Dept. of Computer Science and Engineering, National Sun Yat-sen University
PhD Qualifying Examination, Spring Semester 2018

Subject: Computer Architecture:

1. (16% total) You are designing a processor for an IoT embedded system. Based on the analysis of the monitoring software, the instructions listed in Table 1 show the specified latencies.

Table 1.

Instructions	Frequency	Latency
load	10%	7 cycles
store	15%	10 cycles
branch	15%	4 cycles
add	55%	4 cycles
divide	5%	5 cycles

1.1 (4%) What is the CPI of your processor on this mix of instructions?
1.2 (4%) Based on your design, if you could halve the cycle latency of any single category of instruction, but at the cost of increasing the cycle time by 20%. Should you make this change, and if so, what category of instruction should you speed up? (load, store, branch, add, or divide)
1.3 (4%) What is the CPI of your new design?
1.4 (4%) What is the speedup of your revised design over the original one, rounded to the nearest tenth?

2. (16% total) Please answer the following questions:
2.1 (4%) What do VLIW, superscalar execution, and array processing concepts have in common?
2.2 (6%) Provide two reasons why a VLIW microarchitecture is simpler than a “same-width” superscalar microarchitecture?
2.3 (6%) Provide two reasons why a superscalar microarchitecture could provide higher performance than a “same-width” VLIW microarchitecture?

3. (20% total) In Table 2, we have given you four different sequences of addresses generated by a program running on a processor with a data cache. Cache hit ratio for each sequence is also shown in Table 2. Assuming that the cache is initially empty at the beginning of each sequence, find out the following parameters of the processors’ data cache:
3.1 (5%) Associativity? (1, 2, or 4 ways)
3.2 (5%) Block size? (1, 2, 4, 8, 16, or 32 bytes)
3.3 (5%) Total cache size? (256 bytes, or 512 bytes)
3.4 (5%) Replacement policy? (LRU, or FIFO)

Table 2.

Sequence No.	Address Sequence	Hit Ratio
1	0, 2, 4, 8, 16, 32	0.33
2	0, 512, 1024, 1536, 2048, 1536, 1024, 512, 0	0.33
3	0, 64, 128, 256, 512, 256, 128, 64, 0	0.33
4	0, 512, 1024, 0, 1536, 0, 2048, 512	0.25

4. (10% total) You are requested to parallelize an old program so that it could run faster on modern multicore processors. Please answer the following questions.
4.1 (6%) If your program can perform 90% of its work (measured as processor-seconds) in the parallel portion and 10% of its work in the serial portion. The parallel portion is perfectly parallelizable. What is the maximum speedup of the program if the multicore processor had an infinite number of cores?
4.2 (4%) How many processors would be required to attain a speedup of 4?

5. (20% total) Assume a GPU architecture that contains 10 SIMD processors. Each SIMD instruction has a width of 32 and each SIMD processor contains 8 lanes for single-precision arithmetic and load/store instructions, meaning that each non-diverged SIMD instruction can produce 32 results every 4 cycles. Assume a kernel that has divergent branches that causes on average 80% of threads to be active. Assume that 70% of all SIMD instructions executed are single-precision arithmetic and 20% are load/store. Since not all memory latencies are covered, assume an average SIMD instruction issue rate of 0.85. Assume that GPU has a clock speed of 1.5 GHz.

5.1 (5%) Compute the throughput, in GFLOPS/sec, for this kernel on this GPU.

5.2 Assume that you have the following choices, what is the speedup in throughput for each of these improvements?

- (5%) Increasing the number of single-precision lanes to 16.
- (5%) Increasing the number of SIMD processors to 15 (assume this change does not affect any other performance metrics and that the code scales to the additional processors)
- (5%) Adding a cache that will effectively reduce memory latency by 40%, which will increase instruction issue rate to 0.95.

6. (18% total) The effect of the interconnection network topology on the clock cycles per instruction (CPI) of programs running on 64-processor distributed-memory multiprocessor is examined. The processor clock rate is 3.3 GHz and the base CPI of an application with all references hitting in the cache is 0.5. Assume that 0.2% of the instructions involve a remote communication reference. The cost of a remote communication reference is $(100+10h)$ ns, where h is the number of communication network hops that a remote reference has to make to the remote processor memory and back. Assume that all communication links are bidirectional.

6.1 (9%) Calculate the worst-case remote communication cost when the 64 processors are arranged as a ring, as an 8×8 processor grid, or as a hypercube. (hint: The longest communication path on a $2^{\boldsymbol{n}}$ hypercube has $\boldsymbol{n}$ links.)

6.2 (9%) Compute the base CPI of the application with no remote communication to the CPI achieved with each of the three topologies in 6.1.