You can find the past exams here: https://cs61c.org/su22/resources/exams/

When posting questions, please reference the semester, exam, and question in this format so it's easier for students and staff to search for similar questions:

**Semester-Exam-Question Number**

For example: **SP19-Final-Q1**, or **SU19-MT2-Q3**

**Spring 2019 final walkthrough**

- Q1 Potpourri: https://youtu.be/FY5dAMrXvxo
- Q2 FSM: https://youtu.be/gmHbw6LSeSw
- Q3 C Coding: https://youtu.be/v4B1WTs5UNU
- Q4 RISC-V: https://youtu.be/2VHjG-gy9Dk
- Q5 Data-Level Parallelism: https://youtu.be/oG9Rrzmi0M4
- Q6 RAID and ECC: https://youtu.be/rfCNIzNZ2M
- Q7 Caches: https://youtu.be/xojc8YZaO3Q
- Q8 Spark: https://youtu.be/A37BFXRXmm0
- Q9 Datapath: https://youtu.be/q-T4N3hBhUM
- Q10 Digital Logic: https://youtu.be/3RI36lsDSg4
- Q11 Virtual Memory: https://youtu.be/5_2fKsK4I34

**SU19-MT1-Q2.** Why is it that `receive_buffer` is on the heap - isn't it a pointer / address to the actual value on the heap, and wouldn't it be in static, while `*receive_buffer`, the actual value, is in the heap. Is there a difference of notation, i.e. are they using the `&` symbol to refer to an address of a pointer, like in subquestion 3?

Also, if `receive_buffer` is a value stored in the heap, then why does the prompt say that all of the expressions evaluate to an address value?
Thank you!

Anonymous Ferret  3mth  #637ad
&receive_buffer is stored in static since its declared out of a local scope.

receiver_buffer evaluates to a memory address (It's a pointer!) and since we used malloc that memory address is located on the heap. Note even though we free the memory receive_buffer will continue to point to the same area in memory which is still the heap. Hope this helps!

*receive_buffer would evaluate to the first element which would be a character. Under the hood a character is an integer i.e. '2' is 50

Thus *receive_buffer would be in code.

All expressions definitely evaluate to addresses since there either pointers or some other variables (can also be pointers but it would be the address of the pointer not the address it points to) with the address of operator.

Hope this helps!

Leyla Zokhidova  3mth  #637ae
So, we pick which one each address would point to, i.e. receive_data points to an address in the heap. And &receive_data points to an area in static.

Thank you for the explanation, I think the wording of the question was just confusing.

Peyrin Kao  STAFF  3mth  #637af
_REPLYING TO Leyla Zokhidova
This question is a little different from the questions we normally ask, because it's asking about where the address in the variable points to, not where the variable is stored.

Leyla Zokhidova  3mth  #637ba
_REPLYING TO Peyrin Kao
Thanks for the clarification!

Anonymous Mongoose  3mth  #637aa  _Resolved_
Can anyone explain this? For the first picture, why in a power of 4, the digit 01 must occur exactly once? In the second picture, I don't understand why S3 = “1 followed by odd # of 0s”?

Peyrin Kao  STAFF  3mth  #637ab
In unsigned binary: 4 = 0b01 00. 16 = 0b01 00 00. 64 = 0b01 00 00 00. If you group the bits
into groups of two, there's always one group that's 0b01, and all the other groups are 0b00.

You can define the states in the FSM however you want. In this question, they define one state where you've received an odd number of zeros, and another state where you've received an even number of zeros, since only one of these corresponds to a power of 4, and you might switch between the two states as additional inputs arrive.

---

James Tang 3mth #637c  
https://inst.eecs.berkeley.edu/~cs61c/sp21/pdfs/exams/Fa17_Final_Solutions.pdf#page=7

Is this a typo? Why is the exam asking us to encode a "beq" operation as a S type instead of a B type?

---

Peyrin Kao STAFF 3mth #637d

I don't have access to course resources from this semester, but I agree that this looks like a possible typo.

---

Anonymous Camel 3mth #637b  
Su18 Midterm 1 Q1.4

I am a bit confused about the idea of two's complement. Why do we add 0b00...001 to the flipped bitstring instead of adding 1?

---

Peyrin Kao STAFF 3mth #637e

0b00...001 is 1 with a lot of zeros padded. It's the same thing.

---

Anonymous Hare 3mth #637a  
SP 18 Mt1 # 5

for b, why is it nor 8 * 3 = 24, since we have 3 register fields and each can store 8 registers

---

Peyrin Kao STAFF 3mth #637f

Each register field (rs1, rs2, rd) needs to uniquely identify one register. For example, if you wanted to encode xor a1, a1, a1, you need to be able to put the same value in all of rs1, rs2, and rd. Each register field is 3 bits, so you can have $2^3 = 8$ different registers.