CSM 61C Potpourri/Other

Spring 2020 Exam Question Compilation Solutions

This document is a PDF version of old exam questions by topic, ordered from least difficulty to greatest difficulty.

Questions:

- Spring 2018 Final Q13 - Potourri
- Fall 2019 Final Q10e, Q10f, Q10g
- Fall 2017 Final Q11 - Atomic Instructions
- Fall 2017 Final Q13 - Hamming Codes
- Summer 2018 Final Q12 - Hamming Codes

Problem 13 ***[F-5] Potpourri*** **(8 points)**

Answer the following questions

(a) You have a computer that, well, stinks. It goes down on average 6 times a day and it takes 1 hour to get working again. What is the current system's availability?

- ○ 0.5
- ○ 0.7
- ○ 0.6
- ● 0.8

Solution: Availability = MTTF/(MTTF+MTTR) = 4/(4+1) = 4/5 = 0.8

(b) Assume you have the computer from part (a) when the manufacturer offers you a deal. **a:** A new computer that only crashes 4 times per day or **b:** support that can reduce the time to fix to 6 minutes. Which one should you choose?

- ○ a
- ● b

Solution: Availability(a) = 6/(6+1) = 6/7
Availability(b) = 4/(4+.1) = 4/4.1
Availability(b) > Availability(a)

(c) You have a processor that has a clock rate of 2GHz, a time to poll of 200 cycles for I/O, and you need to poll I/O at 100 Hz. If you use polling, what is the **percentage** of time you will need to spend polling?

- ○ 1%
- ○ 0.01%
- ○ 0.1%
- ● 0.001%

Solution: Polling at 100Hz means 100 times per second. Each time you poll it costs 200 cycles. Thus, you spend $100 * 200 = 20,000$ cycles polling per second. This is $20,000/(2 * 10^9) = 0.001\%$

(d) If the data comes in very infrequently do you want to use interrupts or polling? Why?

- ● interrupts
- ○ polling

Solution: Interrupts because we would be wasting a lot of cycles polling for infrequent data.

e) Which of the following were discussed in Prof. David Patterson's lecture? (select all that apply)

■ The power demands of machine learning are growing 10x per year; Moore's Law is only 10x in 5 years.

☐ The Tensor Processing Unit has a similar performance per watt to a CPU on production applications.

TPU blows away CPU on performance/watt

☐ The marketplace has decided: Open ISAs (e.g., RISC-V) are better than proprietary ones (e.g., ARM).

Marketplace /will/ decide, it hasn't yet (as it has for RISC vs CISC)

☐ Domain Specific Architectures achieve incredible performance but just for one application, like ASICs.

For one /domain/, there are many applications in one domain

f) Which of the following were discussed in James Percy's GPU lecture? (select all that apply)

☐ A square is the base shape used when rendering scenes.

Triangle (since it's planar)

☐ The GPU achieves its speed because all of the threads run different programs on the same data.

The same program on different data (like map square over a list, or 3D project all these points)

☐ A GPU has many more cores than a CPU and operates at a higher frequency.

More cores, but lower frequency (otherwise we couldn't cool it!)

■ When pixels along polygon edges are different between new generations of GPUs, the team investigates it.

Yep, what was it that changed?

g) You have an SSD which can transfer data in 32-byte chunks at a rate of 64 MB/second. No transfer can be missed. If we have a 4GHz processor, which takes 200 cycles for a polling operation, what fraction of time does the processor spend polling the SSD drive for data? Leave your answer in the box provided as a percentage.

10%	**SHOW YOUR WORK** 64 MB/sec / 32 B/poll = 2M polls/sec 2M polls/sec * 200 cycles/poll = 400M cycles/sec 4 GHz clock ⇒ 4000M cycles/sec; 400M c/s / 4000M c/s = 10%

h) You are designing a 64-bit ISA for a simplified CPU with 3 bit-fields: `immediate | register | opcode`. You reserve enough of the rightmost bits to handle 1,500 opcodes, and enough of the leftmost bits to encode unsigned numbers up to 500 trillion. What's the greatest number of registers can you have?

16	**SHOW YOUR WORK** 1500 opcodes requires 11 bits (2048) 500 trillion requires 49 bits (512 Tebi) 64 - (11 + 49) = 4 bits for registers 2^4=16

Q11: Atomic Instructions

Shown below is a simplified implementation of a thread pool. The goal of a thread pool is to manage the execution of multiple threads; the goal of this specific implementation is to limit the total number of threads doing work at any given moment.

```
// There should never be more than 4 threads doing work at any given
// moment.
int counter = 4;

// This method is run by each thread.
void run_thread() {
  acquire(&counter);
  // Do work...
  release(&counter);
}
// Publicly visible method to queue up a new thread to be run.
void queue_thread() {
  // Assume that this method creates and executes a new thread.
  create_and_execute_thread(&run_thread);
}
```

Your task is to implement the acquire() and release() methods in RISC-V using `amoswap`, `lr`, and/or `sc`. **On your answer sheet, please complete the RISC-V code to implement both functions acquire() and release()**. You may not need all of the lines provided.

The expected behavior of the two methods is below. Note: **&counter is stored in register a0**.

`void acquire(int* c)`	Repeatedly checks the value of counter until it finds that counter > 0, then attempts to decrement the value of the counter. Restart if the decrement fails for any reason.
`void release(int* c)`	Repeatedly attempts to increment value of the counter until successful in incrementing

The specifications for these instructions are reproduced from the Green Sheet below:

`lr rd, (rs1)`	R[rd] = M[R[rs1]]; registers a reservation on the memory address R[rs1].
`sc rd, rs2, (rs1)`	If there is a reservation on the memory address R[rs1], then M[R[rs1]] = R[rs2] and R[rd] = 0; otherwise, R[rd] = 1.
`amoswap rd, rs1, rs2`	Atomically, R[rd] = M[R[rs1]] and M[R[rs1]] = R[rs2]

```
acquire:   lr   t0, (a0)
           beq t0, x0, acquire
           addi t0, t0, -1
           sc   t1, t0, (a0)
           bne  t1, x0, acquire
           jr   ra
```

```
release:   lr   t0, (a0)
           addi t0, t0, 1
           sc   t1, t0, (a0)
           bne  t1, x0, release
           jr   ra
```

Q13: KnocKnock

While working at KnockOff Corp., you are in charge of a "new and improved" Amazon's Alexa: KnockOff's Nicky. Your team will design a box to listen to user input, transmit it to KnockOff's servers, and process data. The Hamming code table is provided for your reference.

Bit	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Data	P1	P2	D1	P4	D2	D3	D4	P8	D5	D6	D7	D8	D9	D10	D11
P1	X		X		X		X		X		X		X		X
P2		X	X			X	X			X	X			X	X
P4				X	X	X	X					X	X	X	X
P8								X	X	X	X	X	X	X	X

1. Unfortunately the servers have a poor ComSat network connection and require error correcting codes to repair received packets. If compressed recordings are **34 bits**, how many bits are needed to **correct a single error** using SEC Hamming codes?

   A. 1 bit B. 2 bit C. 3 bits D. 4 bits E. 5 bits F. 6 bits G. More than 6 bits

   6 bits allow you to cover up to 57 data bits; 5 bits only give you 26 data bits)

2. To test your software, you use the following 7 bits (which include the correction code). What data bits are encoded by the following code word?

   7-bit Code Word: **0101101_2** Data Bits: 0101_2

   There was no corruption in the data bits originally

3. KnockOff's single server can't process millions of simultaneous requests. You decide to scale and find that the work comes in two parts: distributing and processing. Distribution cannot be parallelized and is 1% of the total work. Processing is the remainder. If you need an overall speedup of 50x, how many servers will you need to handle processing?

   A. 1 B. 50 C. 55 D. 93 E. 99 F. 101 G. Not possible

   1/(0.01 + 0.99/99) = 50

4. KnockOff would like your opinions on their dependable data storage on their server. Currently, the storage uses RAID 1 but you are suggesting to upgrade to RAID 5.

   a. First, name one inventor of RAID: Katz, Patterson, Gibson

   b. Assume we want to tolerate one disk failure. If a server has 3 disks (of equal capacity) filled with data, implementing RAID5 requires 1 additional disk(s), while RAID1 requires 3 additional disk(s).

   c. True/False: A single small write is faster with RAID5 than it is with RAID1.

      We still need to write to 2 disks but also need to perform 2 XOR operations according to the small-write algorithm.

   d. Small random reads have _____ throughput on RAID 5 than on RAID1

      A. higher B. lower C. about the same

   e. If a single disk fails, recovery time with RAID5 is ___ recovery time with RAID1.

      A. < B. = C. >

      RAID 5 requires XORing parity blocks with all other blocks used to compute the parity block. This also involves multiple reads and then writing the result. RAID 1 requires just 1 read and 1 write.

5. Knockoff also wants to improve their disk performance. Do each of the following improvements successfully decrease seek time:

   a. Decrease radius of the disk True / False
   b. Decrease number of platters True / False

6. KnockOff Labs experiments with replacing servers with a neural botnet of smartphones to process Nicky data. Currently, **1 million** (1,000,000) **servers** each consume **0.1 kW** and result in a **PUE of 2.0** . The smartphone setup will have a **PUE of 1.5**. with **1 million smartphones** that each consume **0.01 kW**. Electricity costs **$0.01/kWh.** Assume there are **10,000 hours/year**. What are the **cost savings** from using smartphones for a year?

   1,000,000 servers * 10,000 hours/year * 0.01 dollars/kWh *
   (2.0 * 0.1 kW/server - 1.5 * 0.01 kW/server) = $18,500,000 or 18.5 million dollars

Question 12: Go Ham[ming] (10 pts)

We will consider a system that works with 7-bit codewords encoded with Single Error Correction (SEC) parity bits. The Hamming code table is provided for your reference to the right.

Bit	1	2	3	4	5	6	7
Data	P1	P2	D1	P4	D2	D3	D4
P1	X		X		X		X
P2		X	X			X	X
P4				X	X	X	X

Suppose we read a codeword of 0b1010101 from our disk. What are the data bits that were encoded?
$p_1 = 0$, $p_2 = 0$, and $p_4 = 0$ so there is no error

0b 1101

Suppose two pesky alpha particles struck our disk (darn!) and changed the first two bits of our codeword so that now, we retrieve 0b**01**10101 from our disk. What data bits will be interpreted?
$p_1 = 1$, $p_2 = 1$, and $p_4 = 0$ so there is an error in the 1 + 2 = 3rd bit

0b 0101

What is the Hamming distance between two valid 7-bit SEC codewords? Hint: you can solve this with or without the previous two codewords.
SEC scheme codewords always have a Hamming distance of three. You could also compare the two previous valid and corrected codewords 0b1010101 and 0b0100101 which do have a Hamming distance of 3.

3

We will now add in a fourth parity bit $\mathbf{p_8}$ to allow our disk to also detect two errors (DED). We read 0b0110101$\mathbf{p_8}$ from our disk, where $\mathbf{p_8}$ is the DED parity bit for the 7-bit string. What **must** be the value of $\mathbf{p_8}$ such that our 8-bit codeword contains a single correctable error?
We know from the previous questions that 0b0110101 has a nonzero parity check. If there was a single error, then p_8 would have to be 1 in order for the parity of the entire 8-bit codeword to be 1 and confirm a single error.

$\mathbf{p_8}$ = ① ⓪

What is the Hamming distance between two valid 8-bit DED codewords?
You can compare any two valid 8-bit codewords (i.e. from above) or know that DED codewords have H.Dist of 4

4

Suppose we have a 5-disk RAID 3 system. Compared to a 5-disk RAID 5 system that is **byte striped**, fill in all the bubbles for the data request patterns / characteristics where RAID 5 **strictly outperforms** RAID 3.

Ⓐ long sequential reads
Ⓑ long sequential writes
Ⓒ recovering from 1 disk failure
Ⓓ small random reads
Ⓔ small random writes
Ⓕ capacity

Consider adding a DMA controller to facilitate bringing in data from our external disk/devices into main memory. The DMA controller can either make transfers in **burst mode**, where it stalls the CPU and writes 32B of data per clock cycle, or **cycle-stealing mode** where it doesn't stall the CPU (i.e. it runs normally) and writes 4B of data per clock cycle. The CPU returns to normal instruction execution after the transfer is completed. Which mode would be better for the following situations? Assume every instruction uses 1 clock cycle with no stalls.

- Dealing with page fault; the page size is 4KiB — Ⓐ burst / Ⓑ cycle stealing
- Playing a 200MB 5-minute video — Ⓐ burst / Ⓑ cycle stealing
- Processing 10 keyboard strokes per second — Ⓐ burst / Ⓑ cycle stealing