



Microbenchmarking the GT200 GPU



Objective

- Measure low-level performance
- Speculate about the microarchitecture of the GT200 GPU



Objective

- Pipeline
 - Several pipelines
 - Pipeline latency and throughput
- Memory hierarchy
 - Caches
 - Off-chip DRAM latency and throughput

Pipeline Benchmarks

- Use clock() for timing
 - Two instructions: move and left shift
- Unrolled sequence of arithmetic ops
 - One thread for latency measurement
 - 512 threads for throughput

```
uint start_time = clock();  
repeat256(a = a + b);  
uint stop_time = clock();
```

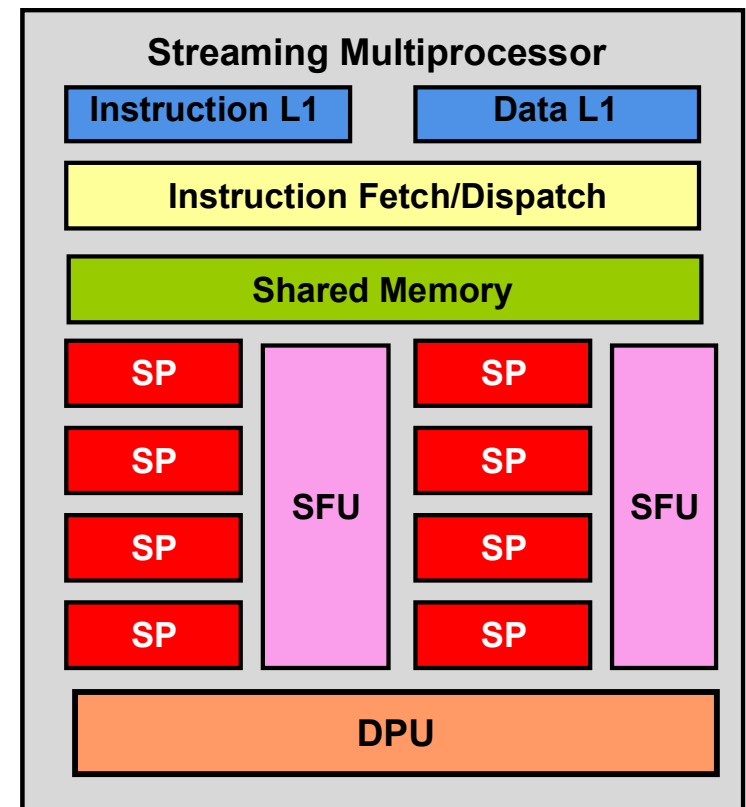
[Save timers to global memory]

Pipeline Benchmark Caveats

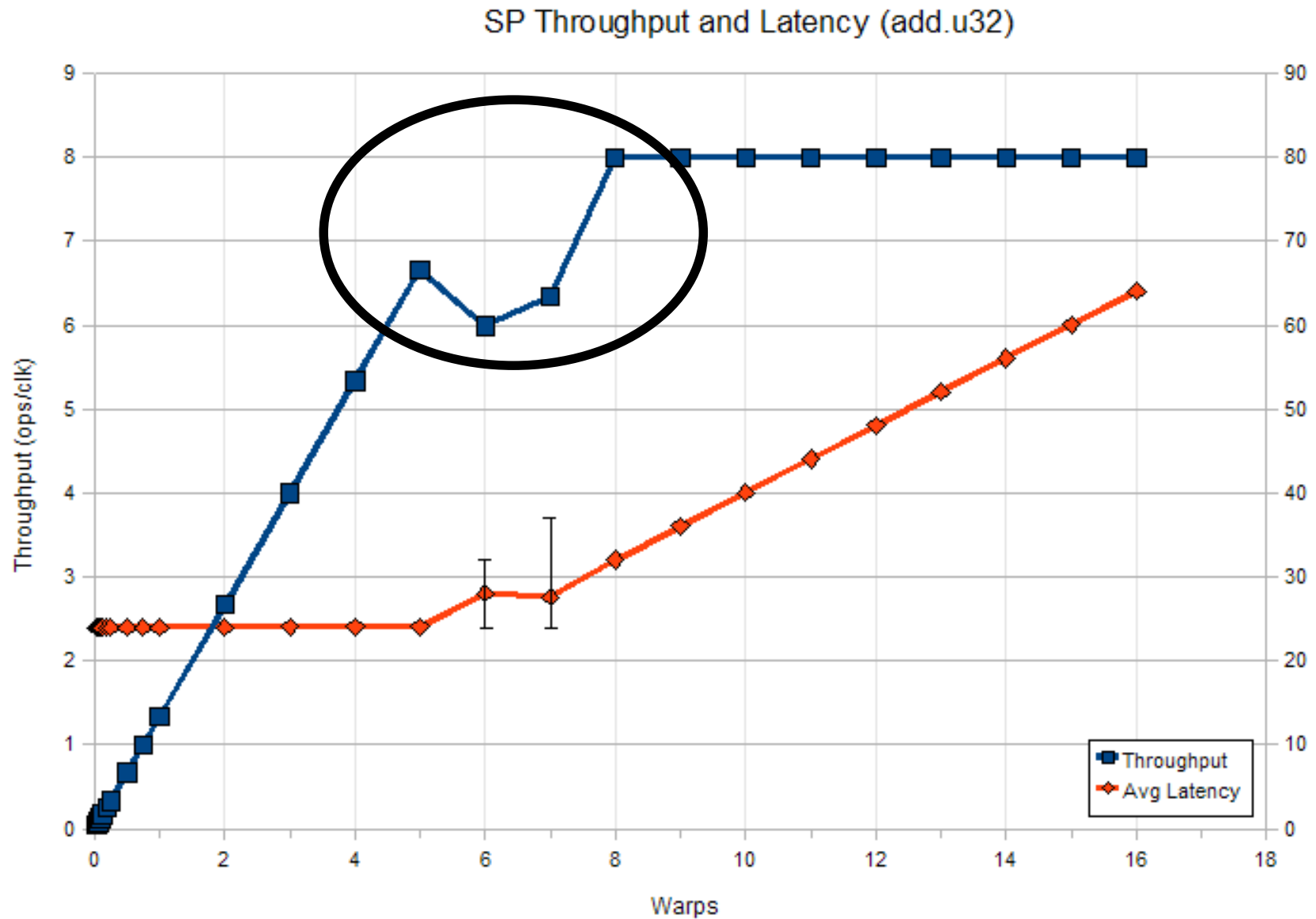
- Unrolled loop must fit in 8 KB icache
 - Variable instruction length (32- and 64-bit)
- Instruction cache cold misses
 - Measure only second iteration of loop
- PTX “instructions” to real instructions mapping not one-to-one
- Optimization by `nvopenccl` *and* `ptxas`
 - Need to circumvent optimizations by clever coding

Pipeline Functional Units

Functional unit		Latency (clocks)	Throughput (ops/clock)
SP	add, sub, (int, float)	24	8
SFU	rsqrt, log2f, (float)	28	2
DPU	add, sub, (double)	48	1



SP Pipeline



32-Bit Multiplication

- $r1 = r1 * r2$
 - $r5 = r2.lo * r1.hi$ (mul24)
 - $r5 = (r2.h1 * r1.lo) + r5$ (mad24)
 - $r5 = r5 \ll 16$ (shl)
 - $r1 = (r2.lo * r1.lo) + r5$ (mad24)
- Latency is 96 cycles
- Throughput is 2 operations per cycle

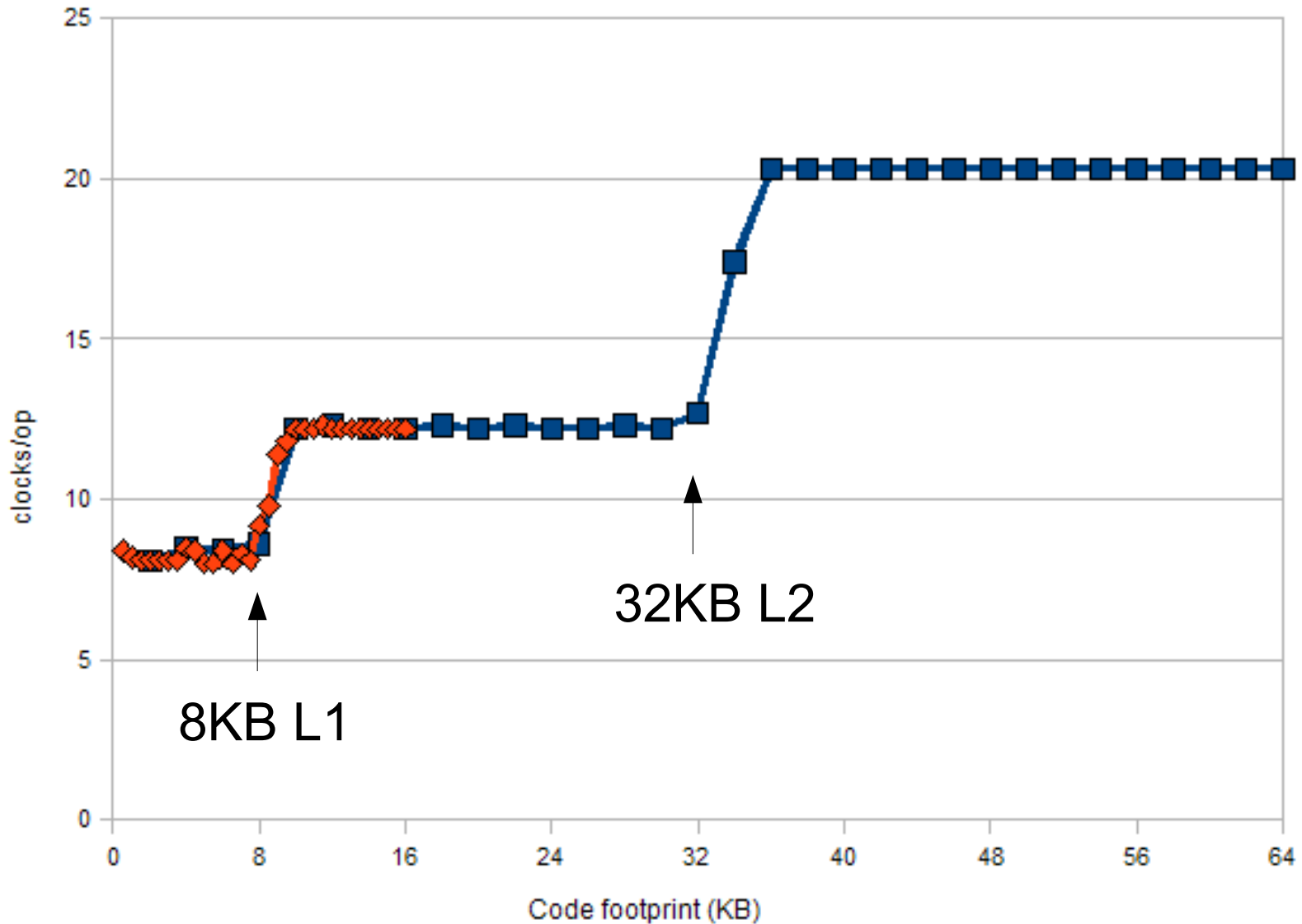
Instruction Cache

- Sequence of independent 64-bit SP instructions (cvt)
- Loop over of varying amount of code
 - Cache misses increase measured runtime
- L2 icache is supposed to be shared with L2 constant cache



Instruction Cache

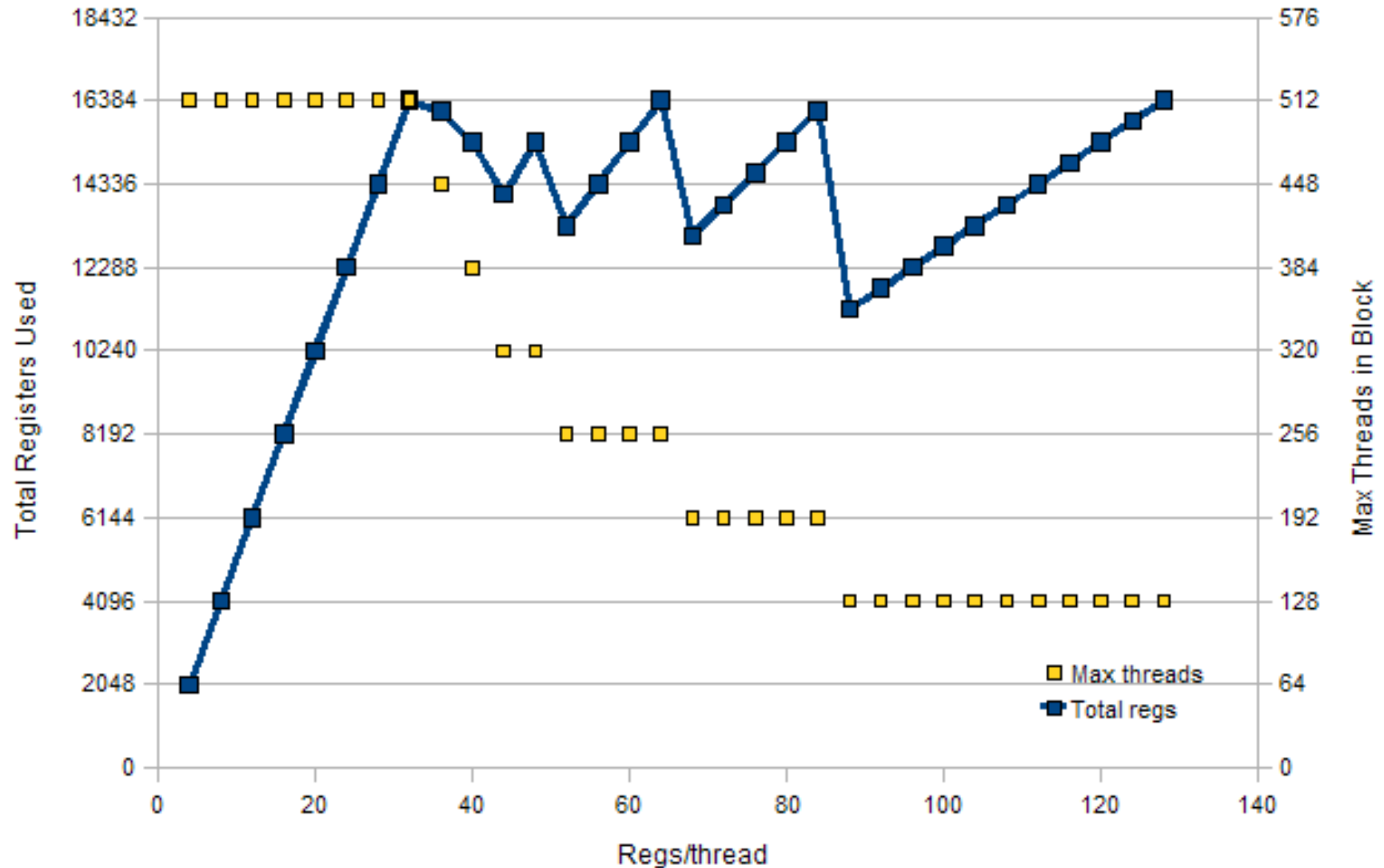
icache_test4 abs





Register File Allocation

Max Threads and Regs Used vs. Regs/Thread



Register File Allocation

- Register file has 64 banks
 - Can not utilize entire register file if not a multiple of 64 threads
 - No bank conflicts, as alluded to in manual
- 8 banks per SP allows single-ported memories to be used for register files
 - $\frac{1}{2}$ bandwidth for ALU
 - $\frac{1}{2}$ bandwidth for memory fetch

Global Memory

- Read Latency
 - 435 - 450 cycles
 - Chain of dependent reads
 - repeat 1024 ($j = \text{array}[j]$)
 - where $\text{array}[j] = j + \text{stride}$