Spring 2014 CSE Qualifying Exam
Core Subjects

February 22, 2014

Architecture

1. ROB trace - single issue
   Assume the following:

   - **Ten ROB slots.**
   - 1 integer Execution unit, 1 reservation stations, one cycle to execute.
   - 1 Floating Add Unit, 3 reservation stations, 7 cycles to execute.
   - 1 Floating MULT Unit, 2 reservation stations, 12 cycles to execute.
   - 1 Load unit with only **two reservation stations.**
   - Loads on hits require 2 clock cycles, both performed in the Load Unit.
   - Functional units are **not pipelined!!!!**
   - There is no forwarding between functional units; results are communicated by the common data bus (CDB).
   - The execution stage (EX) does both the effective address calculation and the memory access for loads and stores.
   - Thus, the pipeline is IF/ ID/ IS/ EX/ WB.
   - Assume all memory references are hits and branch predictions are branch taken and are correct.
   - The issue (IS) and write-back (WB) result stages each require one clock cycle.
   - Assume that the Branch on Not Equal to Zero (BNEZ) instruction requires one clock cycle.

```
loop:  LD     F0, 0(R1)
       MUL.D  F4, F0, F0
       ADD.D  F8, F8, F0
       S.D    F8, 0(R2)
       ADD.D  F6, F6, F4
       DADDIU R1, R1, +8
       DADDIU R2, R2, +8
       BNE    R1, R3, loop
```

(a) Fill in the table below for as much as you can assuming the BNE is taken and succeeds. (Do not add rows to the table.)

Iter.	Instruction	Issues at	Execute	memory	Write CDB	Commit

(b) What are the major differences in Tomasulo's and Reorder buffer (ROB)?

(c) What additional hardware is rquired (if any) to support two CDBs?

(d) What additional hardware is rquired to support loads requiring 10 or 100 cycles due to misses to the L1 and L2 caches.

(e) What additional hardware is rquired to support dual issue?

2. In the system we are analyzing the memory has:

- Separate L1 instruction and data caches, HitTime = Processor Cycle Time
- 64KB L1 instruction cache with 1% miss rate, 32B blocks

- 64KB L1 data cache with 10% miss rate, 32B blocks
- 512K L2 unified cache with 64B blocks, local miss rate 20%, Hit Time = 8 cycles,
- Main Memory Access time is 100 cycles for the first 128 bits and subsequent 128 bit chunks are available every 8 cycles.
- Both L1 caches are direct mapped, L2 four-way associative.
- Assume there are no misses to main memory.

(a) What is the Miss Penalty for accesses to L2?

(b) What is the average memory access time for instruction references?

(c) What is the average memory access time for data references?

(d) Assume the only memory reference instructions are loads(25%) and stores(5%). What percentage of total memory references are data references?

(e) What is the Average memory access time?

3. Your company has just bought a new dual-core processor, and you have been tasked with optimizing your software for this processor. You will run two applications on this dual core processor, but the resource requirements are not equal. The first application needs 75% of the resources, and the other only 25% of the resources.

   (a) Given that 60% of the first application is parallelizable, how much speedup would you achieve with that application if run in isolation?

   (b) Given that 95% of the second application is parallelizable, how much speedup would this application observe if run in isolation?

   (c) Given that 60% of the first application is parallelizable, how much overall system speedup would you observe if you parallelized it, but not the second application?

   (d) How much overall system speedup would you achieve if you parallelized both applications, given the information in parts (a) and (b)?

Compilers

1. Consider the following grammar:

$$\begin{array}{rcl} S & \to & V \;=\; E \\ E & \to & E \,*\, E \;\;|\;\; E \,+\, E \;\;|\;\; (\; E\;) \\ E & \to & V \\ V & \to & \langle id \rangle \\ V & \to & \langle id \rangle \;[\; \langle Elist \rangle \;] \\ \langle Elist \rangle & \to & \langle Elist \rangle \;\mathbf{c}\; E \\ \langle Elist \rangle & \to & E \end{array}$$

   (The **c** stands for "comma.")

   (a) Construct the sets of LR(1) items for this grammar. (Stop at 8 sets.)

      i. Compute I_0

      ii. For which symbols x is $\mathrm{GOTO}(I_0, x)$ nontrivial?

      iii. For each such symbol x, compute $\mathrm{GOTO}(I_0, x)$.

      iv. Compute $\mathrm{GOTO}([P, \$], =)$, where P is $S \to V \cdot \;=\; E$.

   (b) For the sets of LR(1) items from above, provide entries of the LR(1) parse table.

2. Consider the following CFG for formulas in predicate logic (the start symbol is $\langle start \rangle$):

$$\begin{array}{rcl} \langle start \rangle & ::= & \langle formula \rangle \\ \langle formula \rangle & ::= & \langle formula \rangle_1 \vee \langle conjunction \rangle \\ & | & \langle conjunction \rangle \\ \langle conjunction \rangle & ::= & \langle conjunction \rangle_1 \wedge \langle literal \rangle \\ & | & \langle literal \rangle \\ \langle literal \rangle & ::= & \neg \;\langle literal \rangle_1 \\ & | & (\; \langle formula \rangle \;) \\ & | & (\; \langle quantifier \rangle \;\mathbf{VAR}\;)\;\; \langle literal \rangle_1 \\ & | & \mathbf{R}\;\;(\;\mathbf{VAR}_1\;,\;\mathbf{VAR}_2\;) \\ \langle quantifier \rangle & ::= & \forall \\ & | & \exists \end{array}$$

   Here, the token **R** denotes a fixed binary relation symbol, and the token **VAR** represents any variable name.

   We say that a variable occurrence in a formula is *free* if it is not within the scope of a quantifier of the same variable. For example, in the formula

$$R(x, y) \wedge (\forall x)(\exists z)\neg R(x, y),$$

the first occurrence of x and both occurrences of y are free, but the second and third occurrences of x are not free. The occurrence of z is not free, either.

Add semantic rules to the grammar above that correctly set a Boolean attribute *isFree* of each occurrence of **VAR** in the formula. You may assume that each **VAR** has a given *name* attribute, which is a string. You may also use a setOfStrings ADT for an unordered, duplicate-free collection of strings that supports the following nondestructive operations:

setOfStrings emptySet() returns the empty set of strings.

boolean isMember(string x, setOfStrings S) returns true iff x is a member of the set S.

setOfStrings add(string x, setOfStrings S) returns $S \cup \{x\}$.

You may define any attributes you find useful. [Hint: most attributes are inherited.]

3. The following fragment of 3-address code was produced by a nonoptimizing compiler:

```
 1  start:  i = 1
 2  loop1:  if i > n goto part2
 3          j = 1
 4          sum = 0
 5  loop2:  if j > i goto fin1
 6          o = i * 8
 7          o = o + j
 8          s = a[o]
 9          t = j * 8
10          v = a[t]
11          y = s * v
12          if s < n goto loop1
13          sum = sum + y
14          j = j + 1
15          goto loop2
16  fin1:   j = sum
17          goto fin2
18  fin2:   i = i + o
19          o = i * 8
20          a[o] = sum
21          goto loop2
22  part2:  no-op
```

Assume that there are no entry points into the code from outside other than at `start`.

   (a) (20% credit) Decompose the code into basic blocks $B_1, B_2, \ldots$, giving a range of line numbers for each.

(b) (30% credit) Draw the control flow graph, describe any unreachable code, and coalesce any nodes if possible.

(c) (30% credit) Is variable `i` live just before line 7? Is the variable `o` live just before line 9? Is the variable `y` live just before line 14? Explain. Assume that `n` and `sum` are the only live variables immediately after line 22.

(d) (20% credit) Describe any simplifying transformations that can be performed on the code (i.e., transformations that preserve the semantics but reduce (i) the complexity of an instruction, (ii) the number of instructions, (iii) the number of branches, or (iv) the number of variables).

Algorithms

1. Find tight asymptotic bounds on any positive-valued function $T(n)$ satisfying the following recurrence for all $n \geq 2$:

$$T(n) = T(\lfloor \sqrt{n} \rfloor) + T(n-1) + n \; .$$

   That is, find an expression $f(n)$, as simple as possible, such that $T(n) = \Theta(f(n))$. Prove that $T(n) = O(f(n))$ (upper bound only) using the substitution method.

2. You are given as input an array $S[1 \ldots n]$ of n integers and a positive integer d.

   (a) (50%) Design an algorithm to determine whether there are two elements $S[i]$ and $S[j]$ of S with $i < j$ such that $S[j] - S[i] = d$. The algorithm should run in time $O(n \lg n)$.

   (b) (50%) Suppose that S is given in sorted order. Design an algorithm to solve this problem in time $O(n)$.

3. Consider the following decision problem:

   DOUBLE HAMILTONIAN CIRCUIT (DHC)
   Instance: An simple undirected graph $G = (V, E)$ with no self-loops.
   Question: Is there a circuit in G that includes each vertex of G exactly twice?

   As usual, a circuit in G is by definition a sequence of vertices $\langle v_1, v_2, \ldots, v_\ell \rangle$, where $\ell \geq 3$ and $\{v_i, v_{i+1}\} \in E$ for all $1 \leq i < \ell$, and $\{v_\ell, v_1\} \in E$. Also, the same edge is never traversed twice in a row, i.e., $v_i \neq v_{i+2}$ for all $1 \leq i \leq \ell - 2$, $v_{\ell-1} \neq v_1$, and $v_\ell \neq v_2$. The question is whether there exists such a sequence where each vertex shows up in the sequence exactly twice.

   DHC is clearly in NP. Show that DHC is NP-hard via a polynomial reduction from HAMILTONIAN CIRCUIT.

Theory

Not given in Spring 2014.