Written Exam / Tentamen

Computer Organization and Components / Datorteknik och komponenter (IS1500), 9 hp
Computer Hardware Engineering / Datorteknik, grundkurs (IS1200), 7.5 hp

KTH Royal Institute of Technology

2018-06-05

08.00-13.00

Suggested Solutions

Part I: Fundamentals

In part I, on the real exam, only short answers are expected. The elaborated answers given here are just for your information, and are not needed on the real exam.

1. **Module 1: C and Assembly Programming**

   (a) Short answer: `0x230bffe9`.

   Max 3 points. One point if 2 bytes are correct, two points if 3 bytes are correct, and three points if all 4 bytes are correct.

   (b) Short answer: 8, 41, 82, `unknown`

   Max 4 points. One point for each correct value.

   (c) Short answer: The source code line could be

   ```
   boo(list, 1);
   ```

   Note that a value `0` for the second argument also gives a valid solution.

   Max 1 points. One point for a correct answer.

2. **Module 2: I/O Systems**

   (a) Short answer: `temp << 5`

   Max 2 points. One point for stating `temp` and shift left. One point for shifting 5 bits.

   (b) Short answer:

   ```
   lui     $t1,0x1b43
   ori     $t1,$t1,0x0010
   lui     $t2,0xabbb
   ori     $t2,$t2,0x8050

   lw      $t0,0($t1)
   srl     $t0,$t0,6
   andi    $t0,$t0,0x3

   lw      $t3,0($t2)
   andi    $t3,$t3,0xff9f
   sll     $t0,$t0,5
   or      $t3,$t3,$t0
   sw      $t3,0($t2)
   ```

Max 4 points. One point for setting up the two registers with correct addresses. One point for correctly loading the switches values, shifting to the right, and masking (using `andi`). One point for loading the value for the LEDs and masking. One point for `or`-ing and to store back the value to the LEDs port. Shifting correctly is not necessary for full point.

(c) Short answer:

i. False. DMA is an efficient approach to copy data between different parts of the memory. It has nothing to do with pointers in C.

ii. True.

Max 2 points. One point for each correct answer, including a motivation in that case the answer is false.

3. **Module 3: Logic Design (for IS1500 only)**

(a) Short answer:

Tick	A	B	C	D
1	2	3	1	0
2	3	2	1	0
3	0	3	0	0
4	1	0	1	0

Max 5 points. Give 5 points if all 14 numbers are correct, 4 points if at least 11 numbers are correct, 3 points if at least 8 numbers are correct, 2 points if at least 5 numbers are correct, and 1 point if at least 2 numbers are correct.

(b) Short answer: The register file can save 4096 bytes and the address width of the write port is 12 bits.

Elaborated answer: The data bus width is 8 bits (1 byte). There are $2^{12} = 4096$ positions. Hence, the capacity is 4096 bytes. The address bus width is the same for the read port and the write port. Hence, the address bus width of the write port is 12 bits.

Max 3 points. Two points for the capacity and one point for the write port address bus width.

4. **Module 4: Processor Design**

(a) Short answer: $A =$ `0x0`, $B =$ `unknown`, $C =$ `0xffffffcc`, $D =$ `0x11`, and $E =$ `0x40002234`.

Max 5 points. One point for each correct signal value.

(b) Short answer: There is only one data hazard, between the `slt` and the `beq` instructions, involving register `$t1`. The problem can be solved by stalling the pipeline one clock cycle, and then forward.

Max 3 points. One point for stating the right register, one point for pointing out the two correct instructions, and one point for stating stalling.

5. **Module 5: Memory Hierarchy**

   (a) Short answer: There are 256 sets, the associativity is 4, and there are in total 1024 valid bits.

   Elaborated answer: Since the tag field is 20 bits and the block size is 16 bytes (4 bits of the address field), the set field size is $32 - 20 - 4 = 8$ bits. Hence, there are 256 sets. There are in total $16384/16 = 1024$ blocks. Hence, the associativity of the cache is $1024/256 = 4$. There is one valid bit associated with each way and each set. That is, $4 \cdot 256 = 1024$ bits.

   Max 3 points. One point for the correct number of sets, one point for the correct associativity number, and one point for the correct number of valid bits.

   (b) Short answer: The instruction cache miss rate is $\frac{3}{4}$, the data cache miss rate is $\frac{2}{3}$, and there is only spatial locality.

   Elaborated answer:

   The `lui` instruction starts one word into the cache block. We execute 4 instructions (4 memory accesses) and there are 3 cache misses (because of the first instruction is not aligned to the 8 byte blocks). Hence, the instruction cache miss rate is $\frac{3}{4}$

   There are in total 3 load word instructions. By calculating the load addresses, we can directly see that two different cache blocks will be loaded, and it is therefore 2 cache misses. Hence, the data cache miss rate is $\frac{2}{3}$.

   No addresses are revisited, so there is no temporal locality. But, in both the instruction and data cache cases, there is clear spatial locality.

   Max 5 points. Two points for correct instruction cache miss rate (one point for the numerator, one point for the denominator), two points for the correct data cache miss rate, and one point for answering spatial locality (only).

6. **Module 6: Parallel Processors and Programs**

   (a) Short answer: The maximal speedup is $\frac{4}{3}$.

   Elaborated answer: Using Amdahl's law, we get the maximal speedup $\frac{1}{\frac{0.25}{N}+0.75} = \frac{1}{0.75} = \frac{1}{\frac{3}{4}} = \frac{4}{3}$ when $N \to \infty$.

   Max 3 points. Give 3 points for the correct answer, else 0 points.

   (b) Short answer:

   i. False. SISD has been and is still very common. It represents all single core processors with one instruction stream that operates on one stream of data.

   ii. True.

   iii. True.

   Max 3 points. One point for each correct answer, together with relevant explanations for the false answers.

   (c) Short answer: The instruction can perform 4 double precision floating-point number computations in parallel.

   Max 2 points. Two points for the correct answer.

Part II: Advanced

7. The complete solution is omitted. Please see the lecture slides and the course book.

   Max 15 points. For each of the three items, max three points for the explanations of the two concepts, max one point for at least one difference, and max one point for at least one similarity. That is, max five points for each of the three items.

8. (a) A potential solution may be as follows.

```
int foo(const int *p, int len){
  int c = 0;
  for(int i=0; i<len; i++)
    for(int j=0; j<i; j++)
      c += *(p + i*len + j);
  return c;
}
```

   Max 10 points. Two points if the function signature is completely correct. Four points for completely correct `for` loops, including variable updates and comparisons. Four points for completely correct updating and returning of counter, including pointer access, index multiplication etc. For each of these three parts, remove one point for each error (e.g. missing the `const` keyword or incorrect index or variable). Remove one point for each code line that is above 8 code lines.

   (b) The function will return integer value 66.

   Max 5 points. Five points for correct number and if it is possible to follow the solution reasonably. If only a correct answer, max 3 points. Max 2 points if the answer is incorrect, but it is clear that the solution is partially correct.

9. (a) First, we note that the function starts at a static address, which means that we can divide all instructions into 4 instruction blocks (not shown in this solution, but can be done on paper). For instance, the first block is between the first `addi` instruction and the `slt` instruction. We name this block 1. The code is covered by 5 blocks.

   By inspecting the meaning of the program, we can see that the input register value of `$a1` affects the number of loops. Value 0 gives the fewest loops, where the first `beq` instruction will jump to `exit` directly. This means that we will have misses only in block 1, 2, and 5. That is, 3 instruction cache misses. In case the input register `$a1` has value 4, we can easily see that all 5 blocks are used. Hence, in the best case, the number of instruction cache misses is 3, and in the worst case, 5.

   Max 4 points. 2 points for correct best-case number and 2 points for correct worst-case number.

(b) The best-case number of data cache misses is zero, because if the input register `$a1` is zero, the `lw` instruction is never executed.

The worst case is less obvious. The easiest way to solve the problem is to draw the memory and check the program which memory elements that are accessed. As in the case for instruction cache, it depends on input register value `$a1`, but also on pointer input value for input register `$a0`. From the constrains in the task, register `$a0` can be any number. However, since the access pattern repeats itself every 16 bytes (because of the block size), we only need to check these 16 alternatives. Note also that only word aligned offsets are valid inputs, which means that we only have to consider four alternatives: 1) where `$a0` is aligned with the first word in a cache block, 2) it is aligned with second word, etc.

If we study the access pattern of `lw` we can see that it forms triangular access pattern in the input matrix. This means that for alternatives 1 and 4, there will be 3 data cache misses, whereas for alternatives 2 and 3, there will be 4 cache misses. Hence, the best-case number of cache misses is 0, and the worst-case is 4.

Max 6 points. 2 points for correct best-case number and 4 points for correct worst-case number. Partial solutions may be given some points.

(c) The worst-case is when the input register `$a1` is 4, that is, most loops are executed. We examine each of the basic blocks in turn.

- Basic block `count`: This block will only be executed once and each of the two instructions take 1 cycle. Hence, the `count` block takes in total 2 cycles.
- Basic block `outer`: We can see that the outer loop will execute 4 times, but the outer basic block 5 times, where the fifth time, the branch will be taken. There is a data hazard between `slt` and `beq` that needs to be solved using stalling (one cycle). The branch instruction will be flushed just once, when it jumps. Hence, the number of cycles are $5 \cdot 4 + 1 = 21$.
- Basic block `inner`: There is one data hazard that needs to be soled using stalling (between `slt` and `beq`). This block will be executed different number of times each time it comes from the outer block. i) The first time it is entered from `outer` the branch is taken. Hence $2 + 1 + 1 = 4$ cycles. ii) The second time it is entered from `outer`, it will be executed twice, where the branch is taken the last time. Hence $2 * (2 + 1) + 1 = 7$ cycles. iii) The third time it is entered from `outer`, it will executed three times, where the branch is taken the last time. Hence $3*(2+1)+1 = 10$ cycles. iv) Same pattern: $4*(2+1)+1 = 13$. Hence, in total we have: $4 + 7 + 10 + 13 = 34$ cycles.
- Basic block `body`: By inspecting the flow of the program, we can see that the `body` will be executed exactly 6 times. There are 8 instructions and all instructions take 1 clock cycle each, with two exceptions: 1) There is a data dependency between `lw` and `add` that results in a hazard and one cycle stall, and 2) the `j` instruction needs to flush, adding one cycle. Hence, the total number of cycles are: $6 \cdot (8 + 1 + 1) = 60$ cycles.
- Basic block `skip`: This block will be executed 4 times. The two instructions takes one cycle each + 1 extra cycle because of the flush for instruction `j`. Hence, $4 \cdot (2 + 1) = 12$ cycles.

- Basic block `exit`: This block is only executed once. The `jr` instruction takes 6 cycles to complete (including finishing all stages in the pipeline, as specified in the exercise.

Hence, the number of cycles due to instructions are: $2 + 21 + 34 + 60 + 12 + 6 = 135$ cycles. From the previous exercise, we know that the worst-case number of instruction cache misses were 5, and the number of data cache misses 4. Hence, with a cache miss penalty of 10, we have $135 + 5 \cdot 10 + 4 \cdot 10 = 225$ clock cycles.

Max 10 points. For 10 points, the number must be correct, and actual reasoning clear. If the resulting number is incorrect, but the solution is partially correct, up to 6 points can be given, depending on how close the solution is.