?

Outline

- Landscape of Computing Platforms
- **Hardware Architectures**
  - Multicore vs Manycore
  - Instruction level parallelism
  - SIMD
  - **Simultaneous multithreading**
  - Memory hierarchy
  - System hierarchy
- How to Write Fast Code?

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

(4) Simultaneous Multithreading

- Question:
  - When multiple threads of work are available, how do we improve performance?

Assignment Project Exam Help

```
ld r1, x1
ld r4, y1
sub r1, r4, r1
mul r1, r1, r1
ld r2, x2
ld r5, y2
sub r2, r5, r2
mul r2, r2, r2
add r1, r1, r2
ld r3, x3
ld r6, y3
sub r3, r6, r3
mul r3, r3, r3
add r1, r1, r3
sqrt r1, r1
st d, r1
```

https://powcoder.com

Add WeChat powcoder

Simultaneous Multithreading (SMT)

- Question:
  - When multiple threads of work are available, how do we improve performance?

Assignment Project Exam Help

different tasks (instructions)

https://powcoder.com
Add WeChat powcoder

Exploiting SMT Parallelism

- Motivation:
  - Capturing the opportunity to run faster when more than one thread of instructions are available

Assignment Project Exam Help

- Advantages:
  - Gain power-efficiency by increase processor pipeline utilization

https://powcoder.com

Add WeChat powcoder

- Disadvantages:
  - Requires multiple threads available
  - **May trigger conflicts in shared cache during execution**
  - Does not improve latency of each thread

Outline

- Landscape of Computing Platforms
- **Hardware Architectures**
  - Multicore vs Manycore
  - Instruction level parallelism
  - SIMD
  - Simultaneous multithreading
  - **Memory hierarchy**
  - System hierarchy
- How to Write Fast Code?

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

(5) Memory Hierarchy

- **Cache**:
  - A piece of fast memory close to the compute modules in a microprocessor
  - Allows faster access to a limited amount of data
  - Physical properties of wires and transistors determines trade-off between **cache capacity** and **access throughput/latency**

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Capacity: Size, e.g. number of bytes of data

Latency: From start to finish, in units of time, e.g. CPU clock cycles

Throughput: Tasks accomplished per unit time, e.g. GB/s

(5) Memory Hierarchy

- **Cache**:
  - A piece of fast memory close to the compute modules in a microprocessor
  - Allows faster access to a limited amount of data
  - Physical properties of wires and transistors determines trade-off between **cache capacity** and **access throughput/latency**

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

(Capacity), (throughput), (latency)

Exa-Bytes, 0.1 GB/s, ~1-10B cycles

3TB, 6 GB/s, ~1M cycles

16GB, 34.08 GB/s, ~200 cycles

8 MB, 108.8 GB/s, 26-31 cycles

256 KB, 108.8 GB/s, 12 cycles

32 KB Data Cache, 163.2 GB/s, 4-6 cycles

Intel Core2 – 2600k @ 3.4GHz
(viewed from one core)

Working with a Memory Hierarchy

- **Writing fast code → get data from the fastest (closest) level**

- When requesting data from a level in memory hierarchy of limited size, there are two outcomes
  - **Hit**: Data is available in the level
  - **Miss**: Data is missing from the level

- **For fast code → minimize misses, maximize hits**

- What **application behavior** can a memory hierarchy help with?

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Working with a Memory Hierarchy

- **Principle of Locality**
  - The phenomenon of the same value or related storage locations being frequently accessed, usually amenable to performance optimization

Assignment Project Exam Help

- **Temporal Locality**
  - Reuse of specific data and/or resources within relatively small time durations

https://powcoder.com

Add WeChat powcoder

$$A \quad B$$

$$AB = \begin{pmatrix} -16 & -4 & -29 & 37 & -46 \\ 10 & 18 & 21 & -17 & 22 \\ 6 & -22 & -2 & -10 & 12 \\ -13 & -1 & -7 & 6 & -9 \end{pmatrix}$$

$$(-1)(3) + (-4)(4) + (-2)(-1)$$

$$-3 + -16 + 2$$

$$-17$$

- **Spatial Locality**
  - Use of data elements within relatively close storage locations

When would you get a miss?

- Three types of misses in a memory hierarchy
  - **Compulsory** misses: caused by the first reference
  - **Capacity** misses: due to the finite size of the memory hierarchy
  - **Conflict** misses: due to policy of replacement, potentially avoidable

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Exploiting Benefits of Mem Hierarchy

- Motivation:
  - When application exhibit data locality, cache/memory provides faster access to frequently used data

Assignment Project Exam Help

- Advantages:
  - Allows faster access to data for computation

https://powcoder.com

  - Caches are managed storage: transparent to the end-user for functional purposes

Add WeChat powcoder

  - Lower the energy consumption when getting a “cache-hits”

- Disadvantages:
  - When using multiple threads share a cache, they will compete for cache space

Outline

- Landscape of Computing Platforms

- Hardware Architectures
  1. Multicore vs Manycore
  2. Instruction level parallelism
  3. SIMD
  4. Simultaneous multithreading
  5. Memory hierarchy
  6. System hierarchy

- How to Write Fast Code?

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

(6) System Architecture

- A Journey through the opportunities to write ***fast code*** with:
  - Multicore
  - Manycore
  - The Cloud

Cloud

Manycore

Multicore

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Section 1 - Multicore

- Writing fast code to exploit multicore parallelism
  - SIMD level parallelism
  - Core level parallelism
  - Memory hierarchy

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Section 2 - Manycore

- GPU System Architecture

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Section 2 - Manycore

- GPU System Architecture

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Section 3 – The Cloud

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Section 3 – The Cloud

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

A Game of Granularity

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Outline

- Landscape of Computing Platforms

- Hardware Architectures
  1. Multicore vs Manycore
  2. Instruction level parallelism
  3. SIMD
  4. Simultaneous multithreading
  5. Memory hierarchy
  6. System hierarchy

- How to Write Fast Code?

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

Outline

- Landscape of Computing Platforms
- Hardware Architectures
  - Multicore vs Manycore
  - Instruction level parallelism
  - SIMD
  - Simultaneous multithreading
  - Memory hierarchy
  - System hierarchy
- **How to Write Fast Code?**

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

How to Write Fast Code?

Fast Platforms

- **Multicore platforms**
- **Manycore platforms**
- **Cloud platforms**

+

Good Techniques

- **Data structures**
- **Algorithms**
- **Software Architecture**

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

- We focus on the fast hardware in this lecture
- Combines with **good techniques** to produce fast code…

  …in order to solve a problem or improve an outcome