RHODES UNIVERSITY
DEPARTMENT OF COMPUTER SCIENCE

EXAMINATION: NOVEMBER 2019

COMPUTER SCIENCE HONOURS
PAPER 2 – ARCHITECTURE

Internal Examiners: Prof P Machanick

MARKS: 120
DURATION: 4 hours

External Examiner: Prof Ian Sanders

GENERAL INSTRUCTIONS TO CANDIDATES

1. This paper consists of 5 pages, 5 questions and a sheet with login details. **PLEASE MAKE SURE THAT YOU HAVE A COMPLETE PAPER.**
2. Practical Exam conditions: open book, open access to Internet
3. Course notes at http://homes.cs.ru.ac.za/philip/Courses/CSHonsArch/ may be used
4. State all assumptions and show working for all numeric examples.

PLEASE DO NOT TURN OVER THIS PAGE UNTIL TOLD TO DO SO.

Question 1 – Instruction Set Design [8 + 6 = 14 Marks]

In each of the following your own views, as justified by facts you use, are important – I am not looking for a repeat of what I told you.

(a) The RISC-V design is claimed to learn from past mistakes. Discuss design differences from MIPS and use these to evaluate the claim that RISC-V has learnt from past mistakes.

(b) Explain why Intel dominates the desktop and server markets despite the known weaknesses of its instruction set design.

Question 2 – Memory and Quantitative Design [(2+2) + 2 + 4 + (4+2) = 16 Marks]

In each of the following take care to quantify your answer in the terms requested. Assume the following information as a baseline for each part of this question:

- 2GHz clock
- 1 instruction per clock in the absence of stalls
- L1 cache access takes 2 cycles but is fully pipelined
- L2 cache access takes 10 cycles but results in a stall
- DRAM access takes 100 cycles
- Instruction mix: 20% branches, 20% loads, 10% stores, remainder ALU

State any assumptions needed to answer (e.g. filling in missing detail).

(a) In the absence of any stalls:
   i. what is the time per instruction in *ns*?
   ii. what is the execution speed in clock cycles per instruction (CPI)?

(b) With the given DRAM parameters, for a cache block size of 32 bytes, how wide must the bus be to fit a miss to a DRAM into 200 cycles?

(c) If 1% of instructions result in a miss to L2 and 10% of L2 references result in a miss to DRAM, what is the average number of clock cycles per instruction (CPI) assuming all stalls arise from memory accesses?

(d) A design is proposed where blocks replaced from L2 are put in a *victim cache* and can be retrieved; blocks remain in the victim cache until it is full and the least recently accessed is finally evicted. This design results in L2 accesses taking 12 cycles, but reduces misses to DRAM to 5% of L2 accesses.
   i. Calculate CPI in this new scenario.
   ii. Contrast this new result in part (i) with your answer to Question 2(c).

Question 3 – Pipelines and ILP [(8+8) + 3 + (8+3) = 30 Marks]

For this question, use the following code in the machine code notation used in the course. The instruction set is as described in the notes (RISC-V with simplified register naming), as is the 5-stage pipeline to be used in this question.

```
# registers:
# R1: base address of a; a copy we can change
# R2: base address of b; a copy we can change
# R3: N; constant in this code
# R4: i; loop counter
# R5: holds value of a[i]
# R6: holds value of b[i]
# R7: temp

        li R4, 0          # for (int i = 0; i < N; i++) {
        j fortest         # test loop at end
forbody: lw R6, 0(R2)      #   if (b[i] % 2 != 0)
        ori R7, R6, 1     #   // will be 1 if b[1] is odd
        bzero R7, else
        li R5, 2          # a[i] = b[i] * 2
        mult R5, R6, R5
        j endif           # else
else:   move R5, R6       #   a[i] = b[i]
endif:  sw R5, 0(R1)
        addi R4, R4, 1    # increment loop counter
        addi R1, R1, 4    # increment base addresses by 4:
        addi R2, R2, 4    #   (word size in bytes)
fortest: blt R4, R3, forbody # }
```

(a) Draw a timing diagram (in the style of Figure 4.4, p 62 of the notes) and determine the time it takes to do two (2) iterations of the body of the loop, where the branch in the loop body (bzero R7, else) is ***taken in the first iteration*** and ***not in the second iteration*** – under the following assumptions:

i. Maximum number of stalls: assuming hardware interlock, no forwarding and a register value cannot be accessed until after the cycle when it is written – show the diagram and report the number of clock cycles.

ii. Redraw the timing diagram and give the total clock cycles now with maximum possible forwarding and explain how stalls can be eliminated, given these design parameters:

- pipeline registers update in the first half of a cycle and can be read in the second half
- backward branches predicted as taken; forward branches predicted as not taken
- events occur in the pipeline stage indicated in the 5-stage pipeline on p 52 of the notes

(b) Would a 2-level branch predictor make a difference in this case, versus the simple branch predictor of Question 3(a)(ii)? Explain.

(c) We now explore the benefit of loop unrolling. Assume we always do an even number of iterations.

  i. Unroll the loop once (two copies of the loop body) and ***only as necessary*** rename registers and reorder code to minimize dependences. Mark name dependences using the arrow notation used in the course (see p 69 of the notes).

  ii. How many stalls are removed in the unrolled version of the loop compared with that of Question 3(a)(ii) – aggressive forwarding, no rewriting?

Question 4 – Multiprocessors **[(4 + 4 + 6) + (3 + 3) + 10 = 30 Marks]**

In this question, you may use the latencies in Table 5.1, p 89 of the notes if you want to quantify any answers (note that this is not specifically required). A multicore design shares data through the L3 cache; updates have to go to L3 before they are seen elsewhere. A MESI protocol is used.

(a) A programmer has a loop initializing an array as follows:

```
for (int i = 0; i < N; i++)
  a[i] = calc (i); // init entries
```

  i. Explain what false sharing is.

  ii. Explain how the given loop could be split over two threads to minimize false sharing.

  iii. Write out two loops, one for each thread, illustrating your answer to part (ii).

(b) Explain whether a queue lock like M-lock or MCS lock as described in the notes will likely result in significant reduction in memory delays compared with a ticket lock. Answer separately for each of the following scenarios:

  i. A very short critical section that only has one instruction that updates a shared data structure.

  ii. A critical section in which numerous shared data structures are updated.

(c) Two cores attempt to acquire a spinlock on the same clock tick by executing an atomic swap (swap instruction). Outline the steps taken including bus transactions from the start of the first swap instruction that completes to the completion of the second swap instruction. ***Assume both cores at the start have the block in shared (S) state***.

Question 5 – Practical Question [30 Marks]

You are supplied with code that simulates a multilevel cache system. You may run the code, compile it and alter it if you need to. You are supplied with a Linux account for this question; use the given login details (good only for the exam). In this question I am looking for insights so there is not a fixed breakdown between questions: you may answer the parts where you can give the best answers in more detail than the rest.

(a) Look for a function in given source file `rawcache.c` called `rawCacheHit`.

  i. Explain in general terms without reference to the simulation how a hit in a cache is detected.

  ii. Explain how function `rawCacheHit` implements detecting a cache hit.

  iii. Explain how function `rawCacheHit` is used elsewhere in the code to implement an *N*-way associative cache.

(b) Run the given code with the simulation configured as follows (you will need to study the `README.md` file for how to do this):

- 2 levels of cache
  - first level
    - split I and D, each 16KiB, 32B blocks
    - I and D both direct-mapped
    - hit time
      - zero for D cache (absorbed by the pipeline)
      - 1 cycle for I cache
    - lookup time (to check tags) 1 cycle;
  - second level
    - 1MiB, 32B blocks
    - 2-way associative
    - hit time: 10 cycles
    - lookup time: 2 cycles
  - DRAM
    - Unlimited, no misses
    - Hit time: 100 cycles

  i. Report on statistics output by the code and explain their significance

  ii. Vary parameters to see what difference it makes if you use a bigger or smaller cache, vary the block size and change associativity.

  iii. Discuss what effect it would have on the numbers if you realistically adjust access times for higher associativity and larger block sizes.

(c) Comment on any aspects of the simulation in terms of whether it is correct, could be done better or how accurately it captures real execution time.

END OF EXAMINATION