Computer Design Supervision Problems III

Kevin Zhou

November 3, 2015

1 CISC x86 assembly

(a) Find the CPU architecture on your laptop/desktop. You could run lscpu on Linux machines, or running the free “CPU-Z” program on Windows. Briefly explain what do the following cpu architectures stand for and their differences:

(i) 386, i386
(ii) x86, x64, x86-64, amd64
(iii) IA-32, IA-64, Intel 64

(b) Compared to RISC architectures, there is much less number of general purpose integer registers \((A, C, D, B)\). Explain why they made this design choice and why they made the order of \(A, C, D, B\). Look up in the Intel Instruction Set Manual, find the meaning of the prefix \(E, R\) and the suffix \(X, H, L, SI, DI, IP\).

(c) Briefly describe the usage and effects of the x86 instruction LEA \((\text{Load Effective Address})\). You could look up in the Internet from the Intel®64 and Intel-32 architectures software development manual. And think how would the compiler optimise the following two operations while achieving minimum latency and occupying minimum number of registers. Suppose variable \(x\) is stored in \(eax\), \(y\) is \(ecx\) and \(z\) is \(edx\)

\[
y = 41 \times x + 6
\]
\[
z = 12 \times x + y + 10
\]

Calculate the number of cycles based on your proposed assembly. You could find exact cycle count for each instruction from [http://www.agner.org/optimize/instruction_tables.pdf](http://www.agner.org/optimize/instruction_tables.pdf)

(d) For the following two code snippet (loopA) and (loopB), which one runs faster? Explain why you think so.

```assembly
loopA:
  mov eax, [esi]
  xor eax, 0E5h
  add [edi], eax
  add esi, 4
  add edi, 4
  dec ecx
  jnz loopA
```
loopB:

```
mov   eax, [esi]
mov   ebx, [esi+4]
xor   eax, 0E5
xor   ebx, 0E5
add   [edi], eax
add   [edi+4], ebx
add   esi, 8
add   edi, 8
sub   ecx, 2
jnz   loopB
```

(e) The fundamental principle of debuggers (GDB or LLDB) is to temporarily replace an instruction INT 3 in a running program in order to set a breakpoint. Find in the effect of INT in the instruction manual and brief why GDB could attach the running application, insert the breakpoint and halt the application’s progress.

(f) Look up in the Internet and explain what is Stack Buffer Overflow (not the website) and how attackers could use stack overflow to redirect the OS/app to run their virus kernels.

2 Java Virtual Machine

(a) List the advantages and disadvantages of JVM and its application.

(b) Jazelle DBX (Direct Bytecode eXecution) allows some old ARM processors to execute Java bytecode in hardware as a third execution state alongside the existing ARM and Thumb modes. The implementation of Jazelle DBX is to add a low-level binary translation stage between the fetch and decode stage in the processor instruction pipeline. The extension could significantly boost the performance of Java ME games and applications. Besides adding hardware extensions on JVM, could you make a standalone CPU that directly executes JVM byte code? How would it differ from other architectures?

3 Pipelining techniques

<table>
<thead>
<tr>
<th>instruction fetch</th>
<th>branch, decode</th>
<th>execute</th>
<th>memory access</th>
<th>write back</th>
</tr>
</thead>
</table>

(a) With reference to the pipeline above: What is a control hazard and how can it be dealt with? What are data hazards and how can they be eliminated?

(b) Is the pipelining technique scalable to ever more pipeline stages? Justify your answer.

(c) If the memory access results in a cache miss, what happens to the pipeline?

(d) What impact does pipeline length have on clock frequency? Why might a shorter pipeline result in a more power-efficient design?

(e) For arithmetic operations the result is available after the execute stage. These results could be written directly to the register file during the memory access stage, but what would be the disadvantages of doing so?
4 Memory hierarchy and caches

Read the lecture notes 10 and the appendix C “Review of Memory Hierarchy” from the book “Computer Architecture A Quantitative Approach” and answer the following questions:

(a) What is the difference between the latency and the bandwidth of a communication link?

(b) Why do modern processors need caches and what statistical properties do they exploit to improve processor performance? Is there any case that caches are not needed?

(c) Name and describe three reasons for a cache miss. For each reason, suggest a technique for reducing the number of misses.

(d) What is a data cache and how does it differ from an instruction cache?

(e) How are cache tags used in direct-mapped and fully-associative caches?

(f) What cache line replacement policies might be used for set-associative and direct-mapped caches?

(g) What impact will an operating-system-managed context switch have on cache hit rate? Justify your answer.

5 Hardware operating system support

(a) A translation look-aside buffer (TLB) is a form of cache, but what does it store and how are TLB misses handled?

(b) Why are TLBs always much smaller than caches?

(c) TLB misses can be handled in hardware or software. If misses are handled purely in software, how are they made transparent to the user code?

(d) A processor architect proposes the use of a large direct-mapped TLB rather than the traditional small fully-associative TLB. What might the performance implications be?

(e) How does the memory protection realised using the virtual memory technique?

End of Problems