You are viewing this thread in readonly mode.

[Final] Past Exams - 2023 #1044

Jero Wang STAFF
12 months ago in **Exam – Final**

3,214
VIEWS

You can find the past exams here: https://cs61c.org/fa23/resources/exams/. Please check the linked past Piazza/Ed Q&A PDFs first before asking here. Many of the questions are already answered in those! Video walkthroughs (if available), are also linked on that page!

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: **SP22-Final-Q1**, or **SU22-MT-Q3**

Anonymous Kookabura 12mth #1044babd ✓ Resolved

SP23-Final-Q4:

https://edstem.org/us/courses/43491/discussion/3970125?comment=9236636

I have the same question as the link above, but I still couldn't understand this - why is thrashing happening? I'm not sure why they have the same index (as for b we are indexing j and for a we are indexing i+j)

Andrew Liu STAFF 12mth #1044babe

Remember that cache addresses use a T:I:O breakdown to find the index — if you check the bit addresses, you'll find that the index field of the blocks end up being the same.

Anonymous Eagle 12mth #1044baba ✓ Resolved

Sp23-Final-Q2

Why would the Branch comparator not be on the critical path? During the execute stage, don't we need to "wait" for the branch comparator to correctly set the Asel, Bsel signals?

Nikhil Kandkur STAFF 12mth #1044babb

#1044abfc

Anonymous Wildcat 12mth #1044afaf ✓ Resolved

SP23-Final-Q7.12

Why do we need to include this line? I thought the calloc we did for codecopy initialized everything in codecopy to 0.

♡ ···

Jero Wang STAFF 12mth #1044afde

This `0` is used to "erase" the branches and jumps, which are set when you `memcpy` ed

♡ ···

Anonymous Armadillo 12mth #1044afae ✓ Resolved

SP23-Final-Q8.3

How is the answer 200 minutes if the task list is (chip A: 1,3) (chip B:0) (chip C:2,4). I'm getting 210 minutes for my calculations

♡ ···

Jero Wang STAFF 12mth #1044afdf

Could you show the math for 210? Here's what I'm getting:

Chip A does task 1 from t=0 to t=100, task 3 from t=100 to t=200 (task 0 is finished by t=50).

Chip B does task 0 from t=0 to t=50)

Chip C does task 2 from t=50 to t=110, task 4 from t=110 to t=170.

♡ ···

Anonymous Spider 12mth #1044afac ✓ Resolved

SP23-Final-Q8.3

For creating the task list, I don't remember how to come up with it. Are there any specific tips anyone has or any resources that review how to come up with the optimal task list?

♡ ···

Jero Wang STAFF 12mth #1044afea

There's a homework question that's similar, but in general, I'd recommend just going with greedy as a starting point, then try to make some fine adjustments based on what you come up with!

♡ ···

Jiries Kaileh 12mth #1044aeed ✓ Resolved

Just a general question, does the following equation hold for finding the local miss rate for a cache level?

LNC = Level n cache; LMR = local miss rate; GMR = global miss rate

LNC LMR = LNC GMR * (1 / L(N-1)C LMR) * (1 / L(N-2)C LMR) * ... * (1 / L(2)C LMR) * (1 / L(1)C LMR)

♡ ···

Andrew Liu STAFF 12mth #1044baca

This looks right to me; another way to think about it is that the LMR is just (# hits to cache level) / (# accesses to cache level), and you only access a cache if *all* previous cache levels have missed.

Anonymous Monkey 12mth #1044aeea ✓ Resolved

SP23-Final-Q2.1

Why do we only include stage 2 in the critical path for lw? In the discussion this was the full path:

$$\begin{aligned}&\text{lw} = \text{clk-to-Q} + \text{Mem-Read} + \max(\text{RegFileRead} + \text{Mux(ASel)}, \text{Imm-Gen} + \\ &\text{Mux(BSel)}) + \text{ALU} + \text{Mem-Read} + \text{Mux(WBSel)} + \text{RegFileSetup} \\ &= 5\text{ns} + 300\text{ns} + 60\text{ns} + 100\text{ns} + 300\text{ns} + 15\text{ns} + 20\text{ns} = 800\text{ns}\end{aligned}$$

How does switching around the stages decrease the critical path? Why do we not include RegFile Read over ImmGen if it is greater, and why do we only include the MemRead once? The solutions say that lw instructions only goes through stage 2, but it also needs to go through IF and ID. In this question, I am not sure how splitting the stages into 2 informs in that there will be two different critical paths from each one, when for 5 stages, we take the critical path of the whole circuit

Anonymous Monkey 12mth #1044aefa

Nvm, I didn't know that only the longest stage matters for a pipelined circuit. Is this information that was taught in lecture?

Anonymous Capybara 12mth #1044aeff

could you explain why only the longest stage matters?

Sonika Vuyyuru STAFF 12mth #1044afaa

Yes, this should be covered in lecture! It is also explained in question 3 of Discussion 8. I would suggest revisiting that for some more practice.

The main idea here is that when we pipeline the datapath, the longest stage constrains what we can set the clock time to be. The reason for this is that the same clock drives each of the stages, so even if another stage (like IF) takes less time, it needs to wait until the longest stage has completed it's execution to be able to move on to the next clock cycle.

Anonymous Spider 12mth #1044aedc ✓ Resolved

SU23-Final-Q1.5 Why during a thread context switch, the entries of the TLB get invalidated? Do threads in a process share same memory space? If true, then why context switch here should invalidate TLB?

Jero Wang STAFF 12mth #1044afdd

Yes, they do share the same memory space. In the real world, this is implementation specific - you can optimize it such that you don't need to flush if its the same memory space. It's just that this was how it was taught in the summer (verbatim from slides).

♡ ···

Anonymous Cod 12mth #1044aecb ✓ Resolved

SU23-Final-Q4.3

Where did we get the 3^3 from? Is it supposed to be 3^2?

♡ 1 ···

Nikhil Kandkur STAFF 12mth #1044aece

#1044cc

♡ ···

Olivia Xie 12mth #1044aebf ✓ Resolved

SU23-Final-Q2

For part 1, would it be a valid solution if we wrote xori t1 a0 1 for line 4 and mv a0 t1 for line 7?

♡ ···

Jero Wang STAFF 12mth #1044afbf

No, this would not clear the upper 31 bits of your answer, so your return value would not be 0 or 1.

♡ 1 ···

Anonymous Seal 12mth #1044aebd ✓ Resolved

SP23-Final-Q3.4

is it possible to do `&matrix[i][j]` instead of `matrix[i] + j` since that will also give me the pointer to the element at (i, j).

♡ 2 ···

Nikhil Kandkur STAFF 12mth #1044aecc

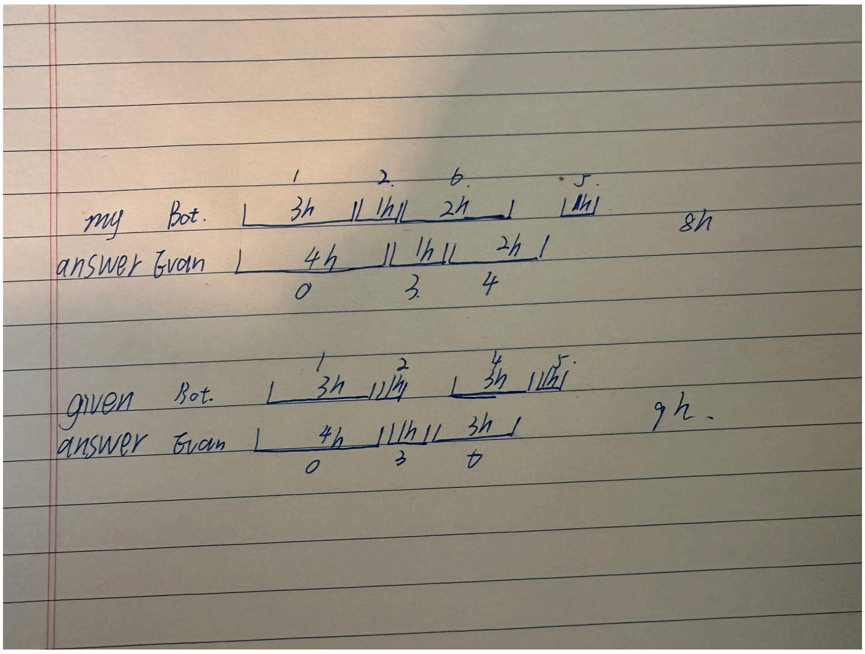
Yup that should work!

♡ ···

Anonymous Cheetah 12mth #1044aeba ✓ Resolved

su23-Final-Q7.3

the answer says the shortest time is 9 hours, but I think it could take just 8 hours to finish the tasks. Is there anything wrong with my answer?



Anonymous Ferret 12mth #1044aedd

It takes EvanBot 3 hours to complete task 4.

Anonymous Cheetah 12mth #1044aede

Thanks, I just realized that lol

Anonymous Armadillo 12mth #1044aeab ✓ Resolved

SP23-Final-Q1.1

Why does readArr ALU calculate rs1+rs2*4. especially on the rs2*4 part. I'm confused on the logic behind this.

Anonymous Chamois 12mth #1044aeef

Integers are 4b wide, so we need to move it by 4 times.

S Sonika Vuyyuru STAFF 12mth #1044aefe

Chamois has the right idea here. We want to read the byte at index rs2. 4 * rs2 tells us how many bytes into the array we need to move to read the element at index rs2, since this is an array of integers and each integer is 4 bytes. We then add this to the pointer address given by rs1.

Anonymous Grasshopper 12mth #1044adff ✓ Resolved

SU23-Final-Q3.13

I am confused why branch instrcution don't have data hazards? if we `add t0,t1,t2` then `bge t0,t1,loop` , this t0 depends on add instructions WB from previous.

1

S Sonika Vuyyuru STAFF 12mth #1044aefd

This question asks what type of hazard can be *caused* by the long branch instruction. Since the branch instruction does not write back to an rd register value, it does not cause any data hazards in future lines. In the example you provided, we could say the add instruction causes a data hazard, which affects the next line, which happens to be a branch instruction.

♡ 1 ···

Anonymous Grasshopper 12mth #1044afad

got it!

♡ 1 ···

Anonymous Grasshopper 12mth #1044adfa ✓ Resolved

Su23-Final-Q2

is it okay for write 14 instead of 0b1110?

♡ ···

Anonymous Cod 12mth #1044aeca

I have the same questionn

♡ ···

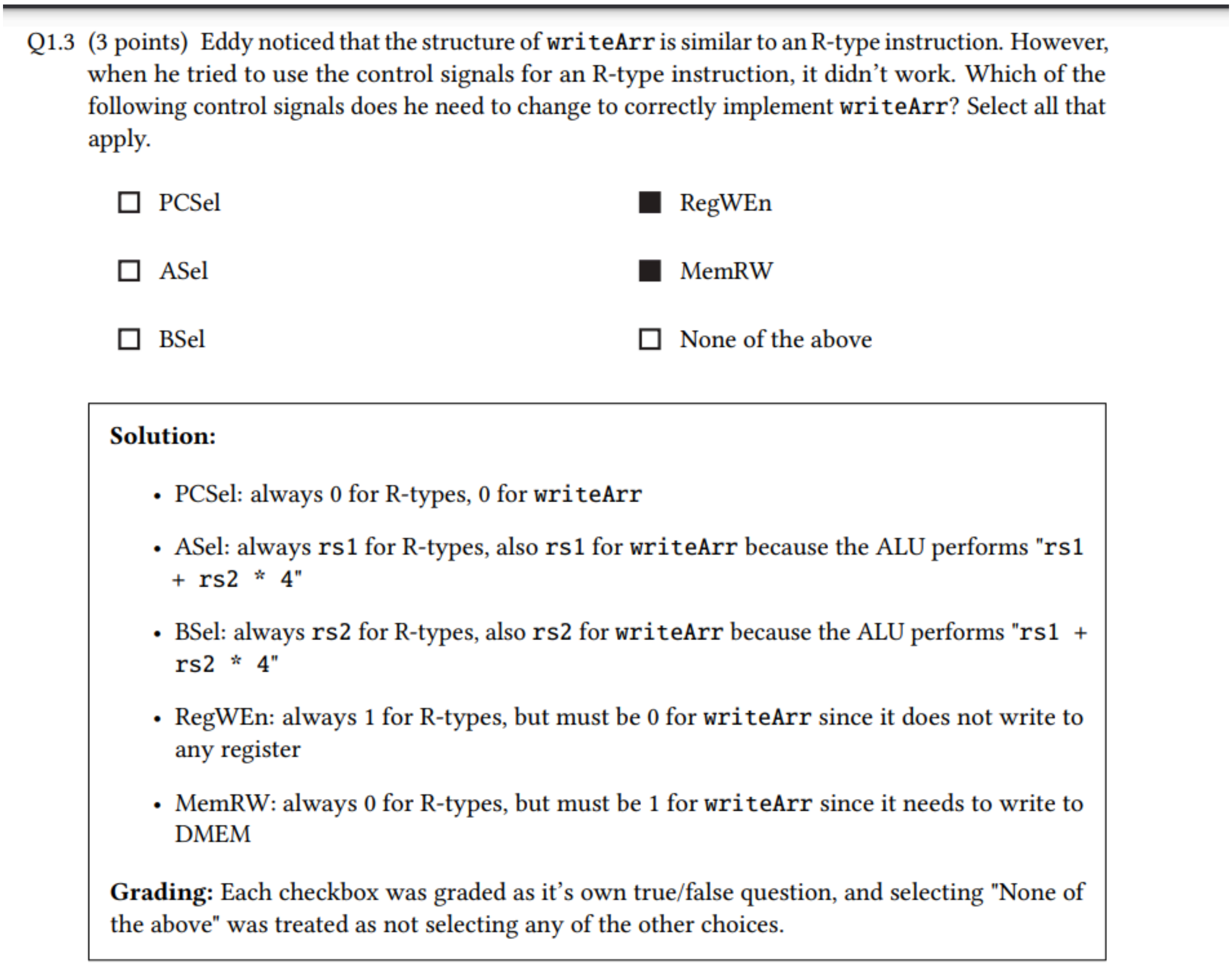
Sonika Vuyyuru STAFF 12mth #1044aefb

Yes, that should be fine.

♡ ···

Anshul Verma 12mth #1044addf ✓ Resolved

sp23-final-q1.3



Q1.3 (3 points) Eddy noticed that the structure of `writeArr` is similar to an R-type instruction. However, when he tried to use the control signals for an R-type instruction, it didn't work. Which of the following control signals does he need to change to correctly implement `writeArr`? Select all that apply.

□ PCSel

□ ASel

□ BSel

■ RegWEn

■ MemRW

□ None of the above

Solution:

- PCSel: always 0 for R-types, 0 for `writeArr`
- ASel: always `rs1` for R-types, also `rs1` for `writeArr` because the ALU performs "`rs1 + rs2 * 4`"
- BSel: always `rs2` for R-types, also `rs2` for `writeArr` because the ALU performs "`rs1 + rs2 * 4`"
- RegWEn: always 1 for R-types, but must be 0 for `writeArr` since it does not write to any register
- MemRW: always 0 for R-types, but must be 1 for `writeArr` since it needs to write to DMEM

Grading: Each checkbox was graded as it's own true/false question, and selecting "None of the above" was treated as not selecting any of the other choices.

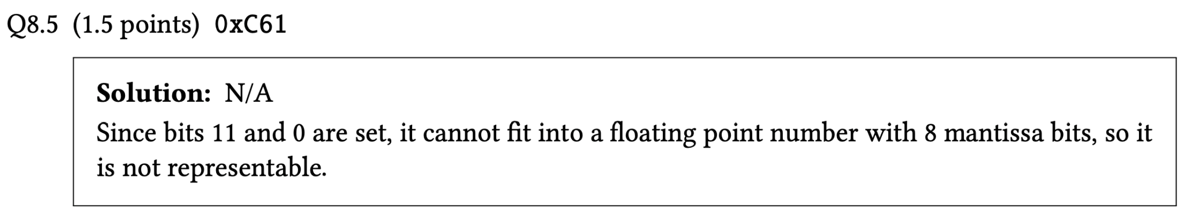
Hi, for this question why does RegWEn need to be changed? I figured that since writeArr does not write back anything to the registers that RegWEn would just not matter.

Robert Yang 12mth #1044adeb

I think the output of the writeback mux is always sent to the register file so if you don't disable RegWEn, rs3 will wrongly be changed.

Anonymous Magpie 12mth #1044adda ✓ Resolved

What does "Since bits 11 and 0 are set" mean? And how does this correlate to the answer being N/A?



Q8.5 (1.5 points) 0xC61

Solution: N/A
Since bits 11 and 0 are set, it cannot fit into a floating point number with 8 mantissa bits, so it is not representable.

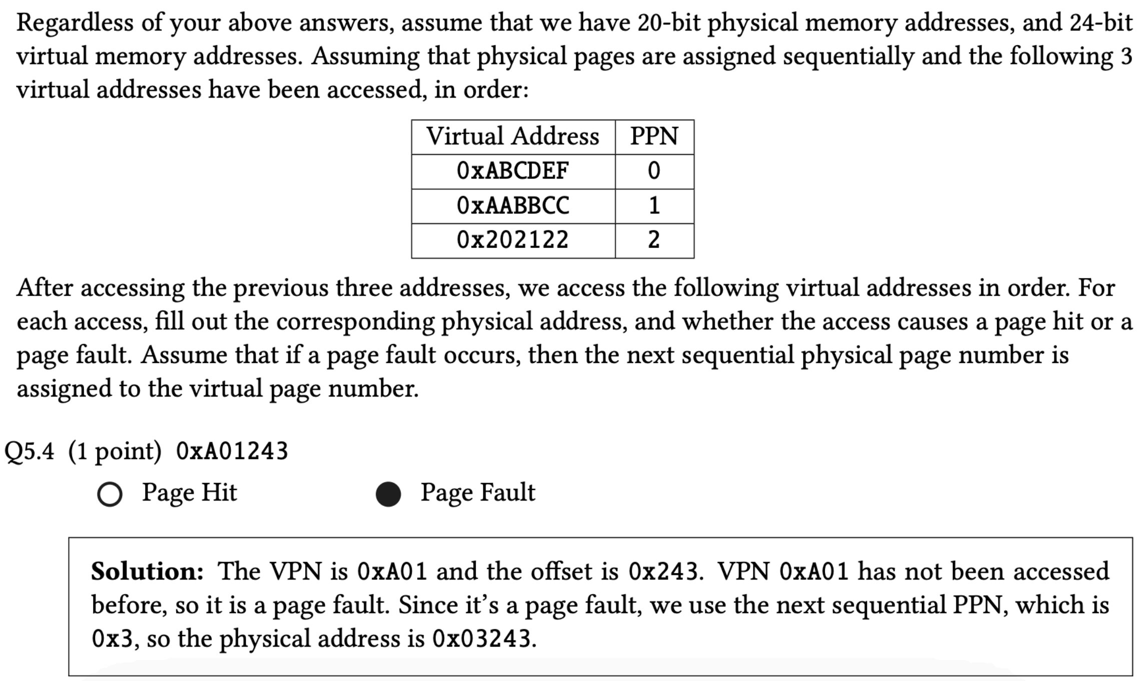
Andrew Liu STAFF 12mth #1044aead

That sentence means that bits 11 and 0 being set to 1. Remember that a floating point representation can only have as much precision as the bit width of its mantissa.

In this case, the precision needed is 11 bits, but the width of the mantissa is 8 bits, so this address is not représentable.

Anonymous Rhinoceros 12mth #1044adcf ✓ Resolved

SU23-Final-Q5



Regardless of your above answers, assume that we have 20-bit physical memory addresses, and 24-bit virtual memory addresses. Assuming that physical pages are assigned sequentially and the following 3 virtual addresses have been accessed, in order:

Virtual Address	PPN
0xABCDEF	0
0xAABBCC	1
0x202122	2

After accessing the previous three addresses, we access the following virtual addresses in order. For each access, fill out the corresponding physical address, and whether the access causes a page hit or a page fault. Assume that if a page fault occurs, then the next sequential physical page number is assigned to the virtual page number.

Q5.4 (1 point) 0xA01243

○ Page Hit ● Page Fault

Solution: The VPN is 0xA01 and the offset is 0x243. VPN 0xA01 has not been accessed before, so it is a page fault. Since it's a page fault, we use the next sequential PPN, which is 0x3, so the physical address is 0x03243.

Hi! For 5.4 to 5.9 we are only given this, how should we figure out the offset bit and VPN? I get how it usually work but in this case do we have enough info to figure out? I am so confused Thank you!

Andy Chen STAFF 12mth #1044aeac

Yeah I think there actually isn't enough information here, I think the question is assuming that we are using the same page size as earlier in the question (which was 4 KiB = 2^{12} Bytes). I think the intent was to use that to figure out the offset, and then use the given physical address length and virtual address length to find out VPN/PPN.

♡ 2

Anonymous Quail 12mth #1044adaf ✓ Resolved

SP23-Final-Q1

The solution in 1.1 says: "The readArr instruction is similar to any other R-type instruction, except that the ALU operation performed must be rs1 + rs2 * 4."

Am I right in thinking that it is different from the R type instructions not only because of the new ALU operation, but also because we would have to read from memory, so the WBSel would be different? I get that we don't have to add anything new to the data path for that part, but just wanted to make sure that my thought process is correct.

I just got a bit confused because to me the readARR instruction is more similar to some kind of load instruction, with the difference being that it uses a value stored in a register instead of an immediate, rather than to an R-type instruction

Robert Yang 12mth #1044aded

I completely agree with your last paragraph. I think the exam says readArr is similar to an R-type instruction just because R-type instructions take in 3 registers as input, and so does readArr. So load and R-types are both similar to readArr in different ways.

♡ 2

Anonymous Gerbil 12mth #1044acfc ✓ Resolved

SP23-Final-Q2.3

How come Immediate is not an answer? Don't we need it in stage 2 as an input to the BSel Mux?

Nvm answered in #1044ea

Anonymous Cobra 12mth #1044aced ✓ Resolved

SU23-Final-Q6

Could someone explain line 3 in the solutions? I get the "#pragma omp parallel for" part, but what does the "reduction (+: count)" do?

Anonymous Mandrill 12mth #1044aeaa

declares count as critical variable so no two threads modify at same time

Andrew Liu STAFF 12mth #1044aeae

The reduction introduces a privatized version of the variable that is collected via the + operator. Check out the OpenMP lab spec for some more details.

Anonymous Armadillo 12mth #1044acdd ✓ Resolved

SP23-Final-Q5.3

Why is this a page fault? I thought the index in the page table shows that at address 0x3, the PPN is CBA8 so the answer would be CBA860

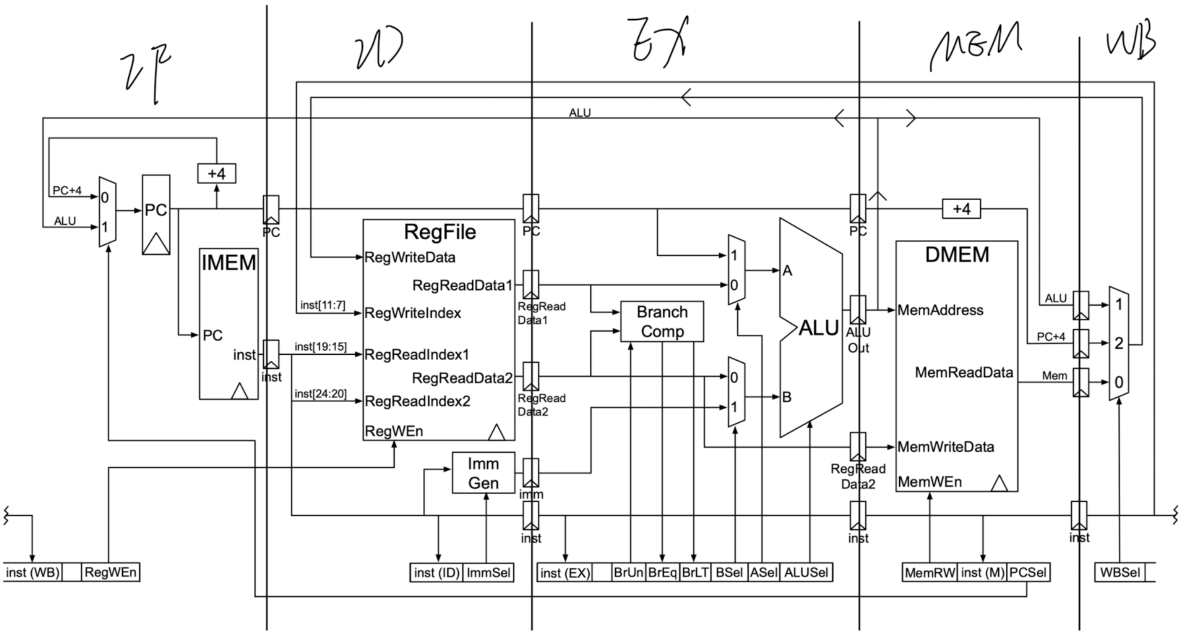
Anonymous Cobra 12mth #1044acde

PTE 0x3561CBA8 is not valid, since 0x3 = 0b0011 (valid bit is 0). So we use the next free physical address in main memory.

Anonymous Cheetah 12mth #1044acdc ✓ Resolved

SP-23-Final-Q2.1

Why do we include ImmGen in the 2nd stage? I think from the reference card ImmGen belongs to ID, which is included in the 1st stage in the question.

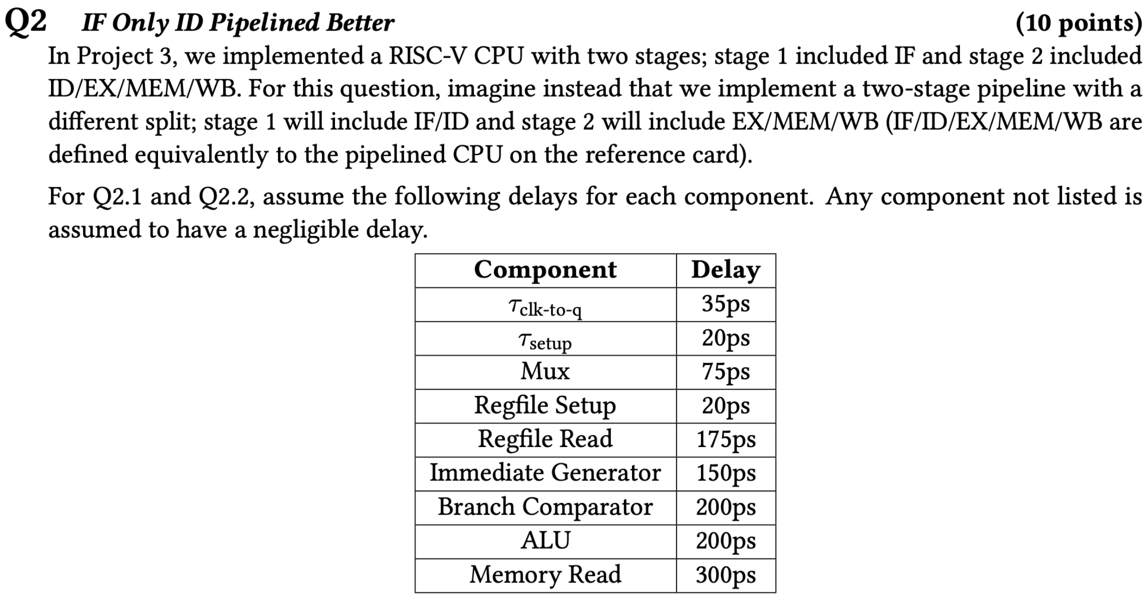


4

Anonymous Goldfinch 12mth #1044acef

SP-23-Final-Q2.1

I have same question here.



Q2 ***IF Only ID Pipelined Better*** **(10 points)**

In Project 3, we implemented a RISC-V CPU with two stages; stage 1 included IF and stage 2 included ID/EX/MEM/WB. For this question, imagine instead that we implement a two-stage pipeline with a different split; stage 1 will include IF/ID and stage 2 will include EX/MEM/WB (IF/ID/EX/MEM/WB are defined equivalently to the pipelined CPU on the reference card).

For Q2.1 and Q2.2, assume the following delays for each component. Any component not listed is assumed to have a negligible delay.

Component	Delay
$\tau_{\text{clk-to-q}}$	35ps
τ_{setup}	20ps
Mux	75ps
Regfile Setup	20ps
Regfile Read	175ps
Immediate Generator	150ps
Branch Comparator	200ps
ALU	200ps
Memory Read	300ps

Question tells us that the datapath is split into two stages: stage 1 = IF/ID and stage 2 = EX/MEM/WB. So Imm generator shouldn't be included in the critical path right?

I'm also confused why DMEM is included. Isn't DMEM a type of register itself?

♡ ···

Martin Lacsamana 12mth #1044acfe

If I'm not mistaken, they moved ImmGen into the ID stage this semester, so our answers would differ now. So yes, the ImmGen should be in stage 1.

♡ 3 ···

Nikhil Kandkur STAFF 12mth #1044aecf

#1044bb

♡ ···

Anonymous Magpie 12mth #1044acbd ✓ Resolved

SU23-Final-Q8.1

Through trial and error knowing that our exponent post-bias needs to be at least 6 (since 2^6 = 64), I figured out that we needed 4 exponent bits pre-bias. I just wanted to clarify if there was an easy intuition to see it being 4 bits or if we had to calculate the exp and bias equations to see which number of bits is at least 6.

♡ 2 ···

Anonymous Chimpanzee 12mth #1044accf

I had the same question and approach

♡ ···

Jero Wang STAFF 12mth #1044afbb

This was mostly trial and error for getting the exact number of exponent bits, since the bias depends on the number of exponent bits.

♡ ···

Anonymous Rail 12mth #1044acaf ✓ Resolved

SP23-Final-Q4.5

What is the formular we use for the different level of caches and is there any comprehensive resource with all the formulas?

Anonymous Kudu 12mth #1044accd

last video of caches has a good example

Anonymous Llama 12mth #1044abfc ✓ Resolved

SP-23-Final-Q2.1

Why do we not include the Branch Comparator Delay into stage2 calculation? Could you please write out the equation to get the correct answer based on this sem's Datapath. I would also appreciate if you list the components for each stage.

Robert Yang 12mth #1044acae ENDORSED

I believe it's because the instruction that takes the longest is a load instruction, which doesn't use the branch comparator. You can also compute how long a branch instruction will take which of course does use the branch comparator, but this turns out to be less than the load instruction time.

Anonymous Quail 12mth #1044adbf

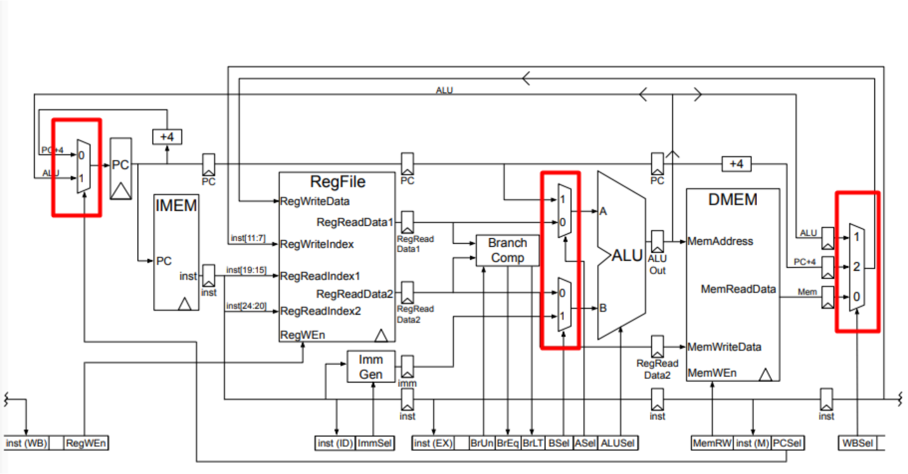
If it's still relevant to you, my calculation ended up being:
35 + 75 + 200 + 300 + 75 + 20 = 705

1

Robert Yang 12mth #1044abfb ✓ Resolved

SP23-Final-Q2.1

The calculation adds up 2 muxes but aren't there 3 muxes that we must go through in stage 2? See my screenshot below

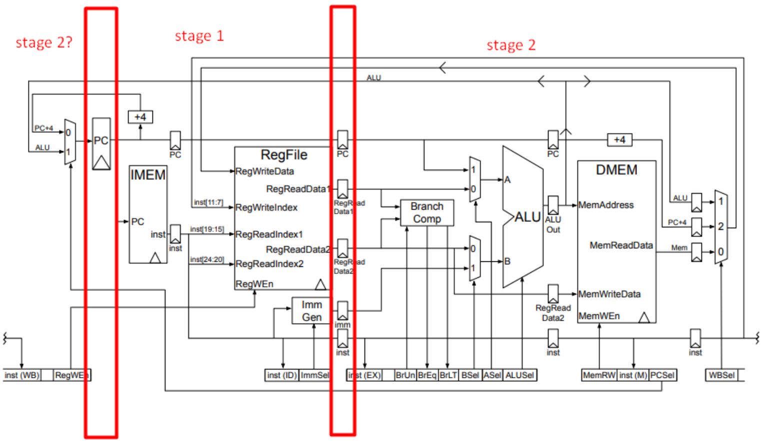


Anonymous Rail 12mth #1044acaa

This mux that does in PC is part of stage 1 I believe (part of the IF stage)

Robert Yang 12mth #1044acad

Hmm I thought everything has to be stored within a register between stages, so stage 1 begins after the first PC register which leaves the PC mux in stage 2. See the image below:



♡ 1 •••

Anonymous Cobra 12mth #1044acec

Replying to Robert Yang

The MUX feeding into PC is part of stage 2, but if you look closely at the paths, its input is the ALU, not the WB MUX (which feeds into RegFile). Thus the longest path has two MUXes.

♡ 1 •••

R

Robert Yang 12mth #1044acee

Replying to Anonymous Cobra

HOLY CRAP YOU'RE A GENIUS MAN/WOMAN THANK YOU SO MUCH this makes sense

♡ 2 •••

Anonymous Lion 12mth #1044abef ✓ Resolved

SP23-Final-Q5.4

Why is this a page fault? I thought that the page table was indexed by VPN and the VPN is 0x000003 which has a corresponding entry in the page table?

♡ 1 •••

Anonymous Gnat 12mth #1044abfe

Also confused on this same part ^

♡ •••

This comment was deleted

Anonymous Lion 12mth #1044acac

Wait to clarify I'm not confused about the TLB miss part but how it's a page fault. Page fault means that there's no entry for it in the page table but there is?

♡ •••

Anonymous Kudu 12mth #1044accc

Replying to Anonymous Lion

look at the valid bit (31/30) of the value in the page table - it's 0.

♡ 1 ···

Anonymous Cobra 12mth #1044aceb

#1044acde

♡ 1 ···

Anonymous Kudu 12mth #1044abee ✓ Resolved

SU23-Final-Q3.8/9

Aren't there 2 implicit zeroes since it can only jump by multiples of zero, as shown in the lecture slides as well? 2^11 and 2^19

♡ ···

J Jero Wang STAFF 12mth #1044afca

Assuming you mean multiples of 4, no, you can jump in multiples of 2 (since RV32C instructions are 2 bytes), so there's one implicit 0.

♡ ···

Anonymous Gerbil 12mth #1044abed ✓ Resolved

SP23-Final-Q6.3

Is 0b110 a valid bit pattern for 0b0?

♡ 1 ···

Anonymous Cod 12mth #1044acbc ENDORSED

I don't think so because the first parity bit and the data bit would sum to 1.

♡ ···

Anonymous Kudu 12mth #1044acce

no cause if it messes up a bit it can go into the region for 000, like 010 or 100, so it won't know if its 0 or 1

♡ ···

Anonymous Zebra 12mth #1044abec ✓ Resolved

SP23-Final-Q5.2

For this question, the solution says that the page table entry has the PPN in its last four bits. Is there a specific reason the PPN is in the last four bits? Is the location of the PPN specific to this question, or will the PPN always be in the final four bits of the page table entry?

♡ ···

Anonymous Kudu 12mth #1044acda

bits 0 to 16 have the ppn and typically the ppn always starts from the last n (16 in this case) bits of the PTE

♡ 1 ···

Anonymous Cod 12mth #1044abeb ✓ Resolved

SP23-Final-Q4

Just to make sure I'm understanding this conceptually, but if say array a is of length 20, then would that result in a taking up two cache lines (one with index and one with index 1, but both with the same tag)? Because then we would also need to access address 0x1010 etc?

♡ 1 ···

Anonymous Chamois 12mth #1044aeee

I think you need more than 2 blocks - seems to be 4? One block is 16b, so it can contain 4 integers (4b each). If we have 20, then we need 4 blocks. However, we know there can only be 4 blocks (64b / 16b), so that's the max.

♡ ···

J Jero Wang STAFF 12mth #1044afec

20 elements or bytes? If its the former, then as Anon Chamois mentioned above, we need 5 blocks, which doesn't fit in our cache. If it's the latter, yes, it takes 2 cache lines.

If it's the latter, both would have the same tag, but different index.

♡ 1 ···

Anonymous Llama 12mth #1044abdf ✓ Resolved

SP23-Final-Q5

Conceptual: When given a Virtual Address, what happens if:

1) when looking in TLB valid bit is 1 vs 0

2) when looking in TLB dirty bit is 1 vs 0

3) if TLB miss, page fault -> do you update TLB and PT, if so, how?

♡ ···

Anonymous Kudu 12mth #1044acdb ENDORSED

assuming u are doing read operations:

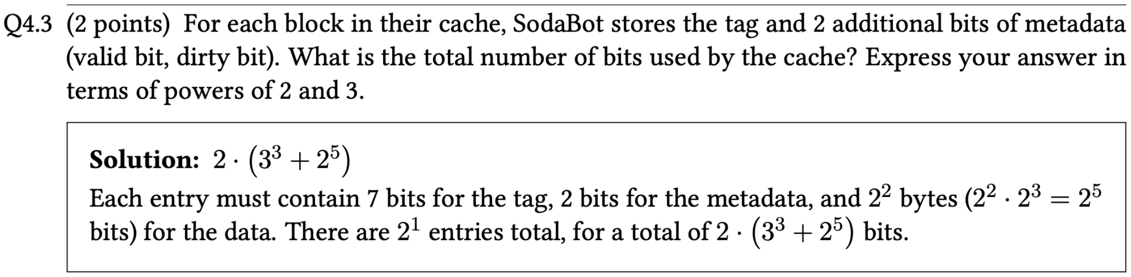
1) 0 causes a page fault (I believe)

2) read operations wont care abt the dirty bit, write ones will evict it from DMEM to disk if dirty bit is 1

3) Yeah, you first update the PT and then copy that over to the TLB

♡ 1 ···

Anonymous Squid 12mth #1044abda ✓ Resolved



Q4.3 (2 points) For each block in their cache, SodaBot stores the tag and 2 additional bits of metadata (valid bit, dirty bit). What is the total number of bits used by the cache? Express your answer in terms of powers of 2 and 3.

Solution: $2 \cdot \left(3^3 + 2^5\right)$

Each entry must contain 7 bits for the tag, 2 bits for the metadata, and 2^2 bytes ($2^2 \cdot 2^3 = 2^5$ bits) for the data. There are 2^1 entries total, for a total of $2 \cdot \left(3^3 + 2^5\right)$ bits.

Can someone explain where 3^3 and 2^5 came from? I understand there is 2 entries since there is only 1 bit for index. there are 7 tag bits +2 metadata bits so this is 9 bits also there are 2 offset bits.

♡ ···

Nikhil Kandkur STAFF 12mth #1044aedb

#1044cc

♡ ···

Anonymous Chimpanzee 12mth #1044abbb ✓ Resolved

SU23-Final-Q8.1

Can someone explain why the answer is 4 bits? I thought we need 6 bits to represent 0-63

♡ ···

Nikhil Kandkur STAFF 12mth #1044abdc

#1044a

♡ ···

Anonymous Manatee 12mth #1044abac ✓ Resolved

SU23-Final-Q8.5-8.7

I don't quite get the explanation for 8.5 and or why we know that the lower 6 bits of the address is the chunk and the chunk address is the upper 6 bits. Can we get some help on that? Thank you!

♡ ···

Suvan Ravi 12mth #1044abad

The lower six is like the fraction of the chunk. Since anything less than multiples of 64 will be byte addresses inside the chunk. And the upper six bits become the whole number chunk addresses since they're multiples of 64.

♡ ···

Anonymous Jay 12mth #1044abaa ✓ Resolved

SU23-Final-Q6

I'm somewhat confused on how this code correctly checks for palindromes based on the implementation in the solution. If we set left = 0 and right = width - 4, let's say we had a 10 integer array as such: [1,8,7,6,4,5,6,7,8,1], then left = 0 and right = 6, and the while loop would check 1,8,7,6 and the reverse of 6,7,8,1, confirm that they are palindromes, and set left = 4 and right = 2, thus failing the condition that left <= right, while maintaining that is_palindrome is true. This inner while loop would then not check our middle 2 elements, which make the array not a palindrome, and would instead increment our count variable by 1. I'm not sure if there's something that I'm missing here? Could someone please explain the solution to the code?

♡ 1 ···

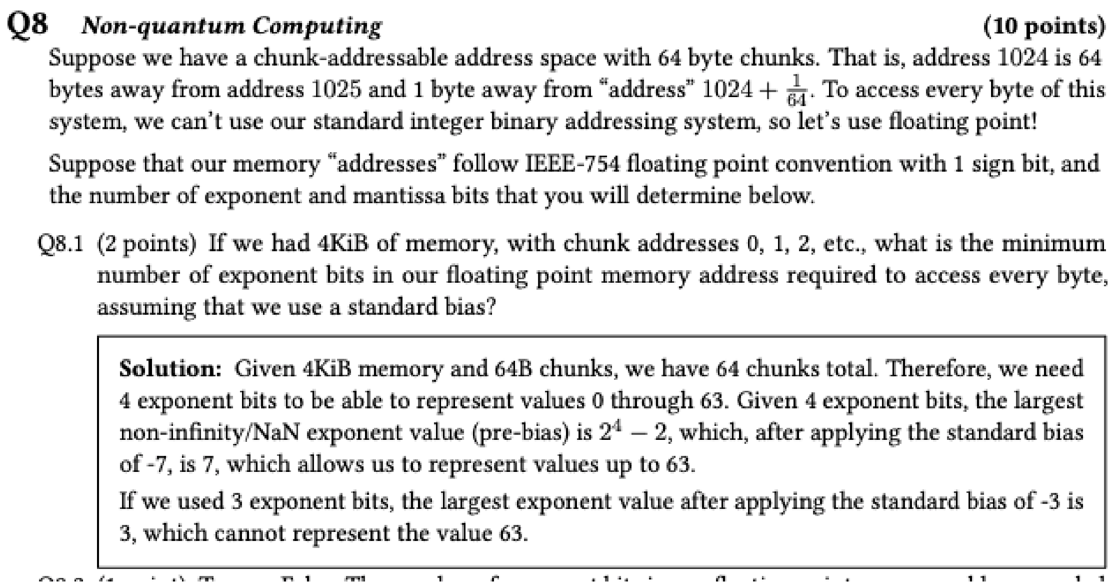
Anonymous Lyrebird 12mth #1044aeaf ENDORSED

i had the same question. originally i made right point to the end of the right array instead of the beginning and then just subtracted 4 from that pointer value whenever vec2 was loaded. i am not sure if this has any bugs but i think this would account for all the values in the array.

♡ ···

Anonymous Emu 12mth #1044aafd ✓ Resolved

[SU 23 Q 8] For question 8.1 how do we get the standard bias of -7 or -3?



Q8 *Non-quantum Computing* **(10 points)**

Suppose we have a chunk-addressable address space with 64 byte chunks. That is, address 1024 is 64 bytes away from address 1025 and 1 byte away from "address" $1024 + \frac{1}{64}$. To access every byte of this system, we can't use our standard integer binary addressing system, so let's use floating point!

Suppose that our memory "addresses" follow IEEE-754 floating point convention with 1 sign bit, and the number of exponent and mantissa bits that you will determine below.

Q8.1 (2 points) If we had 4KiB of memory, with chunk addresses 0, 1, 2, etc., what is the minimum number of exponent bits in our floating point memory address required to access every byte, assuming that we use a standard bias?

Solution: Given 4KiB memory and 64B chunks, we have 64 chunks total. Therefore, we need 4 exponent bits to be able to represent values 0 through 63. Given 4 exponent bits, the largest non-infinity/NaN exponent value (pre-bias) is $2^4 - 2$, which, after applying the standard bias of -7, is 7, which allows us to represent values up to 63.

If we used 3 exponent bits, the largest exponent value after applying the standard bias of -3 is 3, which cannot represent the value 63.

Anonymous Quetzal 12mth #1044abff

im also confused

Anonymous Magpie 12mth #1044acbb

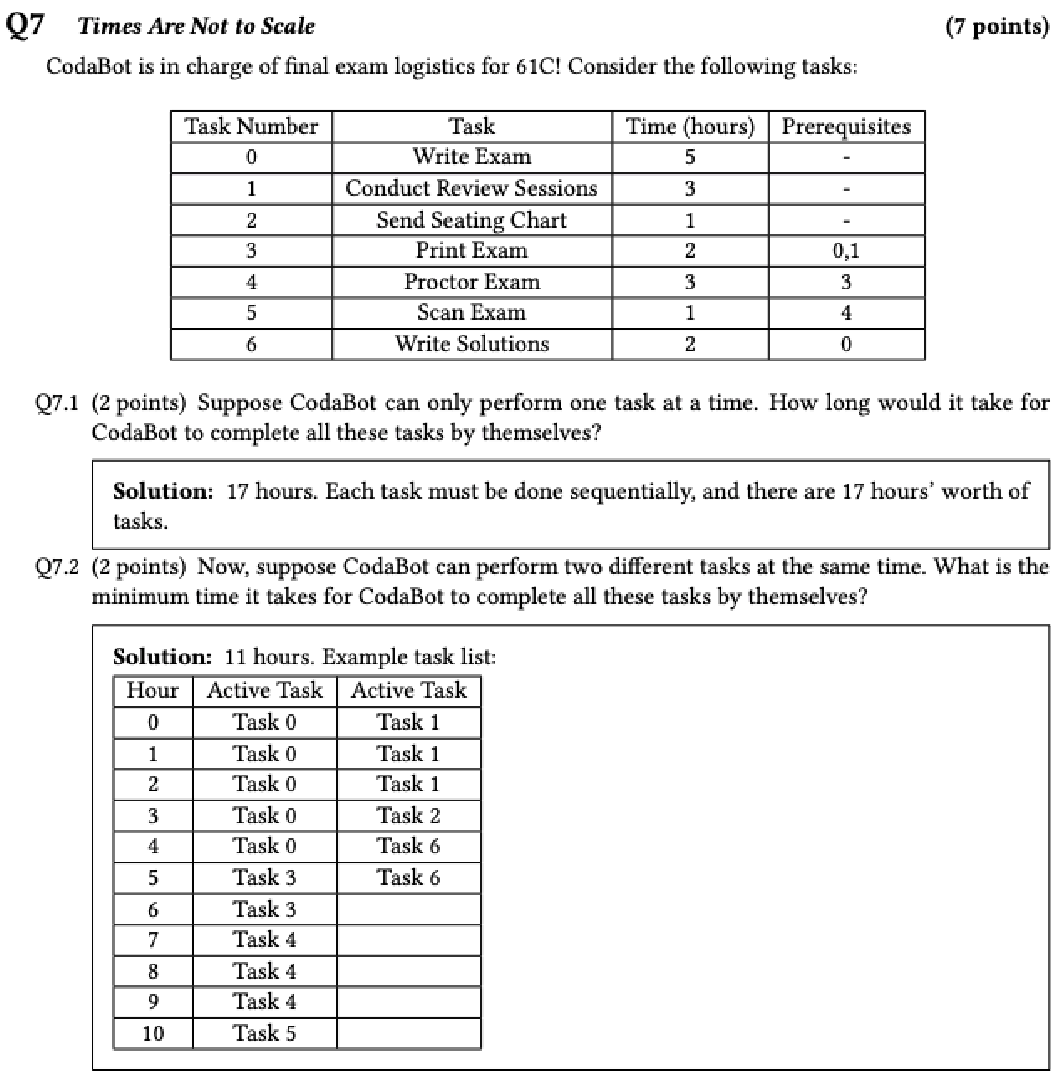
I'm also confused on this still.

Nikhil Kandkur STAFF 12mth #1044aeda

The standard bias for bias representation is $-(2^{n-1} - 1)$, where n is the number of bits we have.

Anonymous Emu 12mth #1044aafc ✓ Resolved

[SU 23 Q 7] For this question part 7.2 how can we start task 6 while task 0 is still executing at hour 5?By this logic task 4 could be started at Hour 6.



Q7 *Times Are Not to Scale* **(7 points)**

CodaBot is in charge of final exam logistics for 61C! Consider the following tasks:

Task Number	Task	Time (hours)	Prerequisites
0	Write Exam	5	-
1	Conduct Review Sessions	3	-
2	Send Seating Chart	1	-
3	Print Exam	2	0,1
4	Proctor Exam	3	3
5	Scan Exam	1	4
6	Write Solutions	2	0

Q7.1 (2 points) Suppose CodaBot can only perform one task at a time. How long would it take for CodaBot to complete all these tasks by themselves?

Solution: 17 hours. Each task must be done sequentially, and there are 17 hours' worth of tasks.

Q7.2 (2 points) Now, suppose CodaBot can perform two different tasks at the same time. What is the minimum time it takes for CodaBot to complete all these tasks by themselves?

Solution: 11 hours. Example task list:

Hour	Active Task	Active Task
0	Task 0	Task 1
1	Task 0	Task 1
2	Task 0	Task 1
3	Task 0	Task 2
4	Task 0	Task 6
5	Task 3	Task 6
6	Task 3	
7	Task 4	
8	Task 4	
9	Task 4	
10	Task 5	

Jero Wang STAFF 12mth #1044afcb

Sorry, it should start at hour 5, not hour 4.

Anonymous Cobra 12mth #1044aaec ✓ Resolved

[SP-23-Final-Q4.2 - Q4.4]

Could someone explain these three parts to me? Specifically, I don't get how we know all accesses to a and b miss for 4.2, and how the only misses are the 3 compulsory misses for 4.3 and 4.4.

Anonymous Pig 12mth #1044adae

For this question, I'm also wondering if we don't write "register int sum", would accessing sum count as accessing the cache?

Jero Wang STAFF 12mth #1044afee

Yes, it would.

Anonymous Pig 12mth #1044affb

Replying to Jero Wang

What about when we're writing a for loop (int i = 0; ...). Then every time we access, i wouldn't that count as a cache access as well?

Jero Wang STAFF 12mth #1044baac

Replying to Anonymous Pig

Yeah...it would. That was not intentional, we should have put `register` there.

Anonymous Pig 12mth #1044baad

Replying to Jero Wang

in other cache couting problems though accessing "i" never counts as accessing the cache...

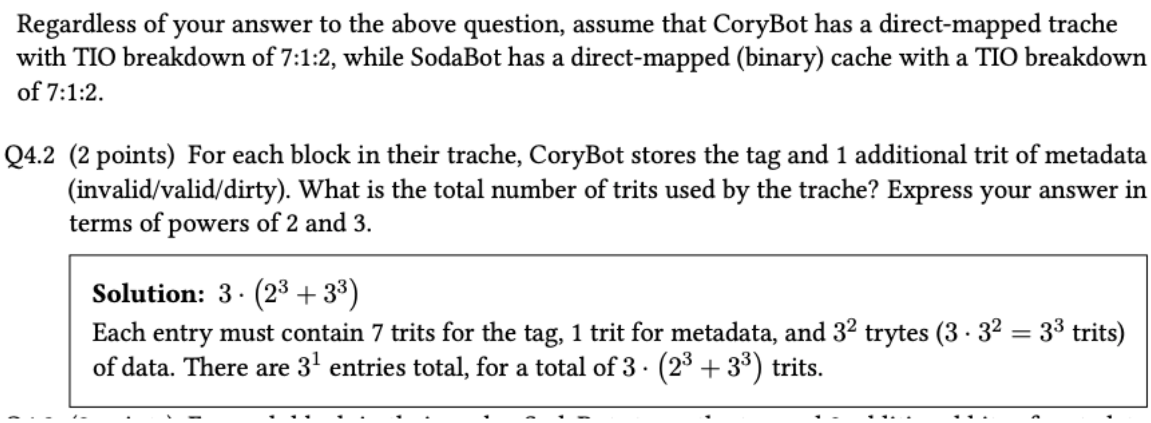
Jero Wang STAFF 12mth #1044affa

a and b miss because we alternate between their accesses (access b, then a, then b, etc), and they have the same index, so they evict each other.

With the updated cache designs in Q4.3 and 4.4, the only misses are compulsory because the other misses we had (from index conflicts) are now resolved, as each index can hold 2 (for Q4.3) or 4 (for Q4.4) cache lines.

Anonymous Emu 12mth #1044aacf ✓ Resolved

[SU 23 Q 4.2] How do we calculate how many trytes of data we have?Also how do we know that there are 3 entries total?



Regardless of your answer to the above question, assume that CoryBot has a direct-mapped trache with TIO breakdown of 7:1:2, while SodaBot has a direct-mapped (binary) cache with a TIO breakdown of 7:1:2.

Q4.2 (2 points) For each block in their trache, CoryBot stores the tag and 1 additional trit of metadata (invalid/valid/dirty). What is the total number of trits used by the trache? Express your answer in terms of powers of 2 and 3.

Solution: $3 \cdot \left(2^3 + 3^3\right)$
Each entry must contain 7 trits for the tag, 1 trit for metadata, and 3^2 trytes ($3 \cdot 3^2 = 3^3$ trits) of data. There are 3^1 entries total, for a total of $3 \cdot \left(2^3 + 3^3\right)$ trits.

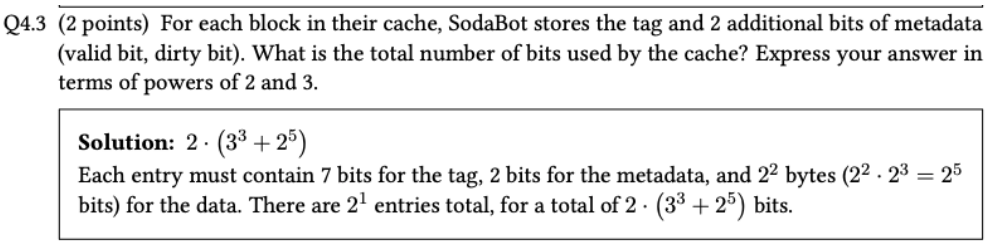
Byron Yang 12mth #1044aadc

According to the TIO breakdown, there is 1 trit for the index. Since 1 trit can hold 3 different values (0, 1, and 2), we have $3^1 = 3$ rows in our cache.

For the data, if we're counting metadata as well, we can just convert the final $3 \times \left(2^3 + 3^3\right)$ trits into trytes using the fact that 1 tryte = 3 trits. This comes out to 35 trytes.

Anonymous Emu 12mth #1044aade

that process doesn't hold for the following problem though...



Q4.3 (2 points) For each block in their cache, SodaBot stores the tag and 2 additional bits of metadata (valid bit, dirty bit). What is the total number of bits used by the cache? Express your answer in terms of powers of 2 and 3.

Solution: $2 \cdot (3^3 + 2^5)$

Each entry must contain 7 bits for the tag, 2 bits for the metadata, and 2^2 bytes ($2^2 \cdot 2^3 = 2^5$ bits) for the data. There are 2^1 entries total, for a total of $2 \cdot (3^3 + 2^5)$ bits.

Byron Yang 12mth #1044aadf

Replying to Anonymous Emu

Hmm, could you explain your thought process? Where do you think it's not working?

Anonymous Emu 12mth #1044aaef

Replying to Byron Yang

For the total entries shouldn't it still be 3^1 because our index bits hasn't changed?

Byron Yang 12mth #1044aafb

Replying to Anonymous Emu

We're using bits now since we're dealing with SodaBot. Bits only take on 2 values and we only have 1 bit for our index, which means we have 2 entries total.

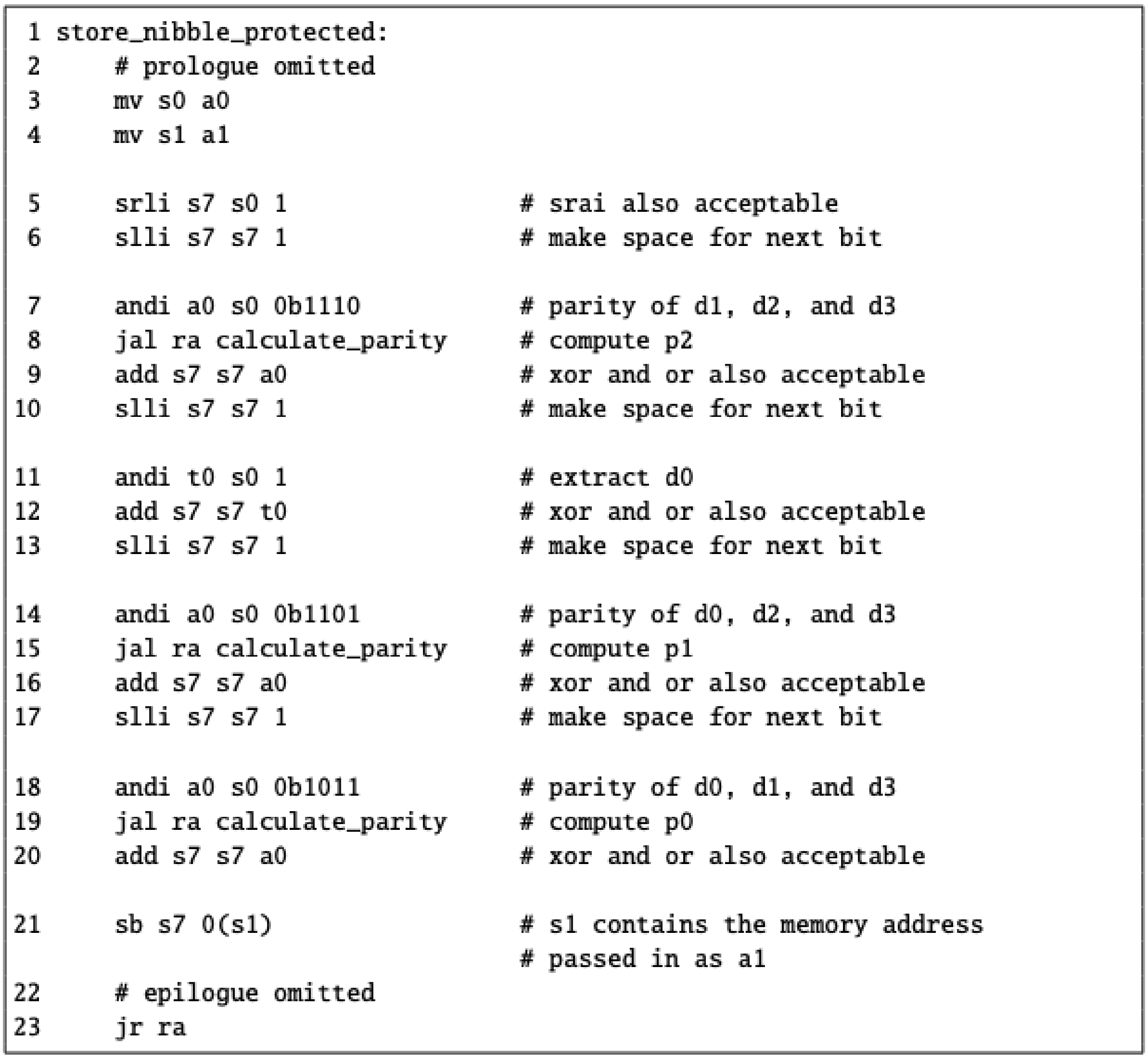
Anonymous Monkey 12mth #1044abbf

Can you explain where 2^3 + 3^3 comes from? I get that we have 7 trit from tag, 1 bit from metadata trit, and 1 trit from index and 2 trit from offset? Here I'm getting 11 trits to get 33 trits? Where is the additional trits coming from?

1

Anonymous Emu 12mth #1044aacc ✓ Resolved

[SU 23 Q 2 Part 2]Could someone please explain what this code is doing? I'm really confused about what is happening.



```
store_nibble_protected:
    # prologue omitted
    mv s0 a0
    mv s1 a1

    srli s7 s0 1                # srai also acceptable
    slli s7 s7 1                # make space for next bit

    andi a0 s0 0b1110           # parity of d1, d2, and d3
    jal ra calculate_parity     # compute p2
    add s7 s7 a0                # xor and or also acceptable
    slli s7 s7 1                # make space for next bit

    andi t0 s0 1                # extract d0
    add s7 s7 t0                # xor and or also acceptable
    slli s7 s7 1                # make space for next bit

    andi a0 s0 0b1101           # parity of d0, d2, and d3
    jal ra calculate_parity     # compute p1
    add s7 s7 a0                # xor and or also acceptable
    slli s7 s7 1                # make space for next bit

    andi a0 s0 0b1011           # parity of d0, d1, and d3
    jal ra calculate_parity     # compute p0
    add s7 s7 a0                # xor and or also acceptable

    sb s7 0(s1)                 # s1 contains the memory address
                                # passed in as a1

    # epilogue omitted
    jr ra
```

♡ 1 ···

J Jero Wang STAFF 12mth #1044afbe

#1044acff

♡ ···

Anonymous Llama 12mth #1044aabc ✓ Resolved

Su23-Final-Q5.4

Given a Virtual Address, how do you determine what is the VPN,PPN, and Page Offset bits?

♡ ···

B Byron Yang 12mth #1044aadd

If you're just given the virtual address, I don't think you can figure out any of those things. You also need to know the page size.

For the VPN, we can figure out how big our virtual memory is from the size of the virtual address. If the size is n bits, we have 2^n bytes in our virtual memory. We can then take this number and divide it by the page size to determine how many pages our virtual memory is. Then, we can do $\log(\text{number of pages})$ to figure out how many bits we need to represent each of our pages. This gives us our VPN in our virtual address and we can do the same thing but with physical memory to get PPN.

To get the page offset, if we know the page size, we know how many bytes are in a page. We can then take the log of this number to get the number of bits needed to address each individual byte. This is how many bits our page offset needs.

Hope this helps somewhat!

♡ 1 ···

Anonymous Mule 12mth #1044aafe

I'm not the person who asked this but thanks so much!

♡ ···

Byron Yang 12mth #1044abab

↩ Replying to Anonymous Mule

Glad to help :)

♡ 1 ···

Anonymous Llama 12mth #1044abae

For this specific, where 0xA01243 is given as the Virtual Address I'm still a bit confused. We are told to assume that we have 20-bit physical memory addresses and 24-bit virtual memory addresses but not sure how that tells us that the VPN is 0xA01 and the offset is 0x243. If they tell us to disregard the answers above this question not sure how we can know there are 12 bits of offset..

♡ 3 ···

Anonymous Monkey 12mth #1044abcc

↩ Replying to Anonymous Llama

^^

♡ 1 ···

Anonymous Quail 12mth #1044acca

↩ Replying to Anonymous Llama

I'm confused about the same thing!

♡ 1 ···

Anonymous Tapir 12mth #1044aefc ENDORSED

↩ Replying to Anonymous Llama

I think it tells us to disregard our answers, but we still need to use the fact that the pages are 4 KiB for question 5

♡ ···

Jero Wang STAFF 12mth #1044afcc

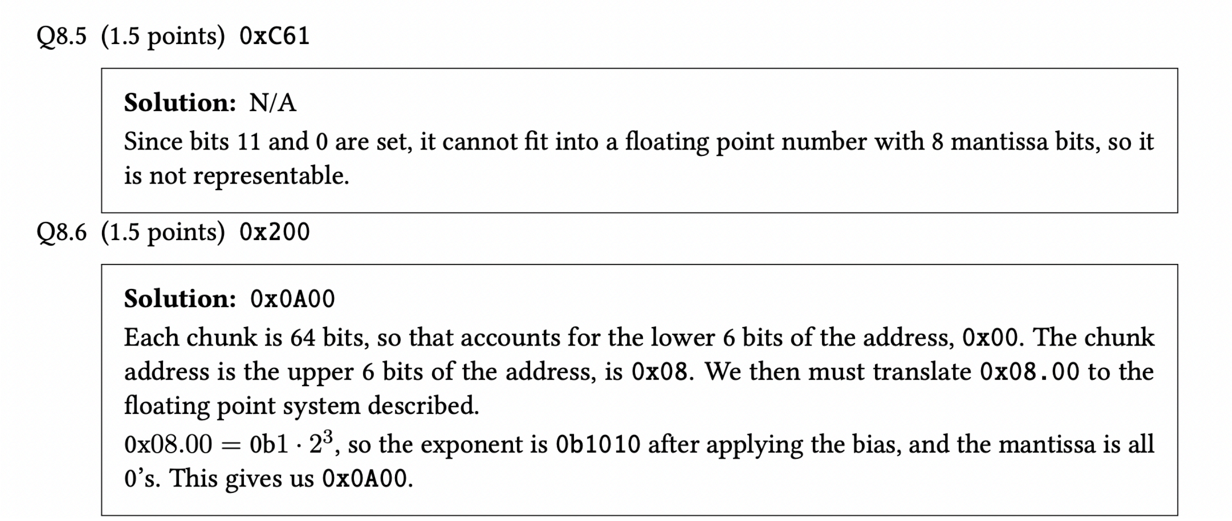
↩ Replying to Anonymous Llama

#1044aefc We ask you to ignore your answers to Q8.1-8.4, not the initial problem setup.

♡ ···

Anonymous Echidna 12mth #1044aaba ✓ Resolved

SU23-Final-Q8



Q8.5 (1.5 points) `0xC61`

Solution: N/A

Since bits 11 and 0 are set, it cannot fit into a floating point number with 8 mantissa bits, so it is not representable.

Q8.6 (1.5 points) `0x200`

Solution: `0x0A00`

Each chunk is 64 bits, so that accounts for the lower 6 bits of the address, `0x00`. The chunk address is the upper 6 bits of the address, is `0x08`. We then must translate `0x08.00` to the floating point system described.

$0x08.00 = 0b1 \cdot 2^3$, so the exponent is `0b1010` after applying the bias, and the mantissa is all 0's. This gives us `0x0A00`.

I don't really understand the explanation for 8.5 and I also don't really understand why we know that the lower 6 bits of the address is the chunk and the chunk address is the upper 6 bits. Any help would be appreciated--thank you!

♡ 2 ···

Anonymous Magpie 12mth #1044adce

Ditto

♡ ···

Jero Wang STAFF 12mth #1044afbc

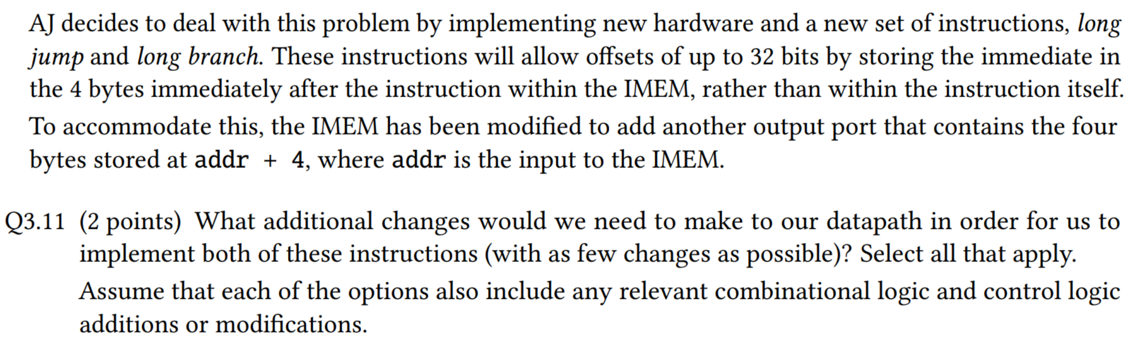
We know that the lower 6 bits are the offsets within the chunk because each chunk is 64 bytes, and we need 6 bits to address each of the 64 bytes.

For 8.5, we know that given a FP system with 8 mantissa bits, the MSB and LSB must be at most 9 bits apart (i.e. `1.0000 0000 1`). Since they are 12 bits apart, we don't have enough precision in this FP system to represent it.

♡ ···

Jiries Kaileh 12mth #1044aaaf ✓ Resolved

SU23-Final-Q3.11



AJ decides to deal with this problem by implementing new hardware and a new set of instructions, *long jump* and *long branch*. These instructions will allow offsets of up to 32 bits by storing the immediate in the 4 bytes immediately after the instruction within the IMEM, rather than within the instruction itself. To accommodate this, the IMEM has been modified to add another output port that contains the four bytes stored at `addr + 4`, where `addr` is the input to the IMEM.

Q3.11 (2 points) What additional changes would we need to make to our datapath in order for us to implement both of these instructions (with as few changes as possible)? Select all that apply. Assume that each of the options also include any relevant combinational logic and control logic additions or modifications.

For this question, I only filled in "add new input to the immediate generator". My idea was that if the branch failed, we could have RegWEn off for the Regfile and also have WBSel just set to PC+4 for the next instruction containing the 32-bit offset so you don't take the branch. Is this a feasible solution? the question does state "assume that each of the options also include any relevant combinational logic and control additions or modifications" and I believe that my method could be implemented in the control logic (albeit less efficient).

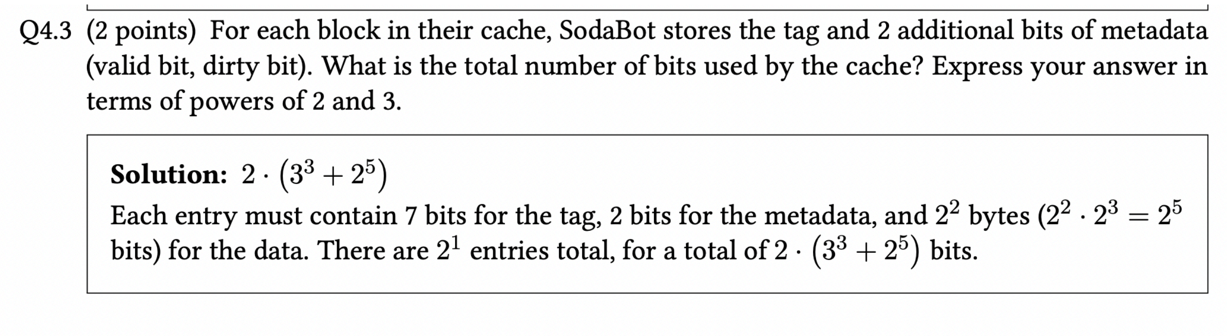
♡ ···

Jero Wang STAFF 12mth #1044afce

There's no way of detecting that the 32-bit offset is that offset without introducing additional logic and state elements, since the 32-bit offset could be an entirely valid RISC-V instruction as well.

Anonymous Echidna 12mth #1044ffd ✓ Resolved

SU23-Final-Q4



Q4.3 (2 points) For each block in their cache, SodaBot stores the tag and 2 additional bits of metadata (valid bit, dirty bit). What is the total number of bits used by the cache? Express your answer in terms of powers of 2 and 3.

Solution: $2 \cdot \left(3^3 + 2^5\right)$

Each entry must contain 7 bits for the tag, 2 bits for the metadata, and 2^2 bytes ($2^2 \cdot 2^3 = 2^5$ bits) for the data. There are 2^1 entries total, for a total of $2 \cdot \left(3^3 + 2^5\right)$ bits.

for this, why is it 3^3 in the parenthesis instead of 3^2? I thought since theres 7 bits for the tag and 2 bits for the metadata, that makes 9 (therefore 3^2) extra bits for each entry?

Nikhil Kandkur STAFF 12mth #1044ffe

#1044cc

1

Anonymous Mongoose 12mth #1044fed ✓ Resolved

SP23-Final-Q2.1

i'm confused about how we get the critical path, 855ps = τclk-to-q (35) + Immediate Generator (150) + Mux (75) + ALU (200) + Memory Read (300) + Mux (75) + τsetup (20)

isn't the immediate generator in the ID stage in the pipelined CPU on the reference card?

1

Nikhil Kandkur STAFF 12mth #1044ffc

#1044bb

Anonymous Echidna 12mth #1044fdf ✓ Resolved

SU23-Final-Q2 Part 2

I read through the answer key for this and I kind of get the logic but not completely--do you think it would be possible to write out how a0 and s7 change on example input and explain it a bit more? Thank you so much!

1

Andrew Liu STAFF 12mth #1044babf

So `s7` starts out with all of `a0` loaded into it:

```
s7 : 0 0 0 0 d3 d2 d1 d0
```

Then, we shift right then left to clear `d0`:

```
s7: 0 0 0 0 0 d3 d2 d1
s7: 0 0 0 0 d3 d2 d1 0
```

Then, we calculate `p2` and store it in the LSB, then shift left by 1 to open up space for the next bit:

```
s7: 0 0 0 0 d3 d2 d1 p2
s7: 0 0 0 d3 d2 d1 p2 0
```

Then, we repeat this pattern, grabbing the next desired bit and then shifting:

```
s7: 0 0 0 d3 d2 d1 p2 d0
s7: 0 0 d3 d2 d1 p2 d0 0
s7: 0 0 d3 d2 d1 p2 d0 p1
s7: 0 d3 d2 d1 p2 d0 p1 0
s7: 0 d3 d2 d1 p2 d0 p1 p0
```

And at the end of this, we store it to the memory in `a1`. `a0` keeps track of the relevant bits as we go, as needed:

```
a0: 0 0 0 0 (d3/0) (d2/0) (d1/0) (d0/0)
```

Anonymous Lion 12mth #1044fdc ✓ Resolved

SU23-Final-Q3.9

Is the implicit 0 considered part of the 12 bits, or does the immediate have 13 total bits? The formula for two's complement range is [-2^n-1, 2^n-1 - 1], so I'm struggling to see where the 2^n-1 - 2 came from. Also, the formula only works if n = 13, but according to risc-v architecture branch immediates only have 12 bits. just generally confused about these range questions so any guidance would help!

B

Byron Yang 12mth #1044feb

The implicit 0 is not considered in the 12 bits, so there are 13 bits total for branch immediates. To get the maximum positive jump, we need a 0 for our 12th bit (to make sure our number is non-negative) and 1's from the 11th bit to the 1st bit, then the implicit 0 for our 0th bit. The value of this is $2^1 + 2^2 + ... + 2^{11}$. To evaluate this, we can pretend that we have an implicit 1 rather than an implicit 0 in our 0th bit. This would make the value $2^0 + 2^1 + 2^2 + ... + 2^{11}$. Using some math (https://math.stackexchange.com/questions/1990137/the-idea-behind-the-sum-of-powers-of-2), this is equal to $2^{12} - 1$. Now since we don't actually have an implicit 1, we subtract this quantity by 1 to get the desired $2^{12} - 2$ as our largest positive jump.

Anonymous Dove 12mth #1044abcf

what is a good way to think about this question? I was completely lost when I looked at it first. Where did two's complement come from and and is this 2 to the power of the branch immediates

Anonymous Lobster 12mth #1044aebc

I get that the largest positive jump is 2^12 - 2 but how is the answer then 2^12 in the solutions

Anonymous Tapir 12mth #1044afba

Replying to Anonymous Lobster

The negative jump is 2^12 and we want the largest, which is why the solution is 2^12

♡ 1

Anonymous Cobra 12mth #1044fdb ✓ Resolved

SP23-Final-Q4.5

Don't we need to multiply hit rates by access times? Previously, AMAT = (L1 hit rate)(L1 access time) + (miss rate)(main memory access time).

Shouldn't the first computation be (0.75 * 6)? The solution only uses miss rates.

Nikhil Kandkur STAFF 12mth #1044ffb

Our formula for AMAT should be L1 access time + (miss rate * main memory access time), since we are always going to look into the L1 cache, and only if we miss look into memory.

Anonymous Swallow 12mth #1044faf ✓ Resolved

SU23-Final-Q6

I'm having some trouble understanding why we are using width = right - 4 and why we keep incrementing and decrementing left and right by 4. Is it because we are comparing each of the arrays 4 elements at a time, working our way from the endpoints to the middle?

Jiries Kaileh 12mth #1044aacd

To add on to this, why are we not assigning right = (width - 4) *sizeof(uint32_t) and incrementing the points by 4 * sizeof(uint32_t)? Is this because left and right are automatically multiplied by this when doing pointer arithmetic?

Thanks!

Jero Wang STAFF 12mth #1044addd

Yep

Anonymous Lion 12mth #1044aaff

same question

Anonymous Chimpanzee 12mth #1044abba

instead of vecReverse(vecLoad(matrix[i]+right)) in line 10, would it work if we just loaded the vector here and then reversed in the arguments for vecEq in line 11?

--> line 11: vecEq(vec1, vecReverse(vec2))

Also: What is the difference between vecLoad(matrix[i]+left) and vecLoad(*(matrix[i]+left)) and vecLoad(*(matrix+i+left))?

♡ ···

Jero Wang STAFF 12mth #1044adea

Replying to Anonymous Chimpanzee

Yes, that works.

`matrix[i] + left` indexes into `matrix` then adds an offset of `left` elements, while the second option dereferences the result (so it would be the same as `matrix[i][left]`). The last option doesn't index into `matrix`, instead returning `matrix` offset by `i + left` elements.

♡ 1 ···

Jero Wang STAFF 12mth #1044adde

Yep exactly

♡ ···

Anonymous Grasshopper 12mth #1044fab ✓ Resolved

SP23-Final-Q7

I just have no idea how to answer it. Any hints about how to deal with questions like this? Where should I start?

Thanks!

♡ ···

Nikhil Kandkur STAFF 12mth #1044fbb

From my side, my main tip for solving the problem is to read the question and code outline carefully: since we can "null terminate" our array with a 0, then we can consider an approach that involves in us continuing to use a copy of an array, and set our branch/jump instructions to 0. The code outline gives us this hint, and in addition tells us that we can keep using the same array, but in fact set our pointer values to various points in the array, rather than making several extra copies of memory.

If you have further questions, please look around the megathread, as there are several comments that provide insight into the problem. You may also reply to this thread if you have questions the other comments couldn't answer.

♡ 2 ···

Anonymous Grasshopper 12mth #1044fbe

help a lot! thanks

♡ ···

Anonymous Pheasant 12mth #1044eef ✓ Resolved

SU23-Final-Q8.1: can you explain where the standard bias of -7 comes from? is there an equation we shoujld use to calculate standard bias? I am very confused on bias in general. What should we know about it?

Anonymous Cod 12mth #1044aaae

The standard bias is calculated as -(2^(n-1)-1) where n represents how many bits the exponent has. I noted that from one of the course staff's slides. It's also shown on the third page of the ref Card.

Anonymous Dinosaur 12mth #1044eed ✓ Resolved

SU23-Final-Q8.6

the solutions say "0x08.00 = 0b1.2^3, so the exponent is 0b1010 after applying the bias"

how do we go from 4 exponent bits, with a bias of -7(?) to 0b1010? and where does the 8 come into this? thank you!

♡ 1

Anonymous Rhinoceros 12mth #1044acbf

^ bump I have the same question

Jero Wang STAFF 12mth #1044addb

`0x08.00 = 0b 0000 1000.0000 0000`, which can be simplified to `0b1000 = 1.0 * 2^3`. After we apply a bias of -7 to the exponent of 3, we get 10, which is `0b1010`.

Anonymous Ferret 12mth #1044ecb ✓ Resolved

SP23-Final-Q2.4

After going over relevant threads, I'm still confused as to why structural hazards can't occur. If we add another RegFile, can't we read from one and write to another? When the problem says that we can't read from and write to Regfile in the same clock cycle, does it mean reading from and writing to the same Regfile, or any two Regfiles?

Anonymous Magpie 12mth #1044ecc

I have the same question as well.

Nikhil Kandkur STAFF 12mth #1044efd

Even if we read from one and write to another, we would have to keep trying to make the regfiles the same values, which is time and energy consuming. Someone else in the threads cited this from discussion which helps with this:

> Through adding additional hardware, we can implement two ’read’ ports as well as a ’write’ port to the RegFile (where registers can be accessed). This solves the hazard of two instructions reading and writing to the same register

> simultaneously. False. The addition of independent ports to the RegFile allows for multiple instructions to access the RegFile at the same time (such as one instruction reading values of two operands, while another instruction is writing to a return register). However, this does not work if both instructions are reading and writing to the same register. Some solutions to this data hazard could be to stall the latter instruction by 1 cycle or to forward the read value from a previous instruction, bypassing the RegFile completely

Anonymous Ferret 12mth #1044eff

The discussion solutions only mentioned that instructions which write to then read from the same register can't be solved by adding a Regfile. However, if we're reading from and writing to different registers, adding a Regfile can help address this issue. Therefore reading from and writing to different registers would still be considered a structural hazard, but it seems like that's not the case. Even if it were time and energy consuming, it could still be solved by adding a Regfile.

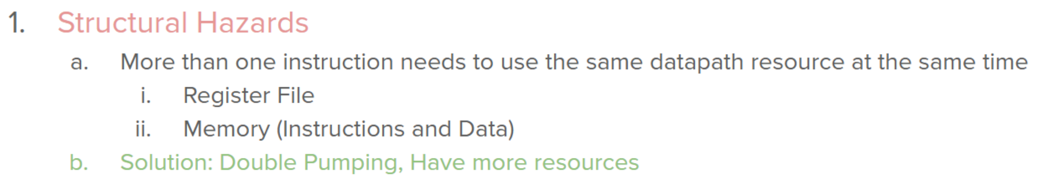
Nikhil Kandkur STAFF 12mth #1044fba

Replying to Anonymous Ferret

The hazard of not being able to write and read in the same cycle is not one that is structural: we are not in the case where there are less ports than available for reading and writing to instructions. For our regfile, we will still have the 2 read ports and 1 write port, which will allow us to write and read to/from the same resource, so we will not have a case where we need more ports than there already are. We therefore have to treat this hazard as a data hazard since this is not structural.

Hayden Loarie 12mth #1044fca

Replying to Nikhil Kandkur



1. Structural Hazards
   a. More than one instruction needs to use the same datapath resource at the same time
      i. Register File
      ii. Memory (Instructions and Data)
   b. Solution: Double Pumping, Have more resources

This is from todays review session. Wouldn't this issue be solved with Double Pumping so we could write to the register file and then read right after?

Nikhil Kandkur STAFF 12mth #1044fda

Replying to Hayden Loarie

Theoretically, but they did say that that could not happen (writing and reading in the same cycle is not allowed for this problem).

Hayden Loarie 12mth #1044fdd

Replying to Nikhil Kandkur

I guess I'm still confused about the problem then. From this slide Double Pumping is a solution to Structural Hazards. The issue that arose in the problem is because we don't have Double Pumping, so shouldn't it be a structural hazard?

Or should Double Pumping have been listed as a solution to Data Hazards instead?

Or am I just misunderstanding both things?

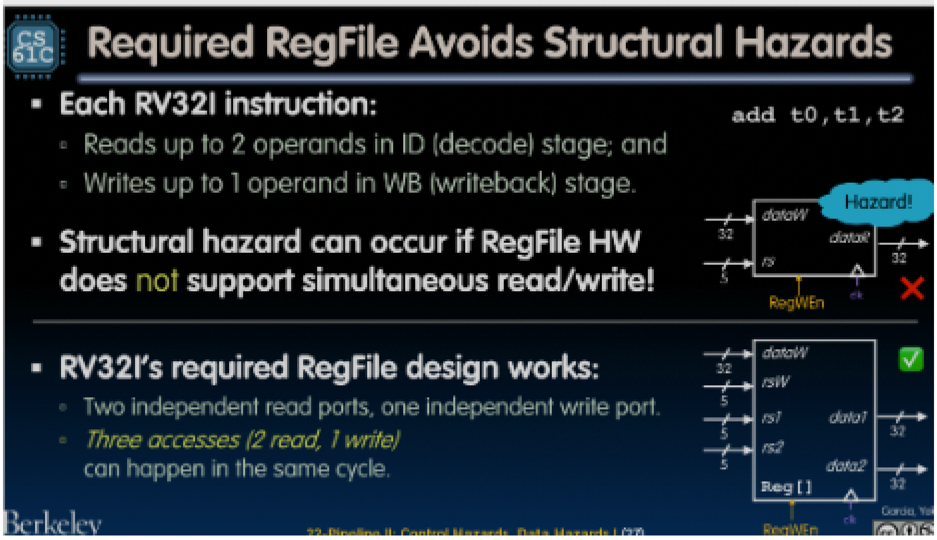
♡ ···

A Aryansh Shrivastava 12mth #1044abcd

Replying to Hayden Loarie

I'm confused as well. If we could theoretically solve the problem of not being able to read and write from the Regfile in the same clock cycle by changing the hardware to make the Regfile allow read and write in the same clock cycle (double pumping as above), doesn't that make this a structural hazard? (I'm not proposing we add a second Regfile or anything, but that we replace the original.)

Even our lecture slides specifically classify not supporting simultaneous read/write as a hardware issue that creates a structural hazard:



Required RegFile Avoids Structural Hazards

- Each RV32I instruction:
  - Reads up to 2 operands in ID (decode) stage; and
  - Writes up to 1 operand in WB (writeback) stage.
- Structural hazard can occur if RegFile HW does not support simultaneous read/write!
- RV32I's required RegFile design works:
  - Two independent read ports, one independent write port.
  - *Three accesses (2 read, 1 write)* can happen in the same cycle.

```
add t0,t1,t2
```

♡ 1 ···

A Aryansh Shrivastava 12mth #1044abfd

Replying to Aryansh Shrivastava

It would be really helpful if we could get a clarification on this before the final because this could potentially impact how we interpret problems on the final!

♡ ···

J Julia Tomaszewska 12mth #1044accb

Replying to Aryansh Shrivastava

I would also appreciate an explanation about this, since I'm also confused.

♡ 1 ···

N Nikhil Kandkur STAFF 12mth #1044adcd

Replying to Hayden Loarie

After talking to the other TAs for help on this, and they've confirmed that double pumping is a solution for data hazards, not structural hazards.

♡ ···

Anonymous Ferret 12mth #1044aebb

Replying to Nikhil Kandkur

Is this the correct interpretation:

If our regfile does not have enough ports for simultaneous read/write (e.g. it only has a read port or only has a write port), then this constitutes a structural hazard since it can be resolved by adding more ports or changing the structure of our regfile.

However, if our regfile already has both read ports and write ports, then we consider the following two situations: 1) one instruction writes to a register and the following instruction reads from a different register. This behavior is supported by our current regfile and not the aforementioned regfile, therefore if we were using the regfile that did not support simultaneous R/W, this would be a structural hazard. 2) one instruction writes to a register and the following instruction reads from the same register. Our new regfile does not support this kind of behavior, and this cannot be resolved by adding regfiles, either. Therefore this constitutes a data hazard, and can be resolved by double pumping, i.e. write-then-read in a clock cycle.

The main takeaway should be: if we can resolve the hazard by adding more hardware (i.e. adding more regfiles or adding more ports to the regfile), then the hazard is a structural hazard.

Jero Wang STAFF 12mth #1044afcf

Replying to Anonymous Ferret

Yes, that's correct.

Anonymous Lark 12mth #1044afcd

Replying to Nikhil Kandkur

Are we supposed to ignore the lecture slides + final exam review session slide then?

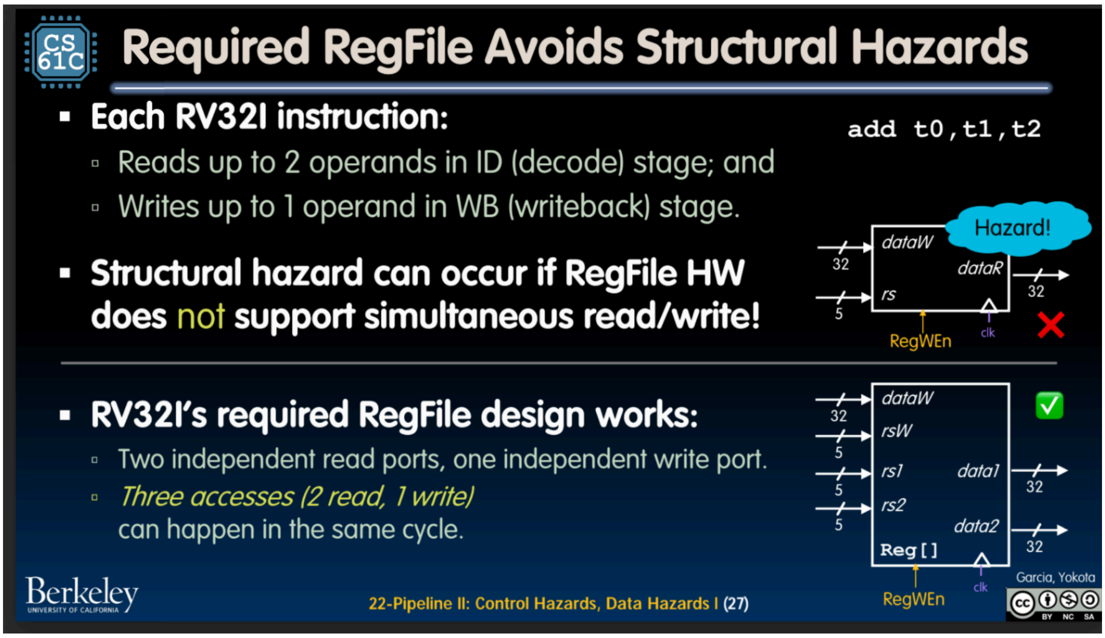
Jero Wang STAFF 12mth #1044afda

Replying to Anonymous Lark

Which lecture slide? I'll ask the final review team to update the review slides.

Anonymous Lark 12mth #1044afdc

Replying to Jero Wang



Required RegFile Avoids Structural Hazards

- **Each RV32I instruction:**
  - Reads up to 2 operands in ID (decode) stage; and
  - Writes up to 1 operand in WB (writeback) stage.
- **Structural hazard can occur if RegFile HW does not support simultaneous read/write!**
- **RV32I's required RegFile design works:**
  - Two independent read ports, one independent write port.
  - *Three accesses (2 read, 1 write)* can happen in the same cycle.

Wouldn't this slide imply that structural hazard should be selected in the exam question? "Structural hazard can occur if RegFile HW does not support simultaneous read/write"

♡ 1 ···

Anonymous Crane 12mth #1044acfa

Replying to Nikhil Kandkur

Same question here!

♡ 3 ···

Anonymous Magpie 12mth #1044eca ✓ Resolved

Sp23-Final-Q7:
Why is the *codecopy = 0 necessary in this code solution.

From my understanding, we are looping through codecopy and whenever we see a jump/branch instruction, we set a data pointer to the next instruction to mark a new segment. Then, we continue to iterate through codecopy and do the same thing as needed -> until we hit the null terminator.

If that's the case, what good does setting *codecopy = 0 here do.?

♡ ···

N Nikhil Kandkur STAFF 12mth #1044eea

We would have to do `*codecopy = 0` since that sets a "null terminator" for our portion of the list. If we have a list `[add x0 x0  x0, slli x2 x2 1, beq x1 x2 label1, add x0 x1 x2]` , after our code we would have `[add x0 x0  x0, slli x2 x2 1, NULL, add x0 x1 x2]` , and this is valid since we specify that

> Each of these arrays should be "null-terminated"; that is, the last element of each array should be the number 0, to signify the end of the array.

♡ ···

Anonymous Cassowary 12mth #1044aabb

can we do codecopy[0] = 0 instead?

♡ ···

N Nikhil Kandkur STAFF 12mth #1044aabe

Replying to Anonymous Cassowary

That works!

♡ ···

Anonymous Cassowary 12mth #1044aacb

Replying to Nikhil Kandkur

Thanks! also for the very last blank of that question i did result = &data instead of *result = data, would that be fine?

♡ ···

Nikhil Kandkur STAFF 12mth #1044aace

↰ Replying to Anonymous Cassowary

That would NOT work, because `result = &data` only sets the local copy of `result` to `&data` . When we return from the function, the value we created will not be there as a result. `*result = data` sets the value stored at the pointer to `data`, which will carry on after returning the function.

♡ 1 ···

Anonymous Oryx 12mth #1044ebd ✓ Resolved

SP-23-Final-Q2.1, 2.1

Why for the critical path occurs in stage 2 for a load instruction we don't need to add Regfile Setup?

855ps = τclk-to-q (35) + Immediate Generator (150) + Mux (75) + ALU (200) + Memory Read (300) + Mux (75) + τsetup (20)

♡ ···

Anonymous Oryx 12mth #1044ece

by the way, is there a case that we need to add both τsetup (20) and Regfile Setup (20) when we calculate the minimum operating clock period.

♡ ···

Nikhil Kandkur STAFF 12mth #1044efe

τ_{setup} = time for Regfile setup so it wouldn't matter which setup time we add.

For your second question, there is not a case when we would have to add both τ_{setup} and Regfile setup since even in the single cycle datapath, you can have the PC setup either happen at the same time of Regfile setup or before Regfile setup (and at the same time as some other combinational logics).

♡ ···

Anonymous Horse 12mth #1044abaf

↰ Replying to Nikhil Kandkur

Why do we factor in the immediate generator into this calculation? I thought that it would be in the ID stage (stage 1 of the problem).

♡ ···

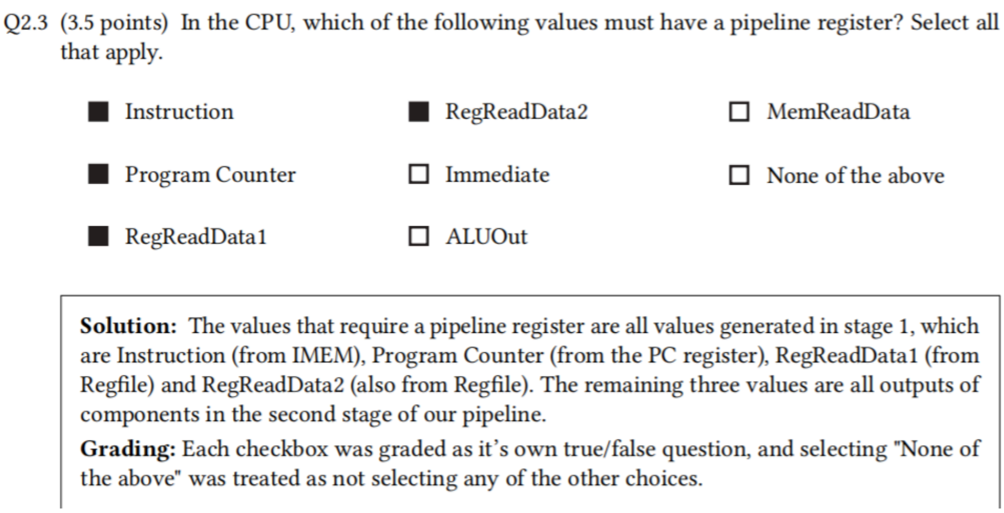
Nikhil Kandkur STAFF 12mth #1044abbe

↰ Replying to Anonymous Horse

#1044bb

♡ ···

Anonymous Chicken 12mth #1044ebb ✓ Resolved



Q2.3 (3.5 points) In the CPU, which of the following values must have a pipeline register? Select all that apply.

■ Instruction

■ Program Counter

■ RegReadData1

■ RegReadData2

□ Immediate

□ ALUOut

□ MemReadData

□ None of the above

Solution: The values that require a pipeline register are all values generated in stage 1, which are Instruction (from IMEM), Program Counter (from the PC register), RegReadData1 (from Regfile) and RegReadData2 (also from Regfile). The remaining three values are all outputs of components in the second stage of our pipeline.

Grading: Each checkbox was graded as it's own true/false question, and selecting "None of the above" was treated as not selecting any of the other choices.

Sp23-final-Q2, why the values that require a pipeline register are all values generated in stage 1, rather than Instruction, Program Counter, RegReadData1, RegReadData2, Immediate, ALUOut, and MemReadData?

N Nikhil Kandkur STAFF 12mth #1044fac

Remember that we have to construct our pipeline such that stage 1 and stage 2 are separated, which means that any values we send from stage to another need to be sent into a pipeline register and the values from the pipeline registers will be used in the next stage. When going from stage 1 to stage 2, the only values that are generated in stage 1 that are used in stage 2 are Instruction (for control logic), Program Counter (for PC input into A Mux and PC + 4), RegReadData1 (for A Mux), and RegReadData2 (for B Mux and memory). Hope this helps!

Anonymous Sand Dollar 12mth #1044ead ✓ Resolved

Sp23-Final-Q7:

After we set *codecopy to 0 for 7.12, how does this allow our data[i]'s to be set correctly? I get that the code saves the initial codecopy address and then we scan through all the valid instructions. Then, when we hit the first invalid element (it's either equal to 0 or it's a jump or branch instruction), we stop and zero it out. However, after doing this, wouldn't our data[i] include all the remaining elements in our array, even those that extend beyond the 0 we set? What's ensuring our data[i]'s stop at those 0's?

1

N Nikhil Kandkur STAFF 12mth #1044edd

Yes `data[i]` would contain the remaining elements in our array, but if you were to print the values of these lists, since we are setting a "null terminator" in our list, and we specified that

> Each of these arrays should be "null-terminated"; that is, the last element of each array should be the number 0, to signify the end of the array.

Anonymous Sand Dollar 12mth #1044faa

I thought stopping when hitting the null terminator was something that only applied to char* arrays. Do int* arrays also follow this behavior?

N Nikhil Kandkur STAFF 12mth #1044fad

Replying to Anonymous Sand Dollar

No, we can only do what we did because we specified that you can set 0 to be a null terminator.

♡ 1 ···

Anonymous Sand Dollar 12mth #1044fae

Replying to Nikhil Kandkur

Oh ok thank you Nikhil!

♡ ···

Anonymous Salmon 12mth #1044eaa ✓ Resolved

SU23-Final-Q8.1

I read the explanation: The answer says that those bits would be the exponent bits, so given standard bias, we would need 4 exponent bits to represent 63 in our IEEE-754 FP standard.

but I am not sure how that works still. 2^4+2^3+2^2+2^1+2^0 will still not attain 63 but only 31?

♡ 4 ···

J Jero Wang STAFF 12mth #1044adcc

The question is asking about how many exponent bits it takes to represent numbers up to 63 in a FP system. To reach 63, the exponent must be at least 5, post-bias (2^5 * 1.11111 = 63). As a result, to get a post-bias exponent value of 5, we need at least 4 exponent bits.

♡ ···

Anonymous Clam 12mth #1044adec

Why can't we use 3 bits to get a post bias exponent value of 5? Also what would the pre-bias exponent value be and how would we find that? Sorry I am just a bit confused because I thought that we usually shift the binary representation and then add the bias but here you seem to be saying that 5 must be exponent after we add the bias and also seem to be shifting the binary representation by 5? A clarification would be helpful.

♡ ···

Anonymous Ferret 12mth #1044dff ✓ Resolved

SU23-Final 7

Is question 7 just trial and error or is there a general approach to dividing the work. I know that we want to minimize the time any worker is "idle". Is this optimized by having all pre-reqs done first, etc or is there no stand-by algorithm

♡ ···

J Jero Wang STAFF 12mth #1044adcb

Yeah, somewhat trial-and-error, but you should use heuristics like the one you mentioned where you minimize the amount of time that each worker is idle. Generally, a greedy algorithm is enough for these questions, or at least they will give you a good starting point to make small modifications.

♡ ···

Anonymous Ferret 12mth #1044dfe ✓ Resolved

SP23-Final-Q2

Why is the Immediate Generator part of EX and not ID? Why is it included in the longest path calculation for load operations in stage 2?

Anonymous Starling 12mth #1044ebc

#1044bb

Anonymous Ferret 12mth #1044dfb ✓ Resolved

SP23-Final-Q7.7

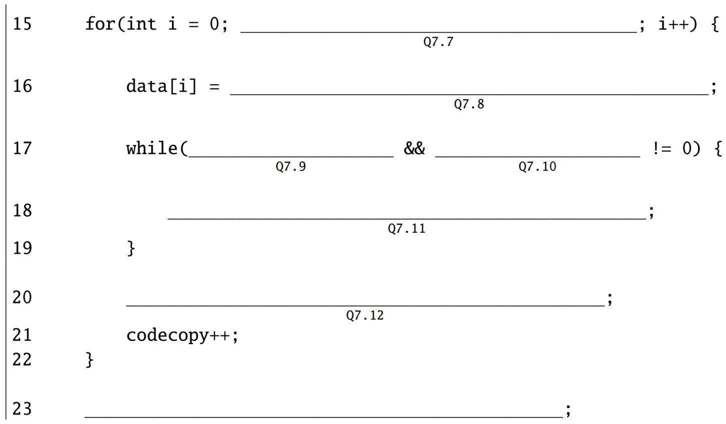
Solutions should say $i < num + 1$ instead of $num + 1$.

Nikhil Kandkur STAFF 12mth #1044edb

Yup! Sorry about that

Anonymous Rail 12mth #1044def ✓ Resolved

SP23-Final-Q7:



```
for(int i = 0; ________________________________________; i++) {
                              Q7.7

    data[i] = ________________________________________________;
                                   Q7.8

    while(____________________ && ____________________ != 0) {
                   Q7.9                      Q7.10

        ________________________________________________;
                             Q7.11
    }

    ________________________________________________;
                         Q7.12
    codecopy++;
}

________________________________________________;
```

What exactly is this second for loop doing?

Anonymous Sand Dollar 12mth #1044eab

From my understanding, it checks whether the current instruction at address codecopy is "valid" (it isn't a jump or branch instruction and it's not a 0) and if so, increments codecopy. So we skip through all the valid instructions starting at address codecopy essentially.

Anonymous Salmon 12mth #1044dee ✓ Resolved

SU23-Final-Q4.2,Q4.3

Why do we assume the offset bits are in tryte or bytes while the tag and index are in trit and bits? Because we could directly write in the number of index+offset but then for the offset had to convert to trit or bit

1

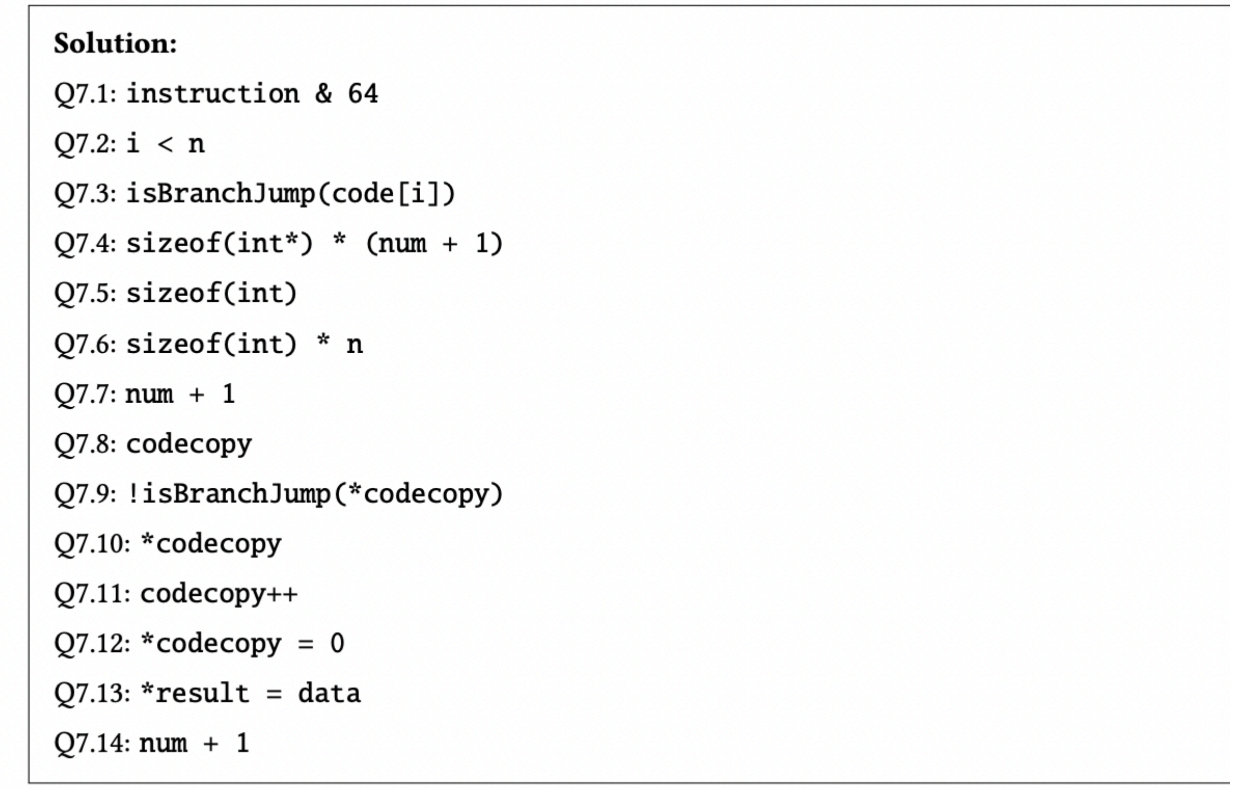
Jero Wang STAFF 12mth #1044adca

Each cache line needs to store the tag itself (so it knows what the tag of the data is), whereas the data is storing the actual data, which is in terms of bytes/trytes. While the size of data is related to offset, you're not directly storing the offset.

Anonymous Caterpillar 12mth #1044dec ✓ Resolved

SP23-Final-Q7.9-7.12

For 7.9, how does referencing codecopy give a single instruction that we can then input into isBranchJump? For 7.10, why are we checking is the dereferenced codecopy is not equal to 0? Also what does codecopy++ do in 7.11? Also what is line 7.12 doing?



Solution:

Q7.1: `instruction & 64`

Q7.2: `i < n`

Q7.3: `isBranchJump(code[i])`

Q7.4: `sizeof(int*) * (num + 1)`

Q7.5: `sizeof(int)`

Q7.6: `sizeof(int) * n`

Q7.7: `num + 1`

Q7.8: `codecopy`

Q7.9: `!isBranchJump(*codecopy)`

Q7.10: `*codecopy`

Q7.11: `codecopy++`

Q7.12: `*codecopy = 0`

Q7.13: `*result = data`

Q7.14: `num + 1`

Anonymous Ferret 12mth #1044eac ENDORSED

From a high-level perspective, we want to scan over all lines of code and determine what gets put into the array we return. The purpose of having `codecopy` is that we are able to use it to keep track of which line we've scanned up to (notice that `codecopy` gets incremented in the for loop).

Q7.9: `codecopy` initially points toward the first line of code, since we performed a `memcpy` operation. After that, notice how in each iteration of the for loop, `data[i]` and the initial value of `codecopy` points towards the instruction right after a jump/branch instruction. In the subsequent parts of each for loop iteration we are essentially trying to find where the following jump/branch instruction is, and have `codecopy` point to the instruction after that, so that the next `data[i]` again points to the instruction right after a jump/branch.

Q7.10: calloc allocates n+1 values. After memcpy, the last element of the codecopy array is a 0. If we didn't check for `*codecopy != 0` when codecopy points to this last element, since isBranchJump(0) is 0, we would increment `codecopy` and end up outside of the `codecopy` array.

Q7.12: A good way to go about looking at this is that you could compare the RISC-V code in the problem statement with the sample array right below it. If you match each element of the array with the RISC-V code, (i.e. in this example, 0 with beq x0 x0 pass, 0 with beq x0 x0 pass, add a0 t0 t1 with add a0 t0 t1, so on and so forth), you can notice that the array we return contains all instructions but with the jump/branch instructions zero-ed out. This is an interesting side effect of null-terminating the array as mentioned in the problem statement. `*codecopy=0` makes this possible.

♡ 1 ···

N Nikhil Kandkur STAFF 12mth #1044eeb

Great explanation! I also wanted to add that for 7.12, we specified in the question description that

> Each of these arrays should be "null-terminated"; that is, the last element of each array should be the number 0, to signify the end of the array.

So just setting our value to 0 in our array would still be valid as a "null terminator".

♡ 1 ···

Anonymous Sand Dollar 12mth #1044deb ✓ Resolved

Sp23-Final-Q7.1:

Would checking if (instruction AND 127 == 99) or (instruction AND 127 == 103) or (instruction AND 127) == 111 work for this question? Also, for the solution, do we AND by 64 because all instructions besides jumps and branches (and ignoring ebreaks and ecalls) have the first digit of their opcode as 0? Thanks!

♡ 1 ···

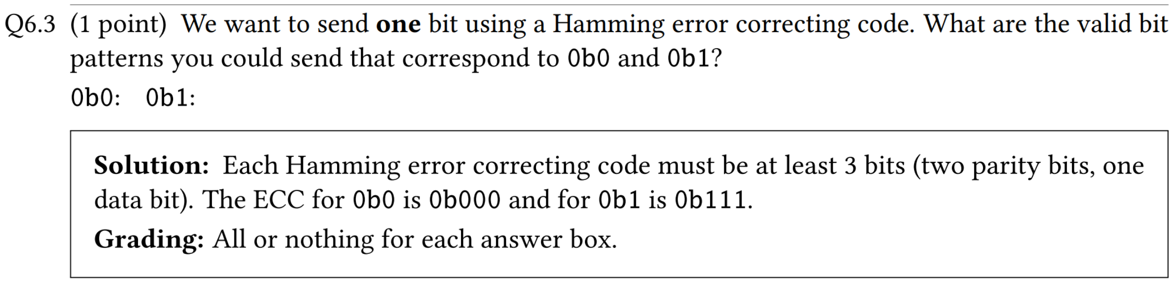
J Justin Yokota STAFF 12mth #1044eba

The first two parts do not work as intended because == has higher operator precedence than AND, and you would get marked off for incorrect AND/OR syntax. Otherwise, this should be fine if it matches the reference card.

♡ ···

Anonymous Rail 12mth #1044dde ✓ Resolved

SP23-6.3:



Q6.3 (1 point) We want to send **one** bit using a Hamming error correcting code. What are the valid bit patterns you could send that correspond to 0b0 and 0b1?

0b0: 0b1:

Solution: Each Hamming error correcting code must be at least 3 bits (two parity bits, one data bit). The ECC for 0b0 is 0b000 and for 0b1 is 0b111.

Grading: All or nothing for each answer box.

Wouldn't 0b1 as 0b111 be incorrect bc there are an odd number of bits when checking the hamming code?

Shouldn't there be multiple valid solutions?

♡ ···

Anonymous Starling 12mth #1044ebe

If I'm understanding your question correctly, 0b111 is the only correct answer for a data bit of 0b1 since we need to have 3 bits for any Hamming error correcting code (data only starts on the third bit). Therefore, if we set 0bxx1, then both the first and second bits are 1 as follows by parity. Hope this helps!

Anonymous Rail 12mth #1044ecd

This makes sense. Thanks!

Anonymous Rail 12mth #1044ddd ✓ Resolved

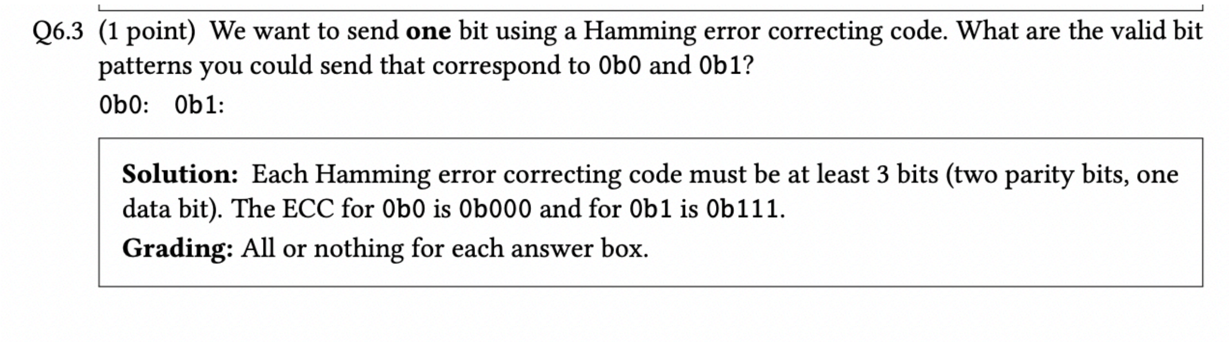
SP23-Final-Q4:

we have a[i + j], b[j], and results[i] as accesses. Do we store (i, j, i+j), (a[i + j], b[j], results[i]) or (0x1000, 0x2000, 0x3030) in cache? How do we know 'what type of data' we store in cache?

Nikhil Kandkur STAFF 12mth #1044fbc

We would have to store the memory at the addresses we are loading/storing from in our cache, so that would be `(a[i + j], b[j], results[i])` .

Anonymous Caterpillar 12mth #1044ddc ✓ Resolved



Q6.3 (1 point) We want to send **one** bit using a Hamming error correcting code. What are the valid bit patterns you could send that correspond to 0b0 and 0b1?

0b0: 0b1:

Solution: Each Hamming error correcting code must be at least 3 bits (two parity bits, one data bit). The ECC for 0b0 is 0b000 and for 0b1 is 0b111.

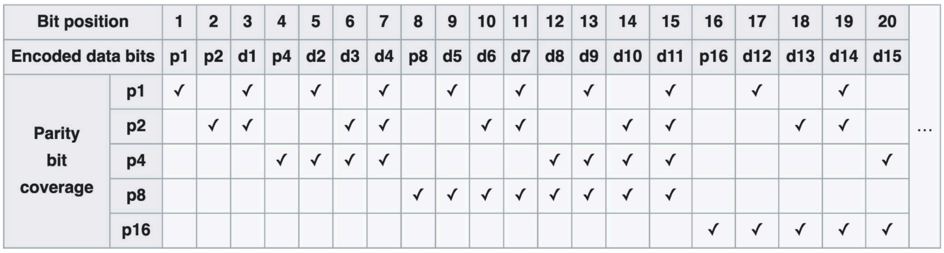
Grading: All or nothing for each answer box.

SP23-Final-Q6.3

How do we know that the Hamming error correcting code must be at least 3 bits? I'm not sure how to determine that

3

Anonymous Rail 12mth #1044ddf ENDORSED



Bit position		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	
Encoded data bits		p1	p2	d1	p4	d2	d3	d4	p8	d5	d6	d7	d8	d9	d10	d11	p16	d12	d13	d14	d15	
Parity bit coverage	p1	✓		✓		✓		✓		✓		✓		✓		✓		✓		✓		
	p2		✓	✓			✓	✓			✓	✓			✓	✓			✓	✓		...
	p4				✓	✓	✓	✓					✓	✓	✓	✓					✓	
	p8								✓	✓	✓	✓	✓	✓	✓	✓						
	p16																✓	✓	✓	✓	✓	

The parity bits p1 and p2 correpond to d1, so the code should be composed of p1, p2, d1

2

Anonymous Opossum 12mth #1044ddb ✓ Resolved

SP23-Final-Q4.2

How do we know that “For each set of accesses to an and b, the program experiences thrashing since the index of these blocks conflict”? How do we use a=0x1000, b=0x2000, and results=0x3030, to determine thrashing/index conflicts?

Nikhil Kandkur STAFF 12mth #1044fbf

There is no mathematically formal definition for thrashing: thrashing is when we consistently have misses when in our accesses due to the structure of our code. If we see how we access our code, we see consistent misses for `a` and `b`, which fits our definition of thrashing. `a=0x1000, b=0x2000, and results=0x3030` tells us that our addresses are located far away from each other, so that means that if we have to access anything from those arrays, they will require separate cache blocks. In addition, we can make our TIO breakdown from this, and see that there is a lot of conflicts due to this.

Anonymous Opossum 12mth #1044aadb

Why do they conflict if they are stored in separate cache blocks?

♡ 6

Anonymous Echidna 12mth #1044abbc

Replying to Anonymous Opossum

I am also confused about this! If they are located farther from each other then why would that cause thrashing? Wouldn't it be better if they were farther away because then there are less conflicts?

Anonymous Echidna 12mth #1044abbd

Replying to Anonymous Echidna

or is it because both of them have the same index (0) that the conflicts occur?

Jero Wang STAFF 12mth #1044adbe

Replying to Anonymous Echidna

Not necessarily farther, but the addresses here are specifically chosen such that their indexes conflict. I think if `b` was at `0x2010`, they won't thrash anymore.

Jero Wang STAFF 12mth #1044adbd

Replying to Anonymous Opossum

They conflict because they have the same index.

Anonymous Kookabura 12mth #1044babc

Replying to Jero Wang

Still couldn't understand this - why is thrashing happening? I'm not sure why they have the same index (as for b we are indexing j and for a we are indexing i+j)

Anonymous Squid 12mth #1044dda ✓ Resolved

for capacity miss, does the whole cache need to be filled? if the specific index is filled but there are open slots for other indices and we want to use the specific index for a different tag is it a capacity or conflict miss

Anonymous Starling 12mth #1044ebf

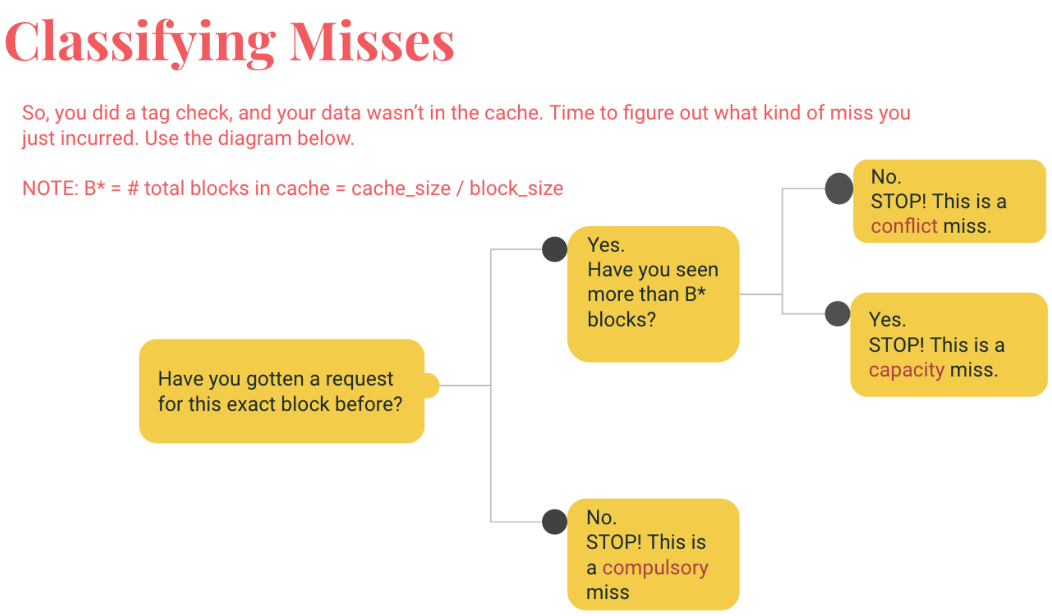
Yes. In the case you described, it would be a conflict miss. You can think of capacity misses as misses which would be resolved if we increased the size of the cache.

♡ ···

N

Nikhil Kandkur STAFF 12mth #1044eda

Yup! I included a infographic that helps identify the differences between misses on my discussion slides. Hope this helps!



Classifying Misses

So, you did a tag check, and your data wasn't in the cache. Time to figure out what kind of miss you just incurred. Use the diagram below.

NOTE: B* = # total blocks in cache = cache_size / block_size

Have you gotten a request for this exact block before?

Yes.
Have you seen more than B* blocks?

No.
STOP! This is a conflict miss.

Yes.
STOP! This is a capacity miss.

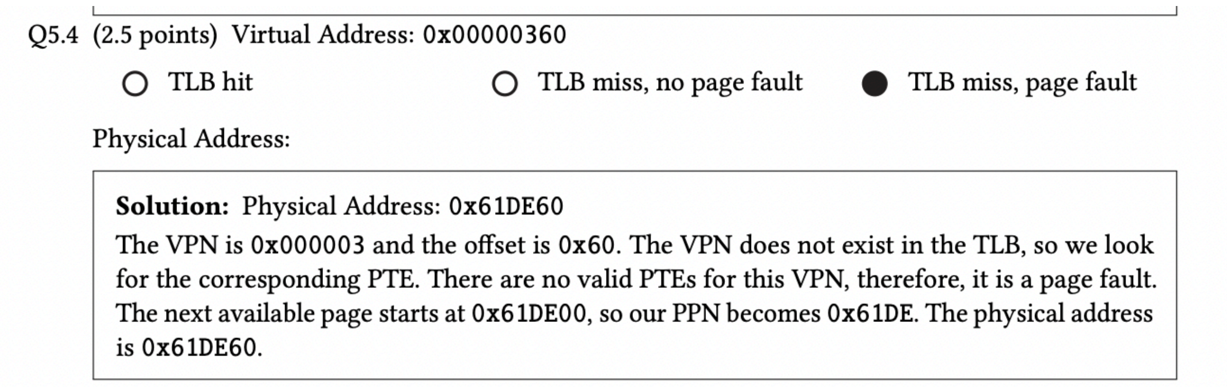
No.
STOP! This is a compulsory miss

♡ ···

Anonymous Caterpillar 12mth #1044dcf ✓ Resolved

SP23-Final-Q5.4

Shouldn't the PPN for 0x61DE00 be 0xDE00 instead of 0x61DE since PPN is bits 0-15?



Q5.4 (2.5 points) Virtual Address: 0x00000360

○ TLB hit ○ TLB miss, no page fault ● TLB miss, page fault

Physical Address:

Solution: Physical Address: 0x61DE60
The VPN is 0x000003 and the offset is 0x60. The VPN does not exist in the TLB, so we look for the corresponding PTE. There are no valid PTEs for this VPN, therefore, it is a page fault. The next available page starts at 0x61DE00, so our PPN becomes 0x61DE. The physical address is 0x61DE60.

♡ 1 ···

B

Byron Yang 12mth #1044dea ENDORSED

You're right that the PPN is bits 0-15, but that's only when you're looking at the page table entry. Here, 0x61DE00 is our physical address and we split it into the PPN (0x61DE) and the offset (0x00). Thus, the PPN is 0x61DE.

♡ ···

Anonymous Grasshopper 12mth #1044dba ✓ Resolved

SP23-Final-Q3

Is `num_cols - 3` acceptable for Q3.2? And is `vec_load(matrix + i + j)` acceptable for Q3.4?

Anonymous Grasshopper 12mth #1044dbe

```
vec_load(matrix + i*col_nums + j)
```

Nikhil Kandkur STAFF 12mth #1044fcb

For your first question, I'm not too sure how `num_cols - 3` will still give us a way to handle the tail case.

For your second question: #1044efc

Anonymous Grasshopper 12mth #1044aabd

Replying to Nikhil Kandkur

For tail case, we still start loop from `num_cols / 4 * 4`, is it ok?

Nikhil Kandkur STAFF 12mth #1044aabf

Replying to Anonymous Grasshopper

Hmm that might be able to work, but it would be best and safer if you set your end condition to num_cols/4*4

Anonymous Grasshopper 12mth #1044aaca

Replying to Nikhil Kandkur

got it, thx!

Anonymous Pig 12mth #1044aaea

If we did num_cols - num_cols % 4 instead of num_cols / 4 * 4 is that ok?

Jero Wang STAFF 12mth #1044adbc

Yeah, that's fine too.

Xiaole Guo 12mth #1044dae ✓ Resolved

SP23-Final-Q9

following IEEE-754 convention, with 5 exponent bits, if the exponent bits and mantissa bits are all 0, then it represents the object 0 right? That's why if the 19-bit input is all zeros, we can still say that it can be represented as a 19-bit float following IEEE-754 convention with 5 exponent bits?

Anonymous Chimpanzee 12mth #1044dce

I don't understand this solution, can someone explain it?

Jero Wang STAFF 12mth #1044adbb

This should work for the 0 answer, since all bits are 0, the expression in the NOT evaluates to 0, and after NOT returns a 1.

Xiaole Guo 12mth #1044dad ✓ Resolved

SP23-Final-Q6.4

for the third statement which is selected, indeed the knowledge is covered in lecture 35. But in lecture 35, I don't think the ppt states that if checksum is not correct, then an error is either in the payload or the checksum. I just wonder where the lecture slides mentions what an incorrect checksum implies?

Nikhil Kandkur STAFF 12mth #1044fcf

I think what they implied by that is that there would be a guaranteed error, but we cannot assume that it is solely from just the payload or just the checksum, since we don't construct our data in a way that allows us to identify if our error is coming from our checksum or payload.

Xiaole Guo 12mth #1044fee

Is the data saved to the disk? If it's saved to memory, then shutting it down will cause data loss.

Nikhil Kandkur STAFF 12mth #1044fef

Replying to Xiaole Guo

We cannot make any assumptions about where the data is saved unless specified. But yes, your reasoning sounds correct in that there would be data loss if saved to the disk and then shutting it down. What's very likely happening in the scenario of an incorrect checksum is that we have it in some local buffer, and we analyze our received data there.

Xiaole Guo 12mth #1044fff

Replying to Nikhil Kandkur

I am so sorry that I posted that follow-up question in the wrong place... #1044fee should be the response to #1044fce. But thanks for the explanation anyway!

Xiaole Guo 12mth #1044dac ✓ Resolved

SP23-final-Q6.2

I wonder why the first statement is correct. If we shut down M how do we make sure the output of the mapping task that M just finishes is saved?

Raiyan Rizwan 12mth #1044dbd

I think in multi-processing the manager would only shut down M if M had already completed its work and sent the data back (in which case that data can be sent to another worker node).

N Nikhil Kandkur STAFF 12mth #1044fce

Finishing the work also implies that the work is saved since we are working with read/write operations, so we would not have to worry too much about that.

X Xiaole Guo 12mth #1044aaaa

Is the data saved to the disk? If it's saved to memory, then shutting it down will cause data loss.

N Nikhil Kandkur STAFF 12mth #1044aaab

Replying to Xiaole Guo

Well if the task is only finished if it's saved to memory, then we won't have to worry about it. However, if the task is not saved to memory, the MapReduce framework will make sure that another active worker will redo the task to prevent data loss.

X Xiaole Guo 12mth #1044aaac

Replying to Nikhil Kandkur

I see. So the key idea is that whether or not the data is saved doesn't matter, because even if it's not saved, the MapReduce framework will ensure that another worker redoes the task and saves the data?

N Nikhil Kandkur STAFF 12mth #1044aaad

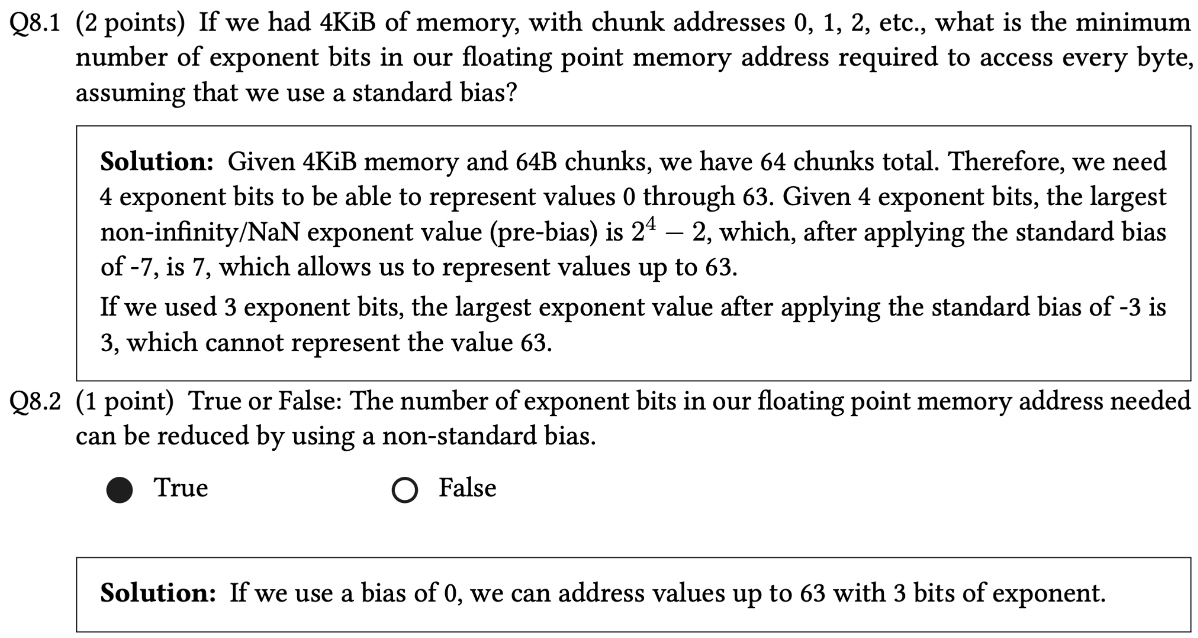
Replying to Xiaole Guo

Yup!

Anonymous Goose 12mth #1044ced ✓ Resolved

SU23-Final-Q8.1: How do we get -7 as the standard bias, shouldn't it be -15? (2^#exponent bits -1).

Also, I am still confused how we can use 4 bits to access 0-63?



Q8.1 (2 points) If we had 4KiB of memory, with chunk addresses 0, 1, 2, etc., what is the minimum number of exponent bits in our floating point memory address required to access every byte, assuming that we use a standard bias?

Solution: Given 4KiB memory and 64B chunks, we have 64 chunks total. Therefore, we need 4 exponent bits to be able to represent values 0 through 63. Given 4 exponent bits, the largest non-infinity/NaN exponent value (pre-bias) is $2^4 - 2$, which, after applying the standard bias of -7, is 7, which allows us to represent values up to 63.

If we used 3 exponent bits, the largest exponent value after applying the standard bias of -3 is 3, which cannot represent the value 63.

Q8.2 (1 point) True or False: The number of exponent bits in our floating point memory address needed can be reduced by using a non-standard bias.

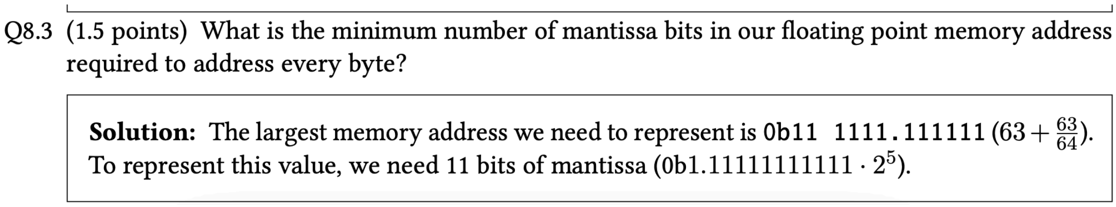
● True ○ False

Solution: If we use a bias of 0, we can address values up to 63 with 3 bits of exponent.

♡ 2

Anonymous Goose 12mth #1044cee

and, how do we know that the largest number we have to represent is <64?



Q8.3 (1.5 points) What is the minimum number of mantissa bits in our floating point memory address required to address every byte?

Solution: The largest memory address we need to represent is `0b11 1111.111111` $(63 + \frac{63}{64})$. To represent this value, we need 11 bits of mantissa $(0b1.1111111111 \cdot 2^5)$.

♡ 1 ···

Jero Wang STAFF 12mth #1044afbd

The question states that we have 4KiB of memory, and each chunk is 64B, so there are 64 chunks.

♡ ···

Anonymous Ferret 12mth #1044dfc

If we have e exponent bits, the bias is $-(2^{e-1} - 1)$.

♡ ···

Anonymous Ferret 12mth #1044dfd ENDORSED

Recall that the range of the exponent pre-bias is 0 through $2^e - 1$. If we have 3 exponent bits, excluding denorms, infinities, and Nans, the range of the exponent value is 1 through 6. After applying bias = 0, the exponent value range is still 1 through 6, meaning that we can represent numbers up to $1.111.... \times 2^6$ (... meaning that the numbers of trailing 1's we have depends on how many mantissa bits we have), which is larger than 63 ($2^6 - 1$).

♡ ···

Anonymous Turkey 12mth #1044cec ✓ Resolved

SP23-Final-Q2.5

How do we know the next available page starts at 0x61DE00?

♡ ···

R Raiyan Rizwan 12mth #1044cfa

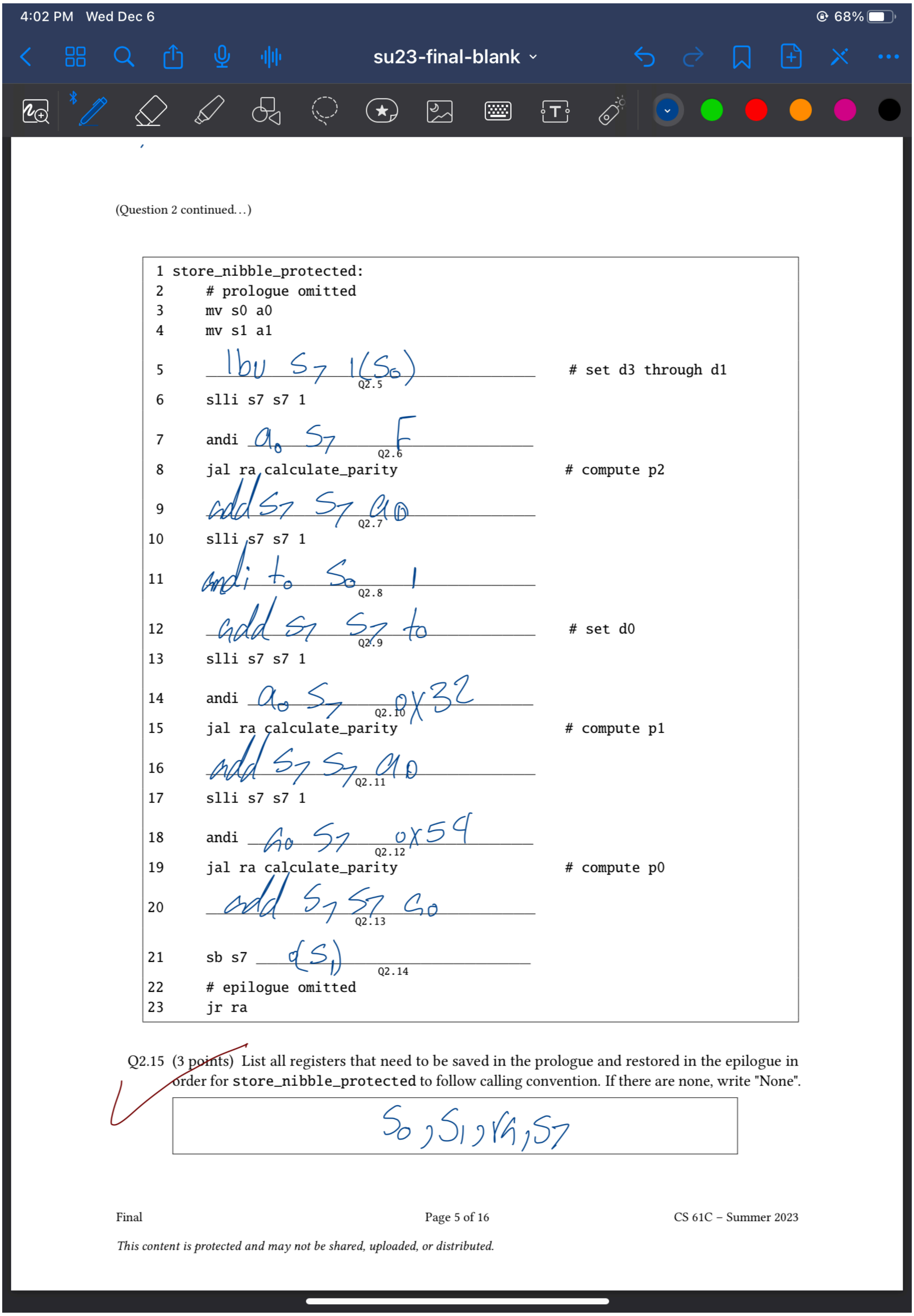
its given in the problem description

♡ ···

Anonymous Weasel 12mth #1044cdf ✓ Resolved

SU23_Final-Q2

Hi, can this solution also be a correct?



(Question 2 continued…)

```
store_nibble_protected:
    # prologue omitted
    mv s0 a0
    mv s1 a1
    lbu S7 1(S0)        Q2.5                    # set d3 through d1
    slli s7 s7 1
    andi a0 S7  F       Q2.6
    jal ra calculate_parity                     # compute p2
    add S7 S7 a0        Q2.7
    slli s7 s7 1
    andi t0 S0 1        Q2.8
    add S7 S7 t0        Q2.9                    # set d0
    slli s7 s7 1
    andi a0 S7 0x32     Q2.10
    jal ra calculate_parity                     # compute p1
    add S7 S7 a0        Q2.11
    slli s7 s7 1
    andi a0 S7 0x54     Q2.12
    jal ra calculate_parity                     # compute p0
    add S7 S7 a0        Q2.13
    sb s7 0(S1)         Q2.14
    # epilogue omitted
    jr ra
```

Q2.15 (3 points) List all registers that need to be saved in the prologue and restored in the epilogue in order for `store_nibble_protected` to follow calling convention. If there are none, write "None".

S0, S1, ra, S7

This content is protected and may not be shared, uploaded, or distributed.

Jero Wang STAFF 12mth #1044adad

Q2.5 doesn't work since `s0` is not a pointer, and you're loading at a byte offset of 1, not a bit offset of 1, so you'd be loading undefined/zeros.

Anonymous Ferret 12mth #1044cde ✓ Resolved

Sp23-Final-Q1.1

Why is readArr an R type instruction? If rs1 contains a pointer to the array we're reading from in memory and we're storing the value in rd, isn't this basically a load instruction with the only difference being our offset is in rs2 instead of being an immediate? How would we know that it is an R-type instruction if we don't have the opcode? Is it just because it's in the form

```
<name> rd rs1 rs2?
```

N

Nikhil Kandkur STAFF 12mth #1044fbd

We don't know that it is an R type instruction, but the way we would store our values would be similar to an R type instruction, which has `rd`, `rs1`, and `rs2` encoded in its instruction. So yes, since it is in a similar form to `<name> rd rs1 rs2`, we don't have to make a lot of changes to support this.

Anonymous Cod 12mth #1044abea

Is the reason why we don't interpret this as a S type instruction because we want to writeback to rd and S-types don't do WB? I'm still a little confused as to why we can't just add a new immediate type that left-shifts rs2 by 2 bits

Anonymous Quail 12mth #1044adba

Replying to Anonymous Cod

we can't add a new immediate type, because this instruction doesn't use an immediate, the value has to be read from a register

Anonymous Cod 12mth #1044adef

Replying to Anonymous Quail

OHHHH i totally missed that, thank youu

Anonymous Clam 12mth #1044cdc ✓ Resolved

Sp23-Final-Q9:

I was wondering how to approach this problem and what concepts are important to understand in order to get the answer to this question. Also I was wondering why it is the case that the "maximum exponent of a 19 bit floating point number is 30 − 15 = 15".

Anonymous Clam 12mth #1044dbf

+1. I do get the ~(A|B|C) but not the second part. For the former, the idea is that exponent = 31 (all ones) is reserved for infinities and NaNs in FP, which is why 30 is the max exponent value. Then with the bias of -15, the max power of 2 in the actual number is = 30 - 15 = 15. Hence, the biggest number possible is 1.11111...1 * $2^{15} < 2^{16}$, so we can't represent anything >= 2^{16} (bits A, B, or C on).

For the second part, aren't the unsigned numbers going to be all whole numbers? Unless there's a decimal point in the binary which the prompt doesn't make clear. In the case of whole numbers, the step size is $2^{e-15} * 2^{-13} = 1$ so e <= 28, right?

I don't really get "The smallest level of precision we can represent is 0b1.0000000000001, which has a total of fourteen bits between the largest and smallest one bit...."

♡ ···

Jero Wang STAFF 12mth #1044adac

It's really important to understand FP and the properties of FP, such as the step size for a given exponent. Additionally, familiarity with the Boolean algebra properties would also be nice, but they are available on the refcard.

It has 5 exponent bits, which means the maximum is exponent is 2^5-1 (31), but exponent of 31 represents NaNs or infinities, so the actual maximum exponent value is 30 for numbers that are representable by an unsigned integer. After applying bias of -15, we end up at an exponent of 2^15.

Yes, unsigned are all going to be whole numbers, so e>=28, not <=.

The smallest level of precision is referring to how many bits it takes to represent the FP number, at the minimum. Since we have 13 mantissa bits, and then including the implicit 1, there are 14 bits of precision (anything 13 bits or less won't be able to represent this number).

♡ ···

Anonymous Magpie 12mth #1044cce ✓ Resolved

Sp23-Final-Q2:

From my understanding, the current critical path is the one for the load instruction - where we include the time for 2 muxes (A/BMux and the WBMux), ALU, and DMEM). I was wondering why we don't include the Regfile setup time in the critical path - since I remember that the critical path is a path from register to register and the load instruction writes the result of DMEM to Regfile (with the starting register being RegReadData1 and the ending register being the Regfile).

♡ 1 ···

Nikhil Kandkur STAFF 12mth #1044eec

We do include the RegFile setup time here, notice that τ_{setup} = time for Regfile setup.

♡ ···

Anonymous Magpie 12mth #1044fde

what's the difference between *τsetup* and Regfile setup? They're listed separately on the table.

♡ ···

Nikhil Kandkur STAFF 12mth #1044fea

↩ Replying to Anonymous Magpie

They meant t_setup to be the setup of a register, which technically is equal to the setup of registers in a regfile.

♡ 1 ···

Xiaole Guo 12mth #1044ccb ✓ Resolved

Sp23-Final-Q2

Why the immediate generator is in stage 2? According to the pipilined datapath in the lecture notes, immediate generator should be in ID stage, which should be in stage 1 of this quesiton.

♡ 2 ···

Anonymous Elk 12mth #1044cea

^

♡ 1 ···

Anonymous Starling 12mth #1044cef

+1

♡ 1 ···

Anonymous Chimpanzee 12mth #1044dab

#1044bb

♡ ···

X Xiaole Guo 12mth #1044cca ✓ Resolved

SP23-Final-Q2.4

The Regfile can't read and write in the same clock cycle, why isn't it a strucural hazard? Isn't it hazard due to HW contraint? Also, what are the possible ways to fix this issue? I can think of stalling and forwarding to fix the issue.

♡ 1 ···

N Nikhil Kandkur STAFF 12mth #1044ded

#1044bba

♡ ···

Anonymous Red deer 12mth #1044cbf ✓ Resolved

Sp23 Final Q6.4

Can someone explain the answer? Why is it that there must be an error in either payload or checksum? I thought checksum is a code that checks the data integrity of payload, so the answer should be that the payload is incorrect.

♡ 1 ···

J Justin Yokota STAFF 12mth #1044cda

If the checksum got corrupted with a correct payload, we'd get an error (since the corrupted checksum doesn't match the payload). So it is possible that the error lies in the checksum, not the payload.

♡ ···

Anonymous Grasshopper 12mth #1044cbd ✓ Resolved

SP23-Final-Q8.3

Is A: 1,3; B: 0,2; C:4 an acceptable task list?

Also, are there any tips or algorithms to follow to answer questions like this? Since there are several combinations of how to order each task, do I just need to list out all the possible combinations and eliminate ones that are obivously too long, or is there a trick to finding the best combination faster?

Thanks!

♡ 1 ···

Justin Yokota STAFF 12mth #1044dbb

Any task list that takes 200 minutes to run was considered fully correct.

In terms of solving this problem, there is no known algorithm to solve this efficiently, and the problem is NP-complete by reduction to the Knapsack problem.

♡ ···

Anonymous Grasshopper 12mth #1044cbc ✓ Resolved

SP23-Final-Q7.3

Is `isBranchJump(*(code+i))` acceptable?

♡ 1 ···

Nikhil Kandkur STAFF 12mth #1044edf

Yup! But do make sure you include the parentheses as you already did, or that could give us some funky values XD

♡ ···

Anonymous Grasshopper 12mth #1044acbe

Got it, thank you!

♡ ···

Anonymous Grasshopper 12mth #1044cbb ✓ Resolved

SP23-Final-Q7.9

Is `!isBranchJump(codecopy[0])` acceptable?

♡ 1 ···

Nikhil Kandkur STAFF 12mth #1044ede

Yup! `*codecopy = codecopy[0]`

♡ 1 ···

Anonymous Grasshopper 12mth #1044cba ✓ Resolved

SP23-FinalQ7.1

Is `return (00000000000000000000000001110000 & instruction) == 1100000` acceptable?

How about `return (112 & instruction) == 1100000` ?

♡ 2 ···

Justin Yokota STAFF 12mth #1044daf

It is not; the second line would check for equality with the decimal number 1,100,000.

Similarly, in the first line, `00000000000000000000000001110000` would get interpreted in octal (and the right side would be treated as a decimal number again), so you'd get the wrong answer.

♡ ···

Anonymous Grasshopper 12mth #1044dcb

I see. If, for the second line, I wrote `== 96` instead of `== 1100000`, would it be correct?

♡ ···

Anonymous Grasshopper 12mth #1044caf ✓ Resolved

SP23-Final-Q5

Hi, I'm a little confused about the purpose of the dirty bit in page tables/TLB. When is the dirty bit usually on? Would the dirty bit just be on whenever we change the VPN-PPN mapping? When would this realistically happen?

Also, is it possible for a VPN to be in the TLB with a valid bit that's off? In my head, I can't think of a scenario where this would be possible, since if data at a VPN is needed by the computer, it will look in the TLB. If the page is not in the TLB, it will go to the page table and bring the lookup data into the TLB. If the page is not a resident according to the page table, the page will be made a resident, and the TLB and the page table's valid bit will be updated. So why would a page's valid bit ever be off in the TLB, since wouldn't the page's lookup data in the TLB be updated just by the act of putting the page's lookup data in the TLB?

♡ 2 ···

Anonymous Clam 12mth #1044dca

+1, also I think whenever the cache "resets" all the blocks are set to valid = 0, since there's no way to "clear out" the memory? or if there is it's more time consuming

♡ ···

Nikhil Kandkur STAFF 12mth #1044edc

The dirty bit is on when we write to the page associated with the entry. If we only read and trigger an eviction in our TLB, then we would have a replaced VPN-PPN mapping in the TLB and set that valid bit to 1 and dirty bit to 0.

Yes it is possible for a VPN to be in the TLB with an off valid bit. This would happen when we trigger a context switch to change programs, and this would involve invalidating our entire TLB since a VPN-PPN mapping in one program may not exist in another program.

♡ ···

Anonymous Chamois 12mth #1044aeec

Does a context switch also invalide the page table? Or only the TLB

♡ ···

Nikhil Kandkur STAFF 12mth #1044afab

Replying to Anonymous Chamois

The page table would not be invalidated, because you can store multiple pages tables for different programs in memory.

♡ 1 ···

Anonymous Grasshopper 12mth #1044cae ✓ Resolved

SP23-Final-Q3.14, 3.15

Does `#pragma omp critical` not require an opening curly brace below it and a closing one after the critical section?

♡ 2 ···

Anonymous Chimpanzee 12mth #1044ccf

I think it does not require curly braces if the critical section is only the one immediate line following the directive.

♡ 2 ···

Justin Yokota STAFF 12mth #1044dbc

Yup. This is also true for if, for, and while statements; the brackets in C only serve to "combine" a full code block into one line, so any one-liner statement doesn't need brackets.

♡ 1 ···

Anonymous Grasshopper 12mth #1044cad ✓ Resolved

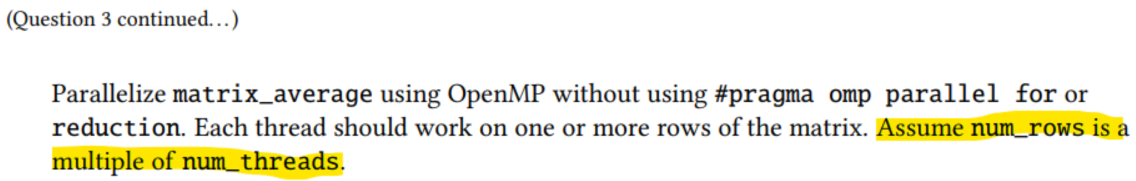
SP23-Final-Q3.11,3.12,3.13

Don't the solutions for these not consider the tail case? for example, if we have a 5x5 matrix, and 3 threads, then the chunk size will be 1, and thread 0 will work on row 0, thread 1 will work on row 1, and thread 2 will work on row 2. Once these threads finish working on these rows, they will exit the for loop. Thus, rows 3 and 4 will never be accounted for.

Is there a flaw in my logic, or do we only assume that the number of rows is a multiple of the number of threads?

♡ 1 ···

Anonymous Caribou 12mth #1044ccd



(Question 3 continued...)

Parallelize `matrix_average` using OpenMP without using `#pragma omp parallel for` or `reduction`. Each thread should work on one or more rows of the matrix. Assume `num_rows` is a multiple of `num_threads`.

you are correct that we're making that assumption

♡ ···

Anonymous Grasshopper 12mth #1044aecd

Somehow I completely missed this, thanks!

♡ ···

Anonymous Grasshopper 12mth #1044cac ✓ Resolved

SP23-Final-Q3.4

Is `vec_load(*matrix + i*num_cols + j)` acceptable? If not, is it because this assumes the arrays storing the elements are contiguous in memory when in reality they may not be?

♡ 1 ···

Nikhil Kandkur STAFF 12mth #1044efc

~~Yes this would work, because based on the structure of the outer `for` loop, you can assume that the matrix rows are stored in indices right next to each other.~~

`matrix[i] + j = matrix + i * sizeof(int*) + j * sizeof(int)` , so we couldn't do `*matrix + i*num_cols + j` . It is simply easier for yourself (and us for grading) to simply write `matrix[i] + j` XD

♡ ···

Anonymous Pig 12mth #1044aaed

I'm a bit confused on why `matrix[i] + j = matrix + i * sizeof(int*) + j * sizeof(int).`

Since we're doing matrix[i] instead of matrix + i, wouldn't matrix[i] + j = *_(matrix + i) + j? Because if we're doing pointer arithmetic with arrays it automatically multiplies by_ the size of the element of the array?

Anonymous Pig 12mth #1044aaee

Replying to Anonymous Pig

So matrix[i] + j basically first retrieves the pointer at the memory address matrix + i * sizeof(*int), then it goes to the memory pointer to by this pointer, then returns the address sizeof(int) * j to the right of that?

Nikhil Kandkur STAFF 12mth #1044aafa

Replying to Anonymous Pig

Yup!

Anonymous Grasshopper 12mth #1044cab ✓ Resolved

SP23-Final-Q2

In RISC-V, which are the fastest and slowest instructions (I believe slowest is load word)? Typically, how can we know which instruction is the fastest or the slowest? Is it just based on how it adds or removes time to the critical path?

For example, is lw the slowest for a 2-stage datapath because it includes reading from memory?

Also, for a typical 5-stage datapath (like that shown on the reference card), which stage is the fastest and which is the slowest? I assume that the slowest would be the MEM stage of lw, because reading from DMEM takes the longest amount of time, but is this correct?

Lastly, can pipelining make some random instruction A faster than instruction B even if B was faster in single-cycle?

Thank you!

1

Nikhil Kandkur STAFF 12mth #1044efa

1) The fastest and slowest instructions are dependent on how much combinational logic is needed to get the correct value. For example, in the 2-stage pipeline featured in the problem, the `add` instruction (in the 2nd stage) would only make use of A/B Muxes, ALU, and WBMux, so we would only add those logic delay times in our path for the `add` instruction. With that in mind, yes `lw` would be the slowest for a 2-stage datapath since out of all the instructions, it makes use of the most combinational logic.

2) It really depends on the times for each of the combinational logics, but in the real world it would usually be MEM. Even then, just pay attention to the timings for each of the combinational logics (mux, ALU, immediate generator, etc.), and once you get the critical path of each stage, you might see that the the longest stage may not necessarily be MEM.

3) Pipelining would not make an instruction faster than it was in a single cycle datapath, due to the fact that each stage would have to take as long as the longest stage, which in fact would increase the latency of completing one WHOLE instruction.

Anonymous Manatee 12mth #1044bec ✓ Resolved

SU23-Final-Q8.1:

I don't understand why we subtract 2 (pre-bias) from 2^4 to get the largest non-infinity/NaN exp value? How did we get the 2 from?

Andrew Liu STAFF 12mth #1044bfc

The largest value in 4 bits is 0b1111 = 2^4 - 1, however this corresponds to Infinity/NaN, so the largest normal exponent value is 0b1110 = 2^4 - 2.

1

Anonymous Flamingo 12mth #1044beb ✓ Resolved

SU23-Final-Q6.6

Would < work as well here?

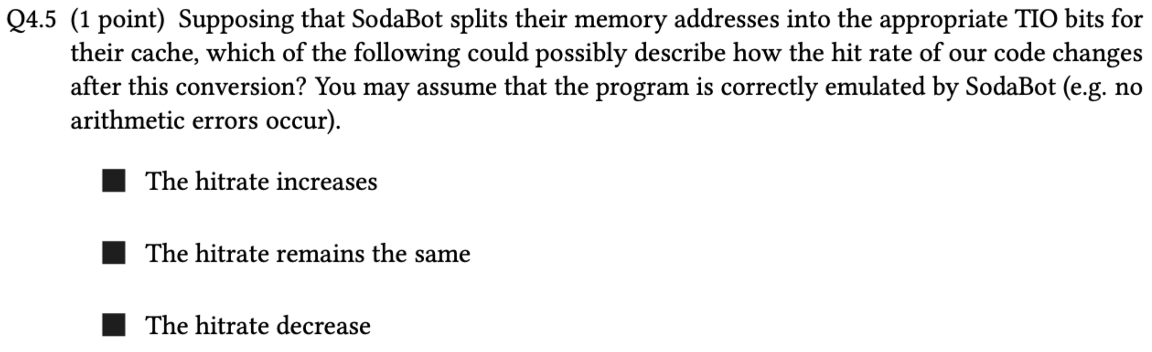
Jero Wang STAFF 12mth #1044adaa

Only doing < would not work, you would end up with an off-by-one error.

Anonymous Sheep 12mth #1044bdf ✓ Resolved

SU23-Final-Q4.5

Can you guys explain the cases for each? Thank you



Q4.5 (1 point) Supposing that SodaBot splits their memory addresses into the appropriate TIO bits for their cache, which of the following could possibly describe how the hit rate of our code changes after this conversion? You may assume that the program is correctly emulated by SodaBot (e.g. no arithmetic errors occur).

■ The hitrate increases

■ The hitrate remains the same

■ The hitrate decrease

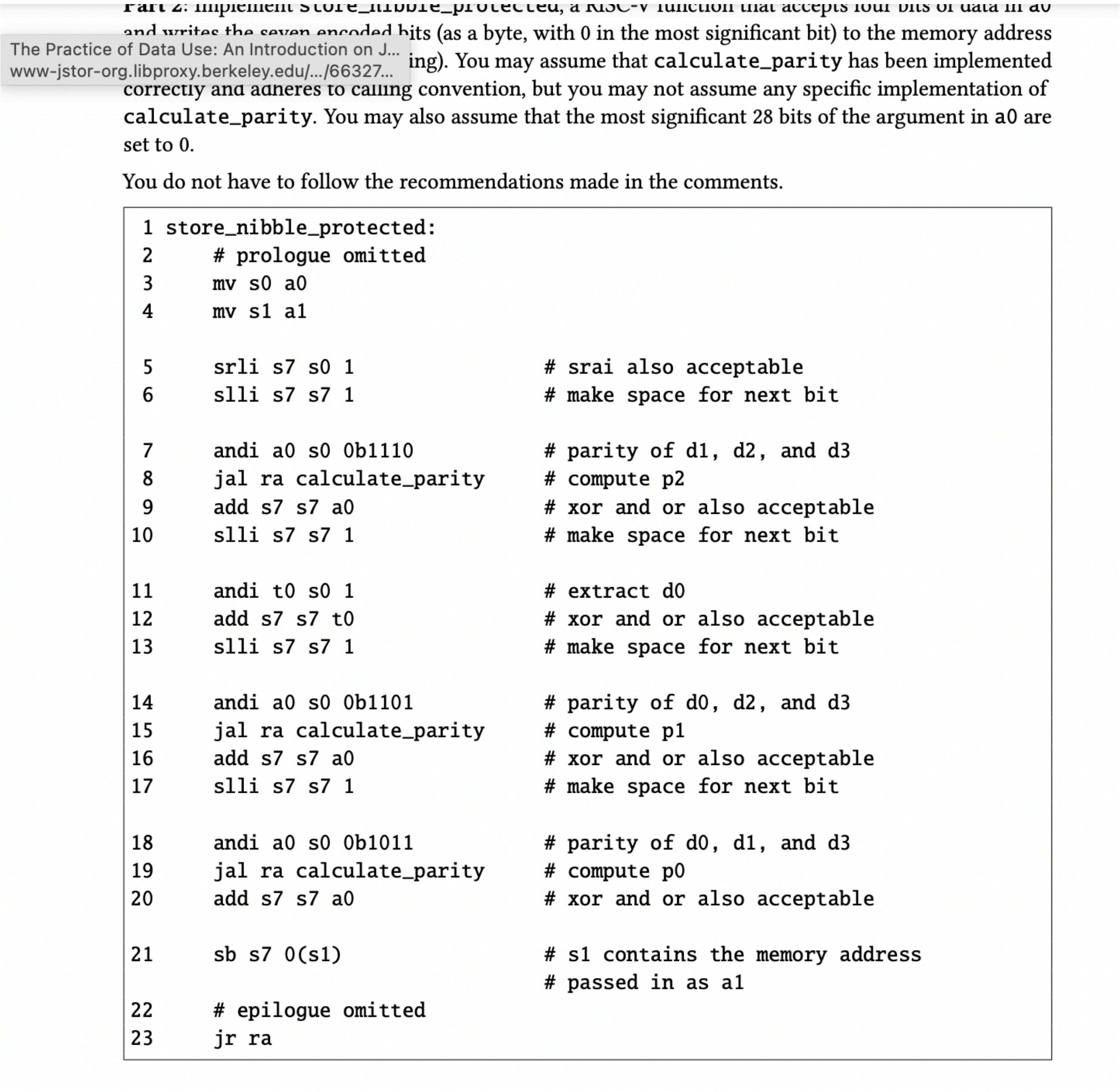
Anonymous Flamingo 12mth #1044bea

#1044abf

Anonymous Caterpillar 12mth #1044bdb ✓ Resolved

SU23-Final-Q2.2

Would lines 5-6 just get rid of the original LSB? I don't get why we are doing that, and in general I just don't understand what's going on with this problem. Also in line 7, what is going on with parity of d1, d2, d3 and what's the point of doing an andi operation with s0? How does using andi in line 11 extract d0?



Part 2: Implement `store_nibble_protected`, a RISC-V function that accepts four bits of data in a0 and writes the seven encoded bits (as a byte, with 0 in the most significant bit) to the memory address

The Practice of Data Use: An Introduction on J...
www-jstor-org.libproxy.berkeley.edu/.../66327...

ing). You may assume that `calculate_parity` has been implemented correctly and adheres to calling convention, but you may not assume any specific implementation of `calculate_parity`. You may also assume that the most significant 28 bits of the argument in a0 are set to 0.

You do not have to follow the recommendations made in the comments.

```
 1 store_nibble_protected:
 2     # prologue omitted
 3     mv s0 a0
 4     mv s1 a1

 5     srli s7 s0 1                 # srai also acceptable
 6     slli s7 s7 1                 # make space for next bit

 7     andi a0 s0 0b1110            # parity of d1, d2, and d3
 8     jal ra calculate_parity      # compute p2
 9     add s7 s7 a0                 # xor and or also acceptable
10     slli s7 s7 1                 # make space for next bit

11     andi t0 s0 1                 # extract d0
12     add s7 s7 t0                 # xor and or also acceptable
13     slli s7 s7 1                 # make space for next bit

14     andi a0 s0 0b1101            # parity of d0, d2, and d3
15     jal ra calculate_parity      # compute p1
16     add s7 s7 a0                 # xor and or also acceptable
17     slli s7 s7 1                 # make space for next bit

18     andi a0 s0 0b1011            # parity of d0, d1, and d3
19     jal ra calculate_parity      # compute p0
20     add s7 s7 a0                 # xor and or also acceptable

21     sb s7 0(s1)                  # s1 contains the memory address
                                    # passed in as a1

22     # epilogue omitted
23     jr ra
```

♡ 5 ···

Anonymous Pigeon 12mth #1044dcd

same question^^

♡ 1 ···

J Jero Wang STAFF 12mth #1044acff

This question works to create the codeword from the MSB to the LSB. If we look at the Hamming code chart, we can see that d3,d2,d1 are the upper 3 bits of the 7 bit codeword. Therefore, we get rid of the LSB of the input, so that now we have d3,d2,d1,0, and we can replace the 0 (the LSB of s7) with p2, which is the next bit in the codeword.

Then, for each of the next 3 bits (d0, p1, p0), we shift left by 1 to make space for the new bit, calculate the bit, and put the bit in the LSB.

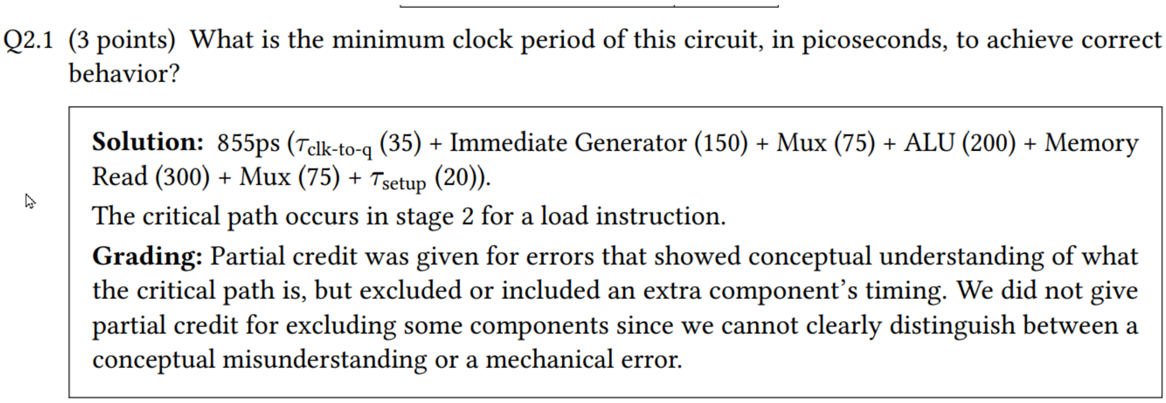
Line 7: We need to calculate p2, which is the XOR of d1, d2, and d3, so that's why we use the AND to get those 3 bits.

Line 11: As mentioned above, you need to extract d0, so you AND the original data to get its LSB, which is d0.

Anonymous Oryx 12mth #1044bcd ✓ Resolved

SP23-Final-Q2.1

Why do we have to account for immediate generator when it is not in stage 2?



Q2.1 (3 points) What is the minimum clock period of this circuit, in picoseconds, to achieve correct behavior?

Solution: 855ps ($\tau_{\text{clk-to-q}}$ (35) + Immediate Generator (150) + Mux (75) + ALU (200) + Memory Read (300) + Mux (75) + τ_{setup} (20)).

The critical path occurs in stage 2 for a load instruction.

Grading: Partial credit was given for errors that showed conceptual understanding of what the critical path is, but excluded or included an extra component's timing. We did not give partial credit for excluding some components since we cannot clearly distinguish between a conceptual misunderstanding or a mechanical error.

Nikhil Kandkur STAFF 12mth #1044bdd

#1044bb

Anonymous Heron 12mth #1044bca ✓ Resolved

SP23-Final-Q2

--

For 2.1, I am confused on where to put the registers when pipelining between IF/ID and EX/MEM/WB. I thought we would put our registers after the imm gen and regfile but this answer has the longest path as being from IMM GEN -> B_MUX -> ALU -> MEM READ -> WB_MUX -> REGISTER?

Nikhil Kandkur STAFF 12mth #1044fcc

#1044bb

Anonymous Squid 12mth #1044bbe ✓ Resolved

spring 23 q3

for Q3.13: ans should be (thread_num + 1) * chunk_size. Can i do start_row+chunk_size instead?

3

Anonymous Heron 12mth #1044bcb

I think these are the same thing so I think it should be fine:

start_row = thread_num * chunk_size

end_row = start_row + chunk_size

end_row = thread_num * chunk_size + chunk_size end_row = chunk_size * (thread_num + 1)

♡ 1 ···

Anonymous Quail 12mth #1044addc

shouldn't it be start_row+chunk_size -1?

♡ ···

N Nikhil Kandkur STAFF 12mth #1044ecf

Doing `start_row + chunk_size` would work!

♡ 1 ···

Anonymous Squid 12mth #1044bbd ✓ Resolved

spring 23 q3,

for Q3.4: can i do vec_load(matrix[i][j]) instead of vec_load(matrix[i] + j)

♡ 1 ···

Anonymous Heron 12mth #1044caa

I don't think so since matrix[i] is a double pointer and matrix[i][j] is a double! Maybe &matrix[i][j] might work?

♡ ···

Anonymous Elk 12mth #1044ceb

i have the same question because they accessed matrix[i][j] to add it to the global sum on line 10.

♡ ···

C Catherine Van Keuren STAFF 12mth #1044dfa

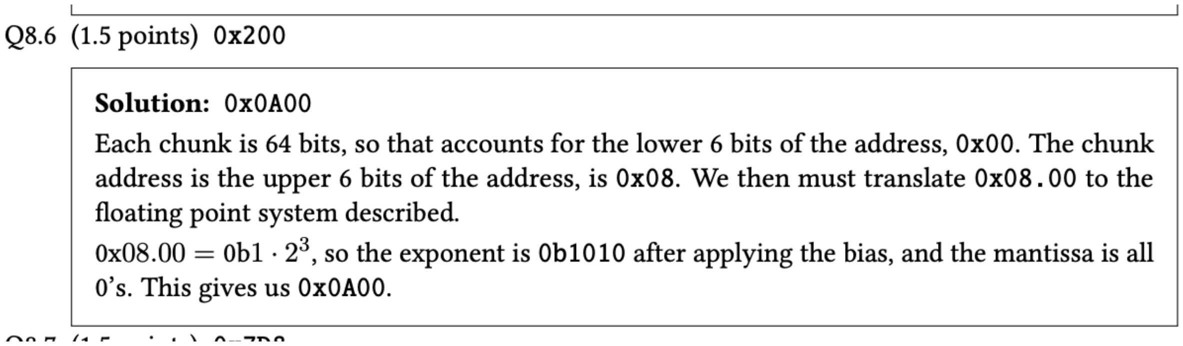
#1044abb

♡ ···

Anonymous Red deer 12mth #1044bbc ✓ Resolved

SU23-Final-Q8.6

How did we know that the lower corresponds to which chunk it is and that the upper is the chunk address? And that they are 6 bits each? I was going to just convert these numbers to decimal and the convert the decimal to the new FP system.



Q8.6 (1.5 points) 0x200

Solution: 0x0A00

Each chunk is 64 bits, so that accounts for the lower 6 bits of the address, 0x00. The chunk address is the upper 6 bits of the address, is 0x08. We then must translate 0x08.00 to the floating point system described.

$0x08.00 = 0b1 \cdot 2^3$, so the exponent is 0b1010 after applying the bias, and the mantissa is all 0's. This gives us 0x0A00.

♡ 5 ···

Anonymous Swallow 12mth #1044efb

yeah i was wondering this too, how do we know that the lower address is 6 bits?

♡ ···

J

Jero Wang STAFF 12mth #1044acfd

I think that should say each byte is 64 bytes, which means that it's addressable by the lower 6 bits.

Anonymous Monkey 12mth #1044adee

Do you mind clarifying this and walk through the solution? I'm a bit confused with the chunks and lower bits as well as the floating system for this specific problem. How did we get 0x08.00 and why do we take the lower 6 bits? Could you also clarify the lower and upper chunk adress

2

J

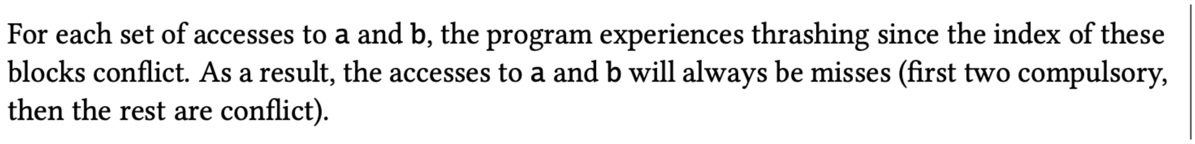
Jero Wang STAFF 12mth #1044afdb

Replying to Anonymous Monkey

`0x200 = 0b0010 0000 0000`. Since we have 64 bytes in each chunk, we have 6 bit "offsets" for each chunk. In the FP representation, offsets are represented as the bits after the decimal point, while the chunk index is represented as the bits before the decimal point. This gives us the FP number `0b001000.000000 = 0x08.00`.

Anonymous Red deer 12mth #1044bbb ✓ Resolved

sp23-final 4.2



For each set of accesses to a and b, the program experiences thrashing since the index of these blocks conflict. As a result, the accesses to a and b will always be misses (first two compulsory, then the rest are conflict).

I am slightly confused as to why the first two misses are compulsory. Isn't only the first access compulsory since the access to a is compulsory, but b shares the same index as a. I think we have two compulsory misses -- one for the first access to a, and the second from the access to result array, right?

N

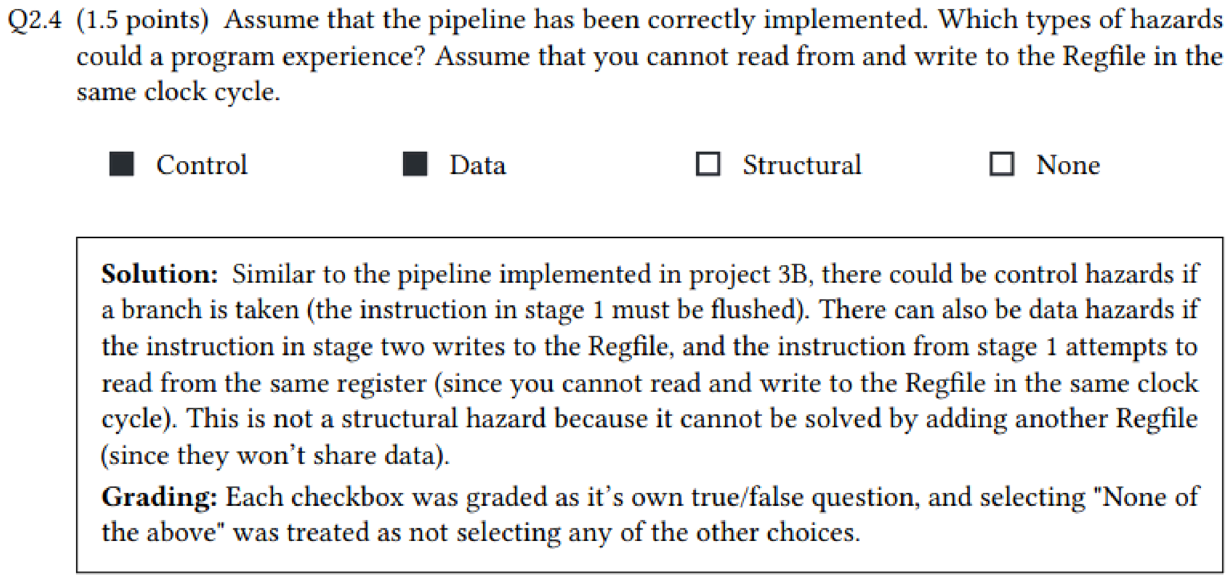
Nikhil Kandkur STAFF 12mth #1044fcd

Yeah I think that's what they meant by the first two compulsory: the first access to each to `a` and `b` are compulsory = the first two accesses in the cache are compulsory.

Anonymous Duck 12mth #1044bba ✓ Resolved

SP23-Final-Q2

Why is there no possibility of a structural hazard? If both the WB and ID read/write to the same shared resource, which is not possible, wouldn't that be the definition of a structural hazard, where more than one instruction needs the same datapath resource? I am confused what it means that it is not a structural hazard because it cannot be solved by adding another Regfile.



Q2.4 (1.5 points) Assume that the pipeline has been correctly implemented. Which types of hazards could a program experience? Assume that you cannot read from and write to the Regfile in the same clock cycle.

■ Control ■ Data □ Structural □ None

Solution: Similar to the pipeline implemented in project 3B, there could be control hazards if a branch is taken (the instruction in stage 1 must be flushed). There can also be data hazards if the instruction in stage two writes to the Regfile, and the instruction from stage 1 attempts to read from the same register (since you cannot read and write to the Regfile in the same clock cycle). This is not a structural hazard because it cannot be solved by adding another Regfile (since they won't share data).

Grading: Each checkbox was graded as it's own true/false question, and selecting "None of the above" was treated as not selecting any of the other choices.

♡ 3 ···

Anonymous Pig 12mth #1044bde

I have the same question

♡ ···

Anonymous Grasshopper 12mth #1044bfe

same question.

♡ ···

N

Nikhil Kandkur STAFF 12mth #1044bff

Structural hazards will occur if our datapath requires more ports than available on a certain resource to work. For example, in the real world, IMEM and DMEM are in the same memory, but there are 2 different ports that will be used for different areas of memory (one for IMEM and one for DMEM).

For our regfile, we will still have the 2 read ports and 1 write port, which will allow us to write and read to/from the same resource, so we will not have a case where we need more ports than there already are. The line "this is not a structural hazard because it cannot be solved by adding another Regfile" refers to the fact that any hazards we get will not be solved by adding additional ports, since the hazards we have do not stem from having limited ports.

Hope this helps!

♡ 1 ···

Anonymous Clam 12mth #1044cfc

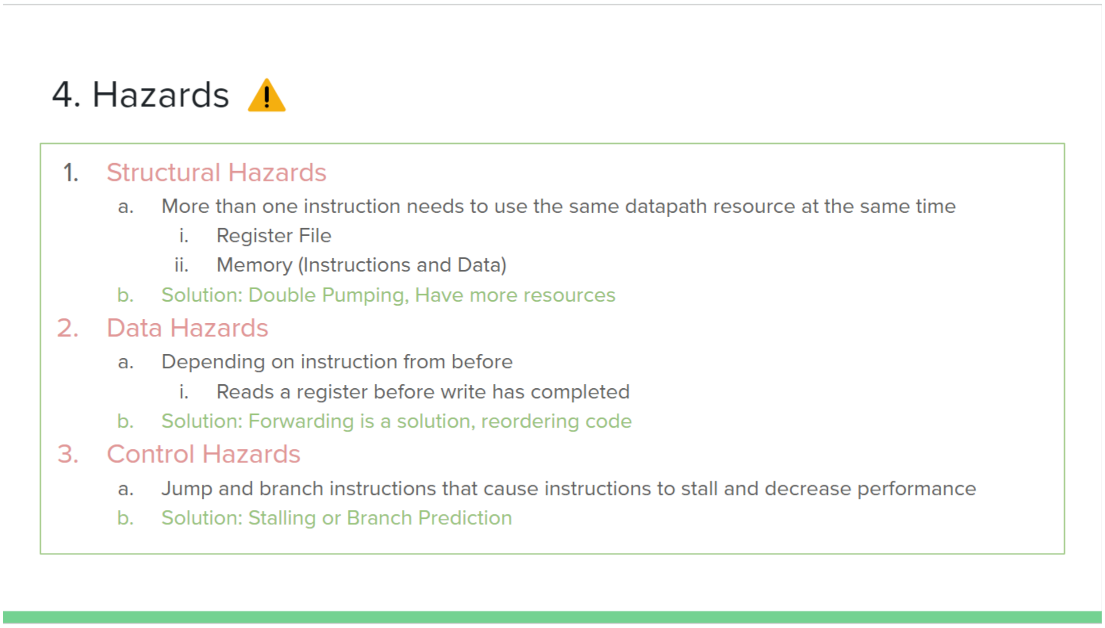
Edit: I initially asked a follow up question but I came across this from the discussions and it helped:

> Through adding additional hardware, we can implement two 'read' ports as well as a 'write' port to the RegFile (where registers can be accessed). This solves the hazard of two instructions reading and writing to the same register simultaneously. False. The addition of independent ports to the RegFile allows for multiple instructions to access the RegFile at the same time (such as one instruction reading values of two operands, while another instruction is writing to a return register). However, this does not work if both instructions are reading and writing to the same register. Some solutions to this data hazard could be to stall the latter instruction by 1 cycle or to forward the read value from a previous instruction, bypassing the RegFile completely

♡ 1 ···

Anonymous Wildcat 12mth #1044aebe

This makes sense but now I'm confused why double pumping is considered a solution to a structural hazard. It sounds like having enough ports to read and write in the same clock cycle solves our hazard and not the actual act of reading and writing in the same clock cycle. Is double pumping more for a data hazard then? The screenshot was taken from review session. Thank you!



4. Hazards ⚠

1. Structural Hazards
   a. More than one instruction needs to use the same datapath resource at the same time
      i. Register File
      ii. Memory (Instructions and Data)
   b. Solution: Double Pumping, Have more resources
2. Data Hazards
   a. Depending on instruction from before
      i. Reads a register before write has completed
   b. Solution: Forwarding is a solution, reordering code
3. Control Hazards
   a. Jump and branch instructions that cause instructions to stall and decrease performance
   b. Solution: Stalling or Branch Prediction

♡ ···

Anonymous Jay 12mth #1044baf ✓ Resolved

SP23-Final-Q7

I'm confused on how the code account for splitting the codecopy into the array of ints it's supposed to. For example if the first instruction is a jump and data[0] points to codecopy and you set *codecopy to 0, does data[0] still not point to the whole array of codecopy?

♡ 1 ···

N Nikhil Kandkur STAFF 12mth #1044eee

Yes data[0] does point to the whole array, but remember, this was specified in the question description:

> Each of these arrays should be "null-terminated"; that is, the last element of each array should be the number 0, to signify the end of the array.

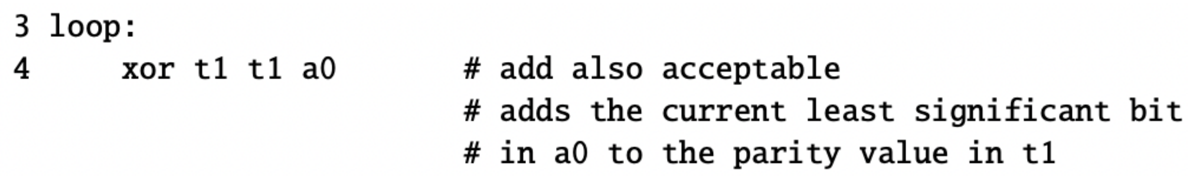
So just setting our value to 0 in the array will count as ending the current array.

♡ ···

Anonymous Caterpillar 12mth #1044bad ✓ Resolved

SU23-Final-Q2.1

How does xor add the LSB of a0 to t1? For example, if t1 = 0 and a0 = 0b 0000 0000 1101, wouldn't t1 end up being 0b 0000 0000 1101 and not the LSB of just 1?



```
loop:
    xor t1 t1 a0        # add also acceptable
                        # adds the current least significant bit
                        # in a0 to the parity value in t1
```

♡ 1 ···

Anonymous Ferret 12mth #1044bae

Yes, for the first iteration of the loop, t1 will be 0b 0000 0000 1001. However, notice that in Q2.2 we are shifting a0 to the right by one bit after the XOR for each iteration of the loop. Therefore, after k iterations of the loop, the LSB of t1 will be the XOR result of the least significant k bits of a0. If we loop over all digits of a0, the LSB of t1 will in effect record the parity of a0 (recall that XOR-ing all bits of a bit sequence tells us if there are even/odd number of 1's).

♡ 1 ···

Anonymous Pheasant 12mth #1044bac ✓ Resolved

SU23-Final-Q4.2: I am confused where these numbers come from. how do you know that there are 3 entries total and that there are 9 trytes?

Screen Shot 2023-12-05 at 1.36.20 PM.png

♡ ···

Nikhil Kandkur STAFF 12mth #1044bda

#1044ab

♡ ···

Anonymous Wren 12mth #1044afe ✓ Resolved

Spring 23: Q2: 1

I am a little confused on where to put each register. I assumed that if we were going to split the stage into 2. One for fetching and decoding that the stage 1 registers would all contain the decoded information. Such as a register for register 1 data, a register for register 2 data, a register after the immediate generator decodes the instruction, and a PC. That would enable us to decode the instruction to let the second stage do its job. However based on 2.2 it seems that 2.1 assumed that the immediate generator was in stage 2, which does not make sense to me since the ID stage is meant to decode all the information in the instruction and is the job of stage 1. Can someone help me understand where each register should be placed to help further my understanding of pipelining?

♡ 1 ···

Anonymous Wren 12mth #1044bab

To clarify my question: Immediate generators dont seem to be a part of the execute stage, so why is it that 2.1 placed it in stage 2 to begin with. Is my understanding of stages and their components wrong?

♡ 1 ···

Anonymous Heron 12mth #1044bbf

I was also super confused on this :(

♡ ···

N Nikhil Kandkur STAFF 12mth #1044bcf

#1044bb

♡ ···

Anonymous Pheasant 12mth #1044acf ✓ Resolved

Summer 23 Q3.10: i thought 2^20 is the max number of bits the instruction can jump by so to get the max number of bytes you would divide by 8 to get an answer of 2^20/8. Could someone explain the correct answer to me and why 2^20 represents the number of bytes?

Screen Shot 2023-12-05 at 11.04.10 AM.png

♡ ···

J Jero Wang STAFF 12mth #1044aea

Please follow the convention for tagging your questions (e.g. SU23-Final-Q3.10)!

The memory system is byte-addressable, so you can't jump to a specific bit. J-types have 21 bits of immediate, allowing us to jump from -2^{20} to $2^{20}-2$.

♡ ···

Anonymous Lion 12mth #1044fec

I'm still a little confused when you say "The memory system is byte-addressable, so you can't jump to a specific bit." How is 2^20 the max number of bytes and not bits?

♡ ···

Anonymous Lemur 12mth #1044acd ✓ Resolved

sp23-final-6: is question 6.1/6.2 in scope? how do you recommend we learn all this? thx.

♡ ···

J Jero Wang STAFF 12mth #1044aeb

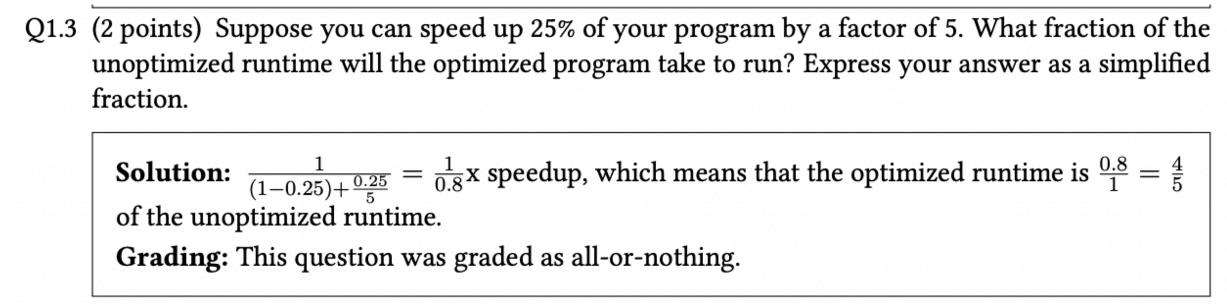
Yes, these were covered in lecture.

♡ ···

Anonymous Caterpillar 12mth #1044acb ✓ Resolved

SU23-Final-Q1.3

How do we get from 1/0.8 times speedup to 0.8 as the final answer?



Q1.3 (2 points) Suppose you can speed up 25% of your program by a factor of 5. What fraction of the unoptimized runtime will the optimized program take to run? Express your answer as a simplified fraction.

Solution: $\frac{1}{(1-0.25)+\frac{0.25}{5}} = \frac{1}{0.8}$x speedup, which means that the optimized runtime is $\frac{0.8}{1} = \frac{4}{5}$ of the unoptimized runtime.

Grading: This question was graded as all-or-nothing.

♡ 2 ···

B Byron Yang 12mth #1044ace

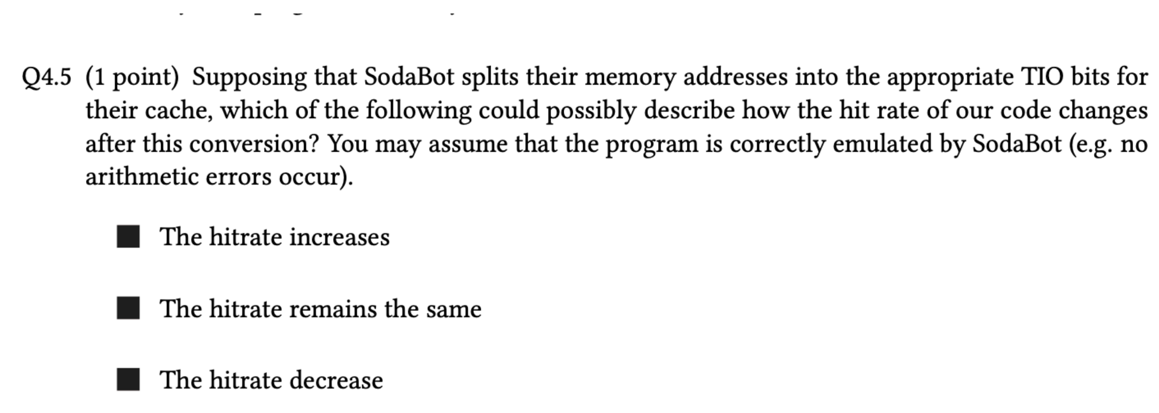
The idea is that if we have x times speedup, our new runtime is $\frac{1}{x}$ of our original runtime. For example, if we have a 2 times speedup ($x=2$), then we can do the task 2 times as fast, which is equivalent to only taking $\frac{1}{2}$ of the total time.

♡ 1 ···

Anonymous Red deer 12mth #1044abf ✓ Resolved

Su23-Final-Q4.5

why is this true?



Q4.5 (1 point) Supposing that SodaBot splits their memory addresses into the appropriate TIO bits for their cache, which of the following could possibly describe how the hit rate of our code changes after this conversion? You may assume that the program is correctly emulated by SodaBot (e.g. no arithmetic errors occur).

■ The hitrate increases

■ The hitrate remains the same

■ The hitrate decrease

♡ 1 ···

Jero Wang STAFF 12mth #1044aff

The hit rate can increase since the SodaBot's blocks are larger than CoryBot's blocks. The hit rate can decrease since the addresses each block contains are different, so what used to be a hit (because a block was loaded in from a previous access), may now be a miss since the memory address belongs to a different block. If everything goes right and neither of these scenarios happen, the hit rate remains the same.

♡ ···

Anonymous Red deer 12mth #1044baa

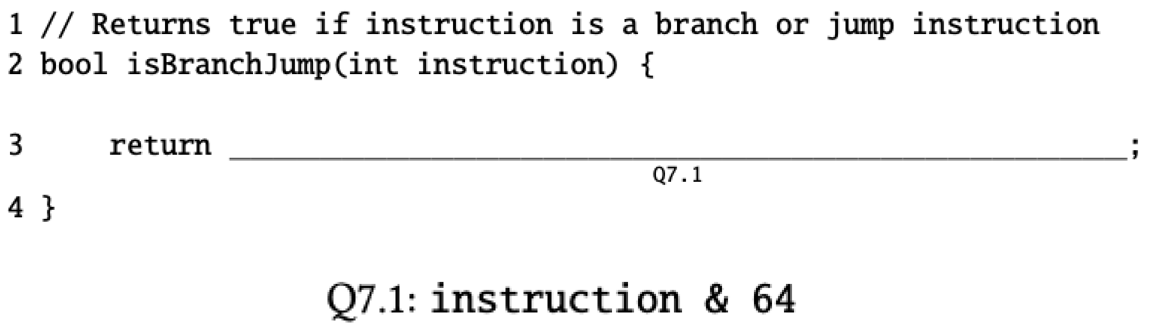
thanks jero!

♡ ···

Anonymous Tapir 12mth #1044abe ✓ Resolved

SP23-final-Q7.1

Why does instruction & 64 work?



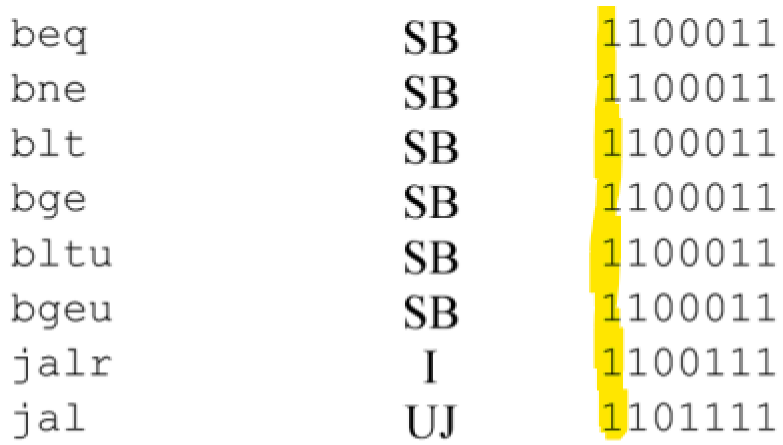
```
// Returns true if instruction is a branch or jump instruction
bool isBranchJump(int instruction) {

    return ____________________________________________________;
                                Q7.1
}
```

Q7.1: instruction & 64

♡ ···

Farzan Hashmi 12mth #1044acc ENDORSED

If you look at the reference card, you can notice that the branch and jump instructions are the only instructions that have the 6th bit of the opcode (and consequently the 6th bit of the entire instruction) set to 1. 64 in binary is 0b1000000 and by performing an & with it we are specifically checking whether that 6th bit of the instruction is set to 1.



beq	SB	1100011
bne	SB	1100011
blt	SB	1100011
bge	SB	1100011
bltu	SB	1100011
bgeu	SB	1100011
jalr	I	1100111
jal	UJ	1101111

♡ 2 ···

Anonymous Red deer 12mth #1044bed

it seems like ebreak and ecall have 1 as the 6th bit as 1. shouldn't we check if the 6th 5th and 4th bit = 110 respectively?

♡ ···

F Farzan Hashmi 12mth #1044bee

Replying to Anonymous Red deer

ah the problem specifically states that the input won't contain ecalls or ebreaks.

♡ ···

Anonymous Red deer 12mth #1044bfb

Replying to Farzan Hashmi

omg ty could we also write instruction & 0b1000000?

♡ ···

Anonymous Cod 12mth #1044acea

Replying to Farzan Hashmi

What does the output of this look like? If the 6th bit is indeed 1, does that mean the output is 0b1000000? How exactly does this result in True?

♡ 5 ···

Anonymous Chamois 12mth #1044aeeb

Replying to Anonymous Cod

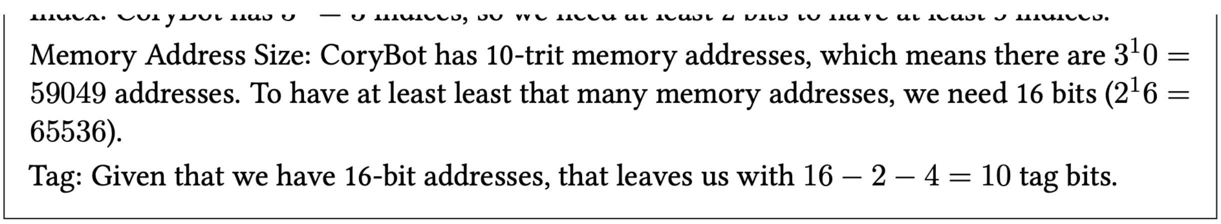
I guess you could potentially do (instruction & 64) == 64

♡ ···

Anonymous Red deer 12mth #1044abc ✓ Resolved

SU23-Final-Q4.4

how were we supposed to know that 2^16 > 3^10?



Index: CoryBot has $3^1 = 3$ indices, so we need at least 2 bits to have at least 3 indices.

Memory Address Size: CoryBot has 10-trit memory addresses, which means there are $3^10 = 59049$ addresses. To have at least least that many memory addresses, we need 16 bits ($2^16 = 65536$).

Tag: Given that we have 16-bit addresses, that leaves us with $16 - 2 - 4 = 10$ tag bits.

♡ ···

J Jero Wang STAFF 12mth #1044adf

Powers of 3 were provided, and you should know/be able to calculate 2^16 since its a power of 2.

♡ ···

Xiaole Guo 12mth #1044aad ✓ Resolved

sp23-final-Q6.4/6.5

Are these two questions covered for our semester?

♡ ···

Jero Wang STAFF 12mth #1044aec

They should be in scope this semester as well.

♡ ···

Anonymous Skunk 12mth #1044cdd

Do you know what lectures this content was covered?

♡ 3 ···

Anonymous Sand Dollar 12mth #1044aaa ✓ Resolved

Sp23-Final-Q2.1:

Why would the branch comparator not be part of the components in our critical path? How do we identify the components that will be counted in our critical path?

♡ 1 ···

Catherine Van Keuren STAFF 12mth #1044aaf

Typically when we determine what our critical path will be, we try to find the path with the highest combinatorial logic delay. In this case, the longest path is a load instruction since it utilizes all three steps in EX/MEM/WB (which make up the second stage in this question). Load doesn't use the branch comparator, so it is unnecessary to include it in the critical path. Please follow up if you have any more questions!

♡ ···

Anonymous Caterpillar 12mth #1044cfd

Why are there two muxes? Doesn't the load instruction only pass through one?

♡ 1 ···

Catherine Van Keuren STAFF 12mth #1044cfe

Replying to Anonymous Caterpillar

The two muxes are the ASel/BSel mux (count as one since they happen in parallel) and the WBSel mux since we need to select that we're writing back what we just loaded from memory.

♡ 1 ···

Anonymous Caterpillar 12mth #1044cff

Replying to Catherine Van Keuren

Makes sense, thanks!

♡ 1 ···

Anonymous Deer 12mth #1044ff ✓ Resolved

SP23-Final-Q4.2:

it is the tags of a and b that are conflicting right? since the index of both a and b is 0b00.

♡ 1 ···

Jero Wang STAFF 12mth #1044aed

Yes

♡ ···

Anonymous Lemur 12mth #1044fd ✓ Resolved

Sp23-Final-Q2:
is the only approach for this question to like test every possible combination? I struggle to know which will be activiated and what the critical path is? Like how are you supposed to know its a load words, and that there isn't a longer combibnation?

also 2- do you include the pipeline timing? They include the clq-to-q, but not setup time at the start for the piplined register? why is that?
thanks

♡ ···

Jero Wang STAFF 12mth #1044aee

Generally, it's enough to walk through one instruction from each instruction type (e.g. all R-type are the same, except different ALUSel). I'd recommend looking for similar past exam questions, and you'll gain more intuition with more practice.

Since this is the second stage, there are no more pipeline registers. Setup time is for when you write to another register.

♡ ···

Anonymous Skunk 12mth #1044fc ✓ Resolved

SP23-Final-Q4.1: why is the tag `0b00_1100_0000`? My steps are as follows, not sure where I'm going wrong:

```
0x3037 = 0011 0000 0011 0111
```

Split into 10-b tag, 2-bit index, 4-bit offset: `0011 0000 00`, `11`, `0111`.

♡ ···

Anonymous Sand Dollar 12mth #1044aac ENDORSED

What you have and what the solution has are the same thing!

♡ 1 ···

Anonymous Skunk 12mth #1044cdb

I don't know how I didn't realize this... thank you lol

♡ ···

Anonymous Red deer 12mth #1044ef ✓ Resolved

SU23-Final-Q1.5. During a thread context switch, the entries of the TLB get invalidated.

I thought this only happens when a new process is happening not thread, don't threads on the same CPU share a page table/TLB?

♡ 2 ···

Anonymous Pig 12mth #1044bdc

I have the same question

♡ ···

Justin Yokota STAFF 12mth #1044eaf

The CPU can't distinguish between a context switch between threads of the same process and threads of different processes. So to be safe it needs to invalidate the TLB.

♡ 1 ···

San Vu 12mth #1044ee ✓ Resolved

For SP23 2.1 Why do we factor int he immediate generator? isnt that part of stage 1(ID phase)? we are just measuring phase 2.

♡ 2 ···

Nikhil Kandkur STAFF 12mth #1044fe

#1044bb

♡ 1 ···

Anonymous Pheasant 12mth #1044ed ✓ Resolved

For SP23 Final Q5.4, why is there a page fault? Since the VPN is 0x000003 I thought the page table has an entry for the index 0x3 which maps to 0X3561CBA8? The solutions say "There are no valid PTEs for this VPN, therefore, it is a page fault." Also, since the physical address is 0x61DE00 why is the PPN not the bottom 16 bits 0xDE00 instead of 0X61DE?

♡ 1 ···

Anonymous Tapir 12mth #1044abd ENDORSED

I think it is because bit 30 is the valid bit and 0x3 = 0011, so this is an invalid PTE

♡ 1 ···

Jero Wang STAFF 12mth #1044aef

^ What Anon. Tapir mentioned, except bit 31.

PPN is always the most significant bits, and offset is always the least significant bits.

♡ 1 ···

Anonymous Pheasant 12mth #1044ec ✓ Resolved

for SP 23 Final Q4.2, why are they cache conflicts instead of cache hits? I thought since all 4 integers of A are loaded into 1 cache block each of the accesses of A will be hits since those values are already loaded into the cache. I thought the same would happen for B with its 2 integers being in the same cache block. Can you explain where cache conflict comes from?

Screen Shot 2023-12-04 at 9.49.27 AM.png

♡ ···

Jero Wang STAFF 12mth #1044afa

The conflict comes from us needing to read `b` immediately after `a`, which would evict the `a` cache entry since both blocks have the same index.

♡ ···

Anonymous Caterpillar 12mth #1044dcc

how do they have the same index if b is indexed by j and b is indexed by i+j?

♡ 2 ···

Anonymous Chamois 12mth #1044aedf

Replying to Anonymous Caterpillar

Memory address of a is 0x1000. This means that its index in TIO is 0b00. Same for b which is at 0x2000.

♡ ···

Anonymous Pheasant 12mth #1044eb ✓ Resolved

For SP 23 Final Q3.4, in SIMD why do you have to do Q3.4: vec_load(matrix[i] + j) instead of vec_load(matrix[i][j])? In C they did matrix[i][j]. Can you not double index for vec_load?

♡ 1 ···

Anonymous Sandpiper 12mth #1044aab

I don't understand how vec_load(matrix[i] + j) works at all. Could anyone explain?

♡ ···

C Catherine Van Keuren STAFF 12mth #1044abb

There is a bit of a subtle difference between 3.4 and line 10 in the given code. For line 3.4, we are assigning values to be of type vector, meaning that we want to load in 4 values at once. Additionally, the parameter of vec_load() is a double*, meaning that we need to pass in a pointer. By using matrix[I] + j, we are indicating that we want to grab the 4 values at the memory address of that row (matrix[I]), but shifted over by j. For example, if we have a row with 8 values in it, and the memory address of the start of the row is 0x0, calling vec_load(matrix[I] + 0) would give us the first four elements starting at 0x0, then in the next iteration if we call vec_load(matrix[I] + 4), we're grabbing elements 5-8 starting at memory address 0x4. On the contrary, if we were to use matrix[I][j] in lieu of matrix[I] + j, we would actually be returning a singular double instead of a pointer to an array of doubles, which is not what we want when we call vec_load. I hope this helps clarify the difference, and please follow up if you have any more questions!

♡ 2 ···

Anonymous Pheasant 12mth #1044ea ✓ Resolved

Screen Shot 2023-12-04 at 8.12.02 AM.png

For Q2.3, if the immediate is now part of stage 1 since our reference card was updated would that be selected as an answer as well? Why does the instruction need to be put in a pipelined register if all the info from it is read by the regfile?

♡ ···

J Jero Wang STAFF 12mth #1044afb

Yes, it will need a pipeline register. Instruction needs to be put in a pipeline register because we need it in later stages to determine control logic signals (e.g. ASel, BSel, WBSel, etc).

♡ ···

Anonymous Seahorse 12mth #1044df ✓ Resolved

SP-23-Final-Q2.1, 2.2 (Conceptual)

Given the new info about ImmGen being part of ID, if we were to calculate the time for Stage 1 anyway, would it be T_c2q + RegFile Setup + RegFile Read + Imm Gen? Are there any components missing?

Given this info, what would 2.2 be? I was considering #1044d's (A+B Mux) response, but if ImmGen is now in ID and we need ImmGen's output for B Mux, is it still valid to add the Mux to

Stage 1 (just because it won't be part of the critical path, not bc it can run in parallel with any other component)?

♡ ···

Jero Wang STAFF 12mth #1044afc

You would also have Memory Read, since you're reading from IMEM. Immediate generator is not in the critical path (which why it doesn't really matter if it's in ID or EX), since it runs in parallel to RegFile Read, and RegFile Read > Imm Gen.

That would be moving two components, which is not allowed here. Moving only AMux or BMux would not affect the critical path. Given the updated datapath, I think the answer would be "Not Possible".

♡ 1 ···

Anonymous Cheetah 12mth #1044de ✓ Resolved

Fa20-Final-Potpourri-PartA

What does the qusetion mean? I couldn't understand the answer.

♡ ···

Catherine Van Keuren STAFF 12mth #1044aba

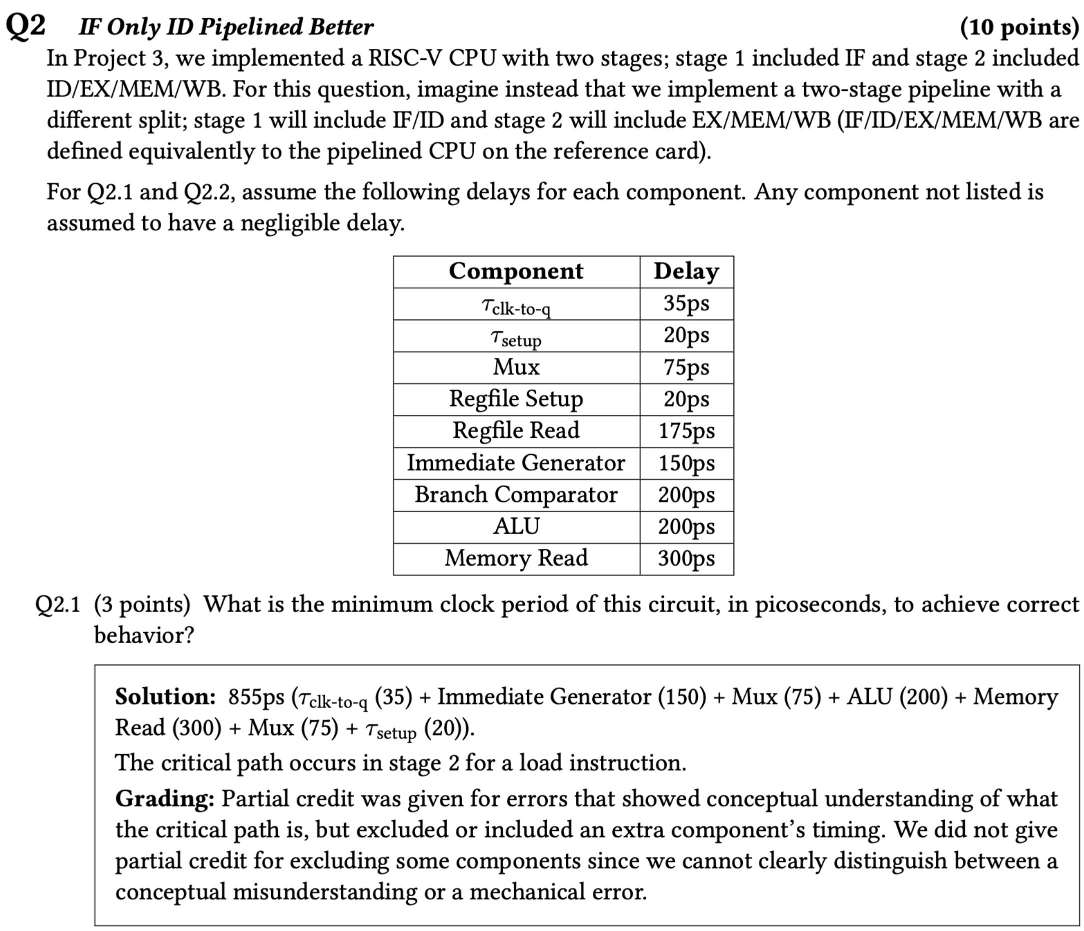
Please ask this on the correct thread!

♡ ···

Anonymous Lion 12mth #1044dc ✓ Resolved

Sp23-Final-Q2.1:

Why is the immediate generator included in the longest combinational path? I thought that the immediate generator is part of Instruction Decode, and shouldn't be included in stage 2?



Q2 ***IF Only ID Pipelined Better*** **(10 points)**

In Project 3, we implemented a RISC-V CPU with two stages; stage 1 included IF and stage 2 included ID/EX/MEM/WB. For this question, imagine instead that we implement a two-stage pipeline with a different split; stage 1 will include IF/ID and stage 2 will include EX/MEM/WB (IF/ID/EX/MEM/WB are defined equivalently to the pipelined CPU on the reference card).

For Q2.1 and Q2.2, assume the following delays for each component. Any component not listed is assumed to have a negligible delay.

Component	**Delay**
$\tau_{\text{clk-to-q}}$	35ps
τ_{setup}	20ps
Mux	75ps
Regfile Setup	20ps
Regfile Read	175ps
Immediate Generator	150ps
Branch Comparator	200ps
ALU	200ps
Memory Read	300ps

Q2.1 (3 points) What is the minimum clock period of this circuit, in picoseconds, to achieve correct behavior?

Solution: 855ps ($\tau_{\text{clk-to-q}}$ (35) + Immediate Generator (150) + Mux (75) + ALU (200) + Memory Read (300) + Mux (75) + τ_{setup} (20)).

The critical path occurs in stage 2 for a load instruction.

Grading: Partial credit was given for errors that showed conceptual understanding of what the critical path is, but excluded or included an extra component's timing. We did not give partial credit for excluding some components since we cannot clearly distinguish between a conceptual misunderstanding or a mechanical error.

♡ 1 ···

N Nikhil Kandkur STAFF 12mth #1044dd

#1044bb

♡ 1 ···

Anonymous Sand Dollar 12mth #1044da ✓ Resolved

Su23-Final-Q4.4

What does it mean by "CoryBot has a tryte-addressable, 10-trit memory system"? What would that look like? I thought since it's tryte-addressable, we could access each tryte (group of 3 trits) at a time, but I don't get how the 10 trits part influences that. Thank you!

♡ ···

J Jero Wang STAFF 12mth #1044ade

10-trit means that each memory address is 10 trits wide, similar to how a 32-bit address space would mean each address is 32-bits wide.

♡ ···

Anonymous Ferret 12mth #1044ce ✓ Resolved

SU23-Final-Q3.11

Is it also acceptable to wire the second output of IMEM directly to the BMux (and add a control in BSel accordingly) instead of having it pass through the Immediate Generator?

♡ ···

Jero Wang STAFF 12mth #1044add

Yes, that's a valid alternate solution for this question.

Anonymous Ferret 12mth #1044cd ✓ Resolved

SU23-Final-Q8

I feel like the wording for this question was quite confusing. From the first few parts, it seemed like the floating point chunk-addressing system was supposed to address chunks using whole numbers and the bytes within each chunk using decimal numbers (i.e. chunks have addresses 0, 1, 2, etc and bytes have addresses $\frac{1}{64}$, $\frac{2}{64}$, etc). However, from what I understand, for Q8.5 through Q8.7 we are asked to interpret the binary addresses as addresses for the bytes within chunks instead of chunk addresses even though they are whole numbers.

Which part of the question should be used to deduce that we are supposed to split the addresses for Q8.5 through Q8.7 into both whole number and decimal parts, instead of interpreting them directly as whole numbers?

Jero Wang STAFF 12mth #1044adc

They are whole numbers - they represent memory addresses in a traditional memory addressing system, and the question asks you to convert them into the address in this FP-based memory addressing system. We'll make sure this is clear on future exams.

Anonymous Ferret 12mth #1044bf ✓ Resolved

SU23-Final-Q6.9 contains a typo where it uses reverseVec instead of vecReverse as given in the problem statement.

Jero Wang STAFF 12mth #1044cb

Thanks for the catch, we've noted it on our end. Please also label your followups with the exam using the format we specified above.

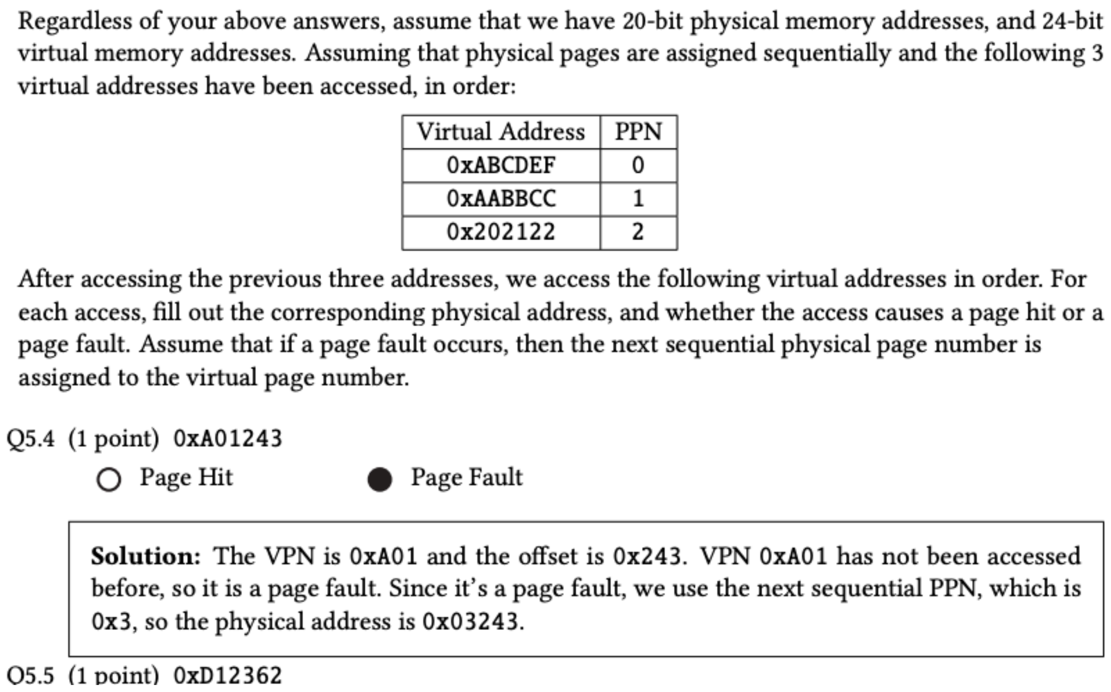
Anonymous Ferret 12mth #1044be ✓ Resolved

SU23-Final-Q5: For Q5.4 through Q5.9, is it fine if our answer omits the leading zeros in the physical address?

Jero Wang STAFF 12mth #1044ca

Generally for answers that require a specific bitwidth, no.

Anonymous Cassowary 12mth #1044ac ✓ Resolved



Regardless of your above answers, assume that we have 20-bit physical memory addresses, and 24-bit virtual memory addresses. Assuming that physical pages are assigned sequentially and the following 3 virtual addresses have been accessed, in order:

Virtual Address	PPN
0xABCDEF	0
0xAABBCC	1
0x202122	2

After accessing the previous three addresses, we access the following virtual addresses in order. For each access, fill out the corresponding physical address, and whether the access causes a page hit or a page fault. Assume that if a page fault occurs, then the next sequential physical page number is assigned to the virtual page number.

Q5.4 (1 point) 0xA01243

○ Page Hit ● Page Fault

Solution: The VPN is 0xA01 and the offset is 0x243. VPN 0xA01 has not been accessed before, so it is a page fault. Since it's a page fault, we use the next sequential PPN, which is 0x3, so the physical address is 0x03243.

Q5.5 (1 point) 0xD12362

SU23 Q 5.4, how do we know the amount of bits that are VPN and the amount of bits for offset? I dont see anywhere in the question that hints this?

Nikhil Kandkur STAFF 12mth #1044ad

Suppose we have a 16MiB physical memory, a 256MiB virtual memory space, and 4KiB pages.

Anonymous Monkey 12mth #1044abcb

It's a bit confusing when it tells to disregard the answer above? Since we derive the bits in page offset, PPN, and VPN from this description, wouldn't they also apply? How did we get 12 bits for VPN and 12 bits for offset? Can you help derive this from the description?

How do we know to use the information here or what is given to us in that question?

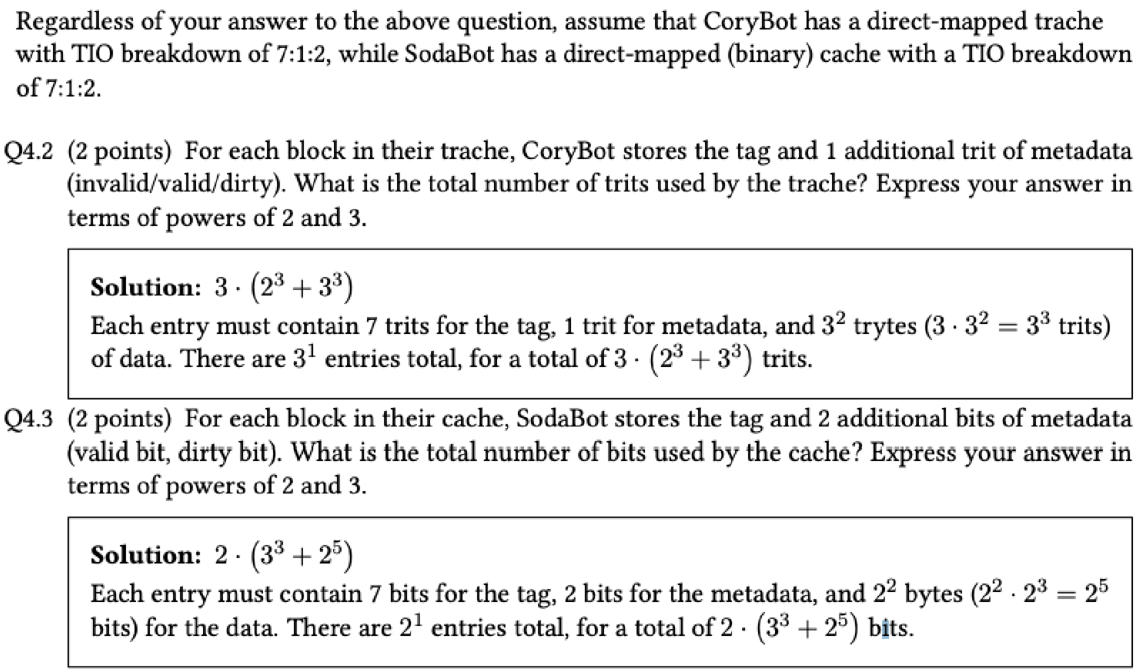
Nikhil Kandkur STAFF 12mth #1044abdb

Replying to Anonymous Monkey

I think we still need to assume 4 KiB pages, since that gives us 12 bits for offset.

Anonymous Cassowary 12mth #1044ab ✓ Resolved

SU-23 Final Q4.3



Regardless of your answer to the above question, assume that CoryBot has a direct-mapped trache with TIO breakdown of 7:1:2, while SodaBot has a direct-mapped (binary) cache with a TIO breakdown of 7:1:2.

Q4.2 (2 points) For each block in their trache, CoryBot stores the tag and 1 additional trit of metadata (invalid/valid/dirty). What is the total number of trits used by the trache? Express your answer in terms of powers of 2 and 3.

Solution: $3 \cdot (2^3 + 3^3)$
Each entry must contain 7 trits for the tag, 1 trit for metadata, and 3^2 trytes ($3 \cdot 3^2 = 3^3$ trits) of data. There are 3^1 entries total, for a total of $3 \cdot (2^3 + 3^3)$ trits.

Q4.3 (2 points) For each block in their cache, SodaBot stores the tag and 2 additional bits of metadata (valid bit, dirty bit). What is the total number of bits used by the cache? Express your answer in terms of powers of 2 and 3.

Solution: $2 \cdot (3^3 + 2^5)$
Each entry must contain 7 bits for the tag, 2 bits for the metadata, and 2^2 bytes ($2^2 \cdot 2^3 = 2^5$ bits) for the data. There are 2^1 entries total, for a total of $2 \cdot (3^3 + 2^5)$ bits.

Why do we have 3^3 instead of 3^2? Isn't 3^3 a count of the tag bits + metadata, which is 7 + 2?

Nikhil Kandkur STAFF 12mth #1044ae

We have 3^3 since we know that there are 3^2 trytes in each cache block based on the TIO breakdown, which is equal to 3 * 3^2 = 3^3 trits. The 2^3 comes from the 7 trits of tag + 1 trit of metadata = 2^3. This gives us (2^3 + 3^3) bits per cache block, and since we only have 1 trit for index, which can represent values 0 through 2, we have a total of 3 * (2^3 + 3^3) trits for the entire trache.

Anonymous Ferret 12mth #1044bd

I think this student is referring to the 3^3 in Q4.3. The solution states that there are 7 bits for the tag and 2 bits for the metadata, which amounts to 3^2 bits. However, the solutions says that it is 3^3, which I also disagree with. Could you please confirm?

Jero Wang STAFF 12mth #1044cc

Replying to Anonymous Ferret

Yeah, that should be 3^2, noted on our end too.

Anonymous Dragonfly 12mth #1044aca

Replying to Jero Wang

So would the solution be 2 (3^2 + 2)

Anonymous Red deer 12mth #1044aae

why are there 3^2 trytes in each cache block?

Anonymous Monkey 12mth #1044abca

Shouldn't the tag be factored out of 3^3 trits? Since it is already included in 2^3

♡ ···

Anonymous Caterpillar 12mth #1044bef

where did the 2^3 come from in 2^2 times 2^3 = 2^5?

♡ 2 ···

Anonymous Cassowary 12mth #1044bfa

1 byte has 8 bits

♡ 1 ···

Anonymous Cassowary 12mth #1044aa ✓ Resolved

SU-23 Final Q2.2

Why do we do sb s7 0(s1), why cant we store directly to a1 instead, like doing something like mv a1 s7

♡ ···

N Nikhil Kandkur STAFF 12mth #1044af

We can't directly do mv a1 s7 since a1 is a memory address and we actually have to save our result to the address, not the register that hold the address. In addition, given the fact that we are doing jal several times and NOT saving a1, we cannot assume that a1 will continue to hold the memory address, so that's why we moved it into s1 in the beginning of the label.

♡ ···

Anonymous Ferret 12mth #1044e ✓ Resolved

SU23-Final-Q5

Is Q5.10 through Q5.13 out of scope for this semester?

♡ ···

J Jero Wang STAFF 12mth #1044ba

Yes

♡ ···

Anonymous Wombat 12mth #1044d ✓ Resolved

SP-23-Final-Q2.1,2.2

Why is Immediate generator included in stage 2 (EX/MEM/WB). I thought it was part of stage 1 since I thought it was part of stage ID.
P

♡ 1 ···

J Jero Wang STAFF 12mth #1044bb

We updated the datapath on the refcard to move ImmGen to ID this semester, but in SP23, it was in EX.

♡ 3 ···

Anonymous Wombat 12mth #1044bc

So in the new datapath, would the answer be 705ps for 2.1 and A&B Mux for 2.2?

♡ 1 ···

S

Sanjit Rao 12mth #1044db

↩ Replying to Anonymous Wombat

That is what I got

♡ 1 ···

J

Jero Wang STAFF 12mth #1044afd

↩ Replying to Anonymous Wombat

Yes for 705ps, but AMux and BMux are two separate component, so it doesn't work for Q2.2. The answer would be "Not Possible".

♡ 3 ···

Anonymous Grasshopper 12mth #1044bfd

↩ Replying to Jero Wang

And for Q2.3, the answer is also included "Immediate", right?

♡ ···

Anonymous Cobra 12mth #1044aada

↩ Replying to Jero Wang

[SP-23-Final-Q2.2] Could you elaborate on why this wouldn't be possible?

♡ ···

J

Jero Wang STAFF 12mth #1044affd

↩ Replying to Anonymous Cobra

There's no single component you can move that reduces the critical path. The only way (I think) is to move both AMux and BMux, but that's not a single component.

♡ ···

Anonymous Goldfinch 12mth #1044adfe

↩ Replying to Jero Wang

Why couldn't we save the value that should come out of the mux in a register, so that we can directly feed into the ALU to get the result?

♡ ···

J

Jero Wang STAFF 12mth #1044affe

↩ Replying to Anonymous Goldfinch

You can, but that wouldn't affect anything unless you move both AMux and BMux, which doesn't qualify as moving one component.

♡ ···

N

Nidhi Nayak 12mth #1044afed

↩ Replying to Jero Wang

What path gets you 705? I'm getting less than that so I am a little confused

♡ ···

Malavikha Sudarshan 12mth #1044afef

↩ Replying to Nidhi Nayak

t_clk_to_q + AMux + BMux + ALU + Memory Read + t_setup

♡ ···

N

Nidhi Nayak 12mth #1044affc

↩ Replying to Malavikha Sudarshan

Why is it both a mix and b mux? Wouldn't it only be the max of the two?

♡ ···

Jero Wang STAFF 12mth #1044baaa

Replying to Nidhi Nayak

Yes, but I think the other list is missing WBMux

♡ ···

Anonymous Dinosaur 12mth #1044baae

Replying to Nidhi Nayak

wait sorry i think i missed the understanding of muxes too! this thread helped https://edstem.org/us/courses/43491/discussion/3970125?comment=9274775

♡ ···

Jero Wang STAFF 12mth #1044afff

Replying to Nidhi Nayak

Everything listed in the solutions except ImmGen.

♡ ···

Anonymous Llama 12mth #1044abdd

[SP-23-Final-Q2.1] Why do we not include the Branch Comparator Delay into stage2 calculation? Could you please write out the equation to get the correct answer based on this sem's Datapath. I would also appreciate if you list the components for each stage.

♡ 2 ···

Anonymous Woodpecker 12mth #1044abfa

Replying to Anonymous Llama

I think because the branch comparator isn't used in a load instruction. You can compute how long a branch instruction takes but it ends up being shorter than a load instruction.

♡ 1 ···

Jero Wang STAFF 12mth #1044baab

Replying to Anonymous Llama

It's the same equation, just no ImmGen. Branch comp happens in parallel to ALU/AMux/BMux.

Also, as Anon Woodpecker mentioned, it's not used.

♡ ···

Anonymous Sand Dollar 12mth #1044c ✓ Resolved

Su23-Final-Q3.7

Why is MemRW 'read' here? I thought it wouldn't matter since WBSel doesn't actually select that branch. Thanks!

♡ ···

Nikhil Kandkur STAFF 12mth #1044f

You are right that WBSel does not select the memory input into the WB mux, BUT, if we are not working with DMEM, we should have a default read operation, because setting MemRW to write could write to DMEM, which we do NOT want.

♡ ···

Anonymous Sand Dollar 12mth #1044cf

Got it, thank you Nikhil!

Anonymous Goose 12mth #1044bcc

Why is WBSel PC+4?

N Nikhil Kandkur STAFF 12mth #1044bce

Replying to Anonymous Goose

WBSel is PC + 4 because we are doing a jump AND link, so in addition to changing our PC, we also need to make sure that we are storing our return address (which is PC + 4) to the specified register.

♡ 1

Anonymous Goose 12mth #1044cbe

Replying to Nikhil Kandkur

Ohh okay, that makes more sense thank you!

Anonymous Dove 12mth #1044abce

Why do you need to read MemRW and what is the purpose of the default read option. i am not seeing how you arrived at this answer

Anonymous Oryx 12mth #1044a ✓ Resolved

SU23-Final-Q8.1

How can we represent 0 to 63 with just 4 bits? Thank you!

♡ 1

N Nikhil Kandkur STAFF 12mth #1044b

The answer says that those bits would be the exponent bits, so given standard bias, we would need 4 exponent bits to represent 63 in our IEEE-754 FP standard.

Jiries Kaileh 12mth #1044adab

Is there a walkthrough video for this problem?

♡ 1