! This class has been made inactive. No posts will be allowed until an instructor reactivates the class.

note @366 291 views

[Exams] Past Exams 2018 Q&A

Discuss all questions pertaining to exams which took place in 2018 here.

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

When posting questions, you **MUST** reference the semester, exam, **AND** question so we can help you.
Please put this at the beginning of your post in this format: **[{Semester}-{Exam}]:Q{Question Number}**
For example: **[SP-MT1]:Q1**, or **[SU-MT2]:Q3**

{Semester} is one of these: SP, SU, FA
{Exam} is of of these: Q, MT, MT1, MT2, F

If you follow this format, it will make it very easy to search for similar questions!

midterm_exam1 midterm_exam2 final_exam

~ An instructor (Jerry Xu) thinks this is a good note ~

Updated 1 month ago by Stephan Kaminsky

followup discussions *for lingering questions and comments* 1 endorsed followup comment

Resolved Unresolved

Anonymous Poet 4 months ago
[SP-MT1]:Q3b

What would print statement #2 print out? I thought it would be normal...

helpful! | 0

Anonymous Helix 4 months ago char artist[100] = "artist" creates the string on the stack. song->artist points artist to that string, but when the function returns, the stack pointer increments and its undefined behavior to have a pointer pointing to a random memory location that's not part of the stack.

helpful! | 1

Resolved Unresolved

Anonymous Calc 4 months ago
[FA-MT1]:Q3

(reposting from above to make sure this question is marked correctly.

I had a question regarding Midterm 1 Spring 2018 Question 2 Part B. My question is specific regarding the Base case when a Leaf node is encountered.

```
list node *curr = malloc(sizeof(list node));
curr->str = malloc(strlen(root->str)+1));
```

Instead of the two lines above, couldn't we have written the following? If not, why? Is it because the string wouldn't be considered copied?

```
curr-str = root-str;
```

```
if (root->str != NULL) {
    list_node *curr = malloc(sizeof(list_node));
    curr->str = malloc(strlen(root->str)+1));
    strcpy(curr->str, root->str);
    return curr;
}
```

helpful! | 0

Anonymous Calc 4 months ago **[SP-MT1]:Q3**

I wish we could correct out posts on piazza...

helpful! | 0

Anonymous Calc 4 months ago **[SP-MT1]:Q2**

helpful! | 0

Jie Qiu 4 months ago Exactly. We do not want the copy's str to change when we change the original copy, and vice verse.

helpful! | 0

Resolved Unresolved

Anonymous Calc 4 months ago
[SP-MT1]:Q3

(I know I marked this one right)

I had a question regarding Midterm 1 Spring 2018 Question 3. The adress of **song2.artist** is on the **static** space, yes?

helpful! | 0

Jie Qiu 4 months ago Yes

helpful! | 0

Resolved Unresolved

Anonymous Calc 4 months ago

[SP-MT1]:Q4

I had a question regarding Midterm 1 Spring 2018 Question 4 part d.

Is sizeof() implicitly called during pointer arithmetic? Also, if 16 bits is 2 bytes, why does this makes sense that str + outer is +2 the address? When we think of addresses, what is their relationship with bytes of variables? Does the size of the address mean anything in terms of how much memory takes space for a variable?

Suppose that we revise the above insertion sort algorithm to support a new character encoding that uses 16 bits (similar to Unicode). These characters are stored in a 16 bit unsigned integer type (called `uni_t`). The function is identical except that all variables are now of type `uni_t` instead of `char`. We will call this function `uni_string_insertion_sort`. We call this new sort in a similar fashion:

```
uni_t str[4] = "Ωβα";
uni_string_insertion_sort(str);
```

Suppose that the first character in `str` is stored at address `0xFFF8`. Answer the following questions assuming `uni_string_insertion_sort` is run from the start. If the answer cannot be determined or would cause an error from the provided information, write "Unknown".

(d) After line 4 has executed, what is the value of `str + outer`?

Solution: 0xFFFA (0xFFF8 + `sizeof(uni_t)*1`)

helpful! | 0

Anonymous Poet 2 4 months ago Pointer arithmetic does an implicit sizeof before adding onto the pointer. The size of a single uni_t is 2 bytes, so it makes sense that the difference between str and str+outer is 2 (as addresses are measured in bytes, not bits). The size of the address itself depends on how many bits the computer is running at, but the memory it points to "takes up" the amount of space needed to take up the size of that variable. So if str was a char*, str+1 would only be 1 byte away from str and 4 bytes if str was an int*.

I like to think of pointer arithmetic as indexing into the array following the pointer. The memory given to the pointer must be large enough such that what it's pointing to can fit into the slot it's pointing to, and adding x onto it will return an address x*sizeof bytes down the array.

helpful! | 1

Resolved Unresolved

Anonymous Calc 4 months ago

[FA-MT]:Q1

I had a question regarding Midterm Fall 2018 (listed as Midtem 2 in some places there was a quest during that semester) for Question 1 part d).

I am having a hard time understanding what the solution is saying and how the answer, 8.0, was arrived. Can someone please re-word the explanation?

Why did it say, "There are 2 mantissa bits, so there are always 4 numbers in the range" and how was the range determined?

I thought the answer was 8,000,000,000 because initally i thought the minifloat max number was ($(1 + 2^{-1} + 2^{-2}) * 2^{23}$ for 0b01111011 or 0xFB

d) What does **should_be_a_billion()** return? *(assume that we always round down to 0)* ___8.0____

This is basically the value when you start counting by 2s, since once you start counting by 2s and always round down, your sum doesn't increase. There are 2 mantissa bits, so there are always 4 numbers in the range [2^i, 2^{i+1}). So you ask yourself, what gap has exactly 4 numbers between consecutive values of [2^i, 2^{i+1}), meaning when are you counting by 1? Easy, [4-8) ⇒ {4, 5, 6, 7}. When you get to 8 you're counting by 2s since you have to cover 8 to 16 with only 4 numbers: {8, 10, 12, 14}. **So it's 8.0** and you didn't have to do any work trying to encode numbers in and out of minifloats, since that was what question c was supposed to be about.

```
minifloat should_be_a_billion() {
        minifloat sum = 0.0;
        for (unsigned int i = 0; i < 1000000000; i++) { sum = sum + 1.0; }
        return(sum);
}
```

helpful! | 0

Daniel Zhang 4 months ago The gist of the question is, once we reach 8.0, we no longer have enough significand bits to represent 8 + 2 = 10 (remember– you only have two significand bits). Thus, we always round 10 down to 8, trapping us in a loop where we are stuck at 8 for all eternity

helpful! | 0

Anonymous Calc 4 months ago in the question, it says there are 3 mantissa bits though?

helpful! | 0

Anonymous Calc 4 months ago even though the blank is different from the answer key i understand now!

helpful! | 0

Resolved Unresolved

Anonymous Calc 4 months ago

[SU-MT]:Q1

I had a question regarding Midterm 1 Summer 2018. If the exponent field is now in two's complement instead of in bias notation, would this mean we wouldn't be able to represent Nan anymore?

7) We now modify the floating point description in part 6 so that the exponent field is now in **two's complement** instead of in bias notation. Compute the floating point value of 0b1111 1100.

Exponent: 0b111 = -1, Signif: 0.75 = (-1) x 2^-1 x 1.75 = -0.875

helpful! | 0

Jie Qiu 4 months ago We would still be able to. NaNs have exponent bits all 1's and significand non-zero. It's not the decimal value of the exp field that matters, but the binary representation.

helpful! | 0

Resolved Unresolved

Anonymous Calc 4 months ago
[SU-MT1]:Q2

I had a question regarding Question 2 on the Summer 2018 Midterm 1 for part 9.

The Answer for part 9 was D but wouldn't garbage be printed since the function returned a pointer to an array (**res**) ? Therefore, shouldn't the answer be B?

In cases when the array is on the stack and there is a pointer to it from outside of the function, once the function is done, the data on the stack is "gone." So whatever pointer is pointing to data that was previously on that stack would be garbage right?

helpful! | 0

Nikhil Pimpalkhare 4 months ago Since res is a local variable, after function "a" returns the return value now points below the stack pointer. Then, printf will dereference below the stack pointer, which is an illegal operation.

helpful! | 0

Resolved Unresolved

Anonymous Calc 4 months ago
[SU-MT1]:Q4

I had a question regarding Question 4 on the Summer 2018 Midterm 1 for the remove_ptr function.

While looking at the solution, when the node in the chain is removed, how is there NOT any memory leaks from linking the previous node back to the node that next to the node that points to the marked meta data?

There is no code segment linking the next node that was not supposed to be free back to the previous node.

```
bool remove_ptr (m_node **node_addr, void *ptr) {
     if (node_addr && *node_addr && ptr) {
          if ((*node_addr)->data_ptr != ptr) {
               return remove_ptr (&(*node_addr)->next, ptr);
          }
          m_node *temp = (*node_addr);
          free (temp->data_ptr);
          *node_addr = temp->next;
          free (temp);
          return true;
     }
     return false;
}
```

helpful! | 0

Anonymous Calc 4 months ago nvm this is hard to explain why it works

helpful! | 0

Resolved Unresolved

Anonymous Atom 4 months ago

[SP-MT1]:Q3

Why is song1 on the heap? I thought *song1 would be in the heap since that is the memory allocated. I thought song1 was just a locally defined pointer, so shouldn't it be in the stack?

```
Song * createSong() {
     Song* song = (Song*) malloc(sizeof(Song));
     song->title = "this old dog";
     char artist[100] = "mac demarco";
     song->artist = artist;
     return song;
}


int main(int argc, char **argv) {
     Song *song1 = createSong();
     printf("%s\n", "Song written:");
     printf("%s\n", song1->title); // print statement #1
     printf("%s\n", song1->artist); // print statement #2

     Song song2;
     song2.title = malloc(sizeof(char)*100);
     strcpy(song2.title, song1->title);
     song2.artist = "MAC DEMARCO";
     printf("%s\n", "Song written:");
     printf("%s\n", song2.title); // print statement #3
     printf("%s\n", song2.artist); // print statement #4

     return 0;
}
```

helpful! | 1

Anonymous Calc 4 months ago I had the same thought as well, I think for this question over it was trying to get us to answer where the value in the variable is located.

"address does each **value** evaluate"

For exams, we should ask for clarification if something like this arises

helpful! | 0

Anonymous Comp 2 4 months ago But, still, the address of song is the address of a pointer, which lies in stack... May I please ask if ambiguity has risen for this question in the past?

helpful! | 0

Resolved Unresolved

Anonymous Comp 4 months ago

[SP-MT1]:Q2b

In the second if-statement below, shouldn't we also be setting curr->next to NULL as well since malloc isn't guaranteed to return empty memory? curr->next needs to be null for find_end to work.

```
Solution: list_node * flatten(tree_node *root) {
     if (root == NULL)
          return NULL;
     if (root->str != NULL) {
          list_node *curr = malloc(sizeof(list_node));
          curr->str = malloc(strlen(root->str)+1));
          strcpy(curr->str, root->str);
          return curr;
     }
     list_node *left_list = flatten(root->left);
     list_node *right_list = flatten(root->right);
     list_node *rend = find_end(right_list);
     if (rend == NULL) {
          return left_list;
     } else {
          rend->next = left_list;
          return right_list;
     }
}
```

helpful! | 0

Jie Qiu 4 months ago hmm I agree.

helpful! | 0

[REDACTED] 4 months ago don't you need to add

```
curr->next = NULL;
```

for findend() to work properly? otherwise your pointer->next points to garbage values?

helpful! | 0

Daniel Zhang 1 month ago Yeah– I'm guessing that whoever wrote the original solution to this problem meant to use calloc instead of malloc.

helpful! | 0

Resolved Unresolved

Anonymous Helix 4 months ago

[SP-MT1]:Q6b

Im having trouble tracing the risk v code. My confusion stems from the fact that the problem says a0 holds the address of input. Input is a pointer so input itself also holds an address, but a0 only holds the address of the pointer correct? But when the code does addi a0, a0, 4, isn't a0 now just holding the value of some random memory? Its not moving across the memory in the heap since thats what input contains. Now when we load word from a0 I have no idea what a0 contains.

When they do lw t1, 0(a0) isn't t1 just equal the address of the malloced location in the heap input points to?

Problem 6 ***RISC-V to C Magic*** **(15 points)**

Assume we have two arrays `input` and `result`. They are initialized as follows:

```
int *input = malloc(8*sizeof(int));
int *result = calloc(8, sizeof(int));
for (int i = 0; i < 8; i++) {
    input[i] = i;
}
```

You are given the following RISC-V code. Assume register `a0` holds the address of `input` and register `a2` holds the address of `result` when `MAGIC` is called by `main`.

```
main:
    ...
    # Start Calling MAGIC
    addi a1, x0, 8
    jal ra, MAGIC      # a0 holds input, a2 holds result
    # Checkpoint:  finished calling MAGIC
    ...
exit:
    addi a0, x0, 10
    add a1, x0, x0
    ecall      # Terminate ecall
MAGIC:
    # TODO: prologue.  What registers need to be stored onto the stack?
    mv s0, x0
    mv t0, x0
loop:
    beq t0, a1, done
    lw t1, 0(a0)
    add s0, s0, t1
    slli t2, t0, 2
    add t2, t2, a2
    sw s0, 0(t2)
    addi t0, t0, 1
    addi a0, a0, 4
    jal x0, loop
done:
    mv a0, s0
    # TODO: epilogue.  What registers need to be restored?
    jr ra
```

helpful! | 0

Jie Qiu 4 months ago At first, your a0 holds the address of the first element of the array (a0 = &input[0]). addi a0, a0, 4 increments a0 by 4 bytes, which means now a0 is the address of the second element (a0 = &input[1]). Every time you increment a0 by 4 you are moving it to point to the next element in the array. Note that the input array is contiguous on the heap memory.

helpful! | 0

Anonymous Helix 4 months ago Wait, but doesn't it say that a0 holds the address of input and not the value of input? input itself is just a pointer, so it holds the address x of the first element of the array, but input is also stored somewhere(like the stack) at memory location y and doesn't a0 hold y not x?

If not then what would the instructions have said if they meant that a0 holds the actual address of the pointer input?
helpful! | 0

Jie Qiu 4 months ago yes a0 holds the address of input. when you call lw t1 0(a0) you are loading the word at the **address** 0 + a0 and storing that to t1.
helpful! | 0

Resolved Unresolved

Anonymous Gear 4 months ago
[SP-MT1]:Q4c

(c) Is this program correct for all null-terminated strings?

○ Yes ● No

If you answered yes, leave this blank. If you answered no, provide a string that would serve as a counterexample.

Solution: '\0', as well as any statically defined string.

I understand it doesn't work for '\0', but why doesn't it work for a statically defined string?
(does that mean strings defined in stack or heap work?)

helpful! | 0

Jie Qiu 4 months ago You can't modify a statically-defined string.
helpful! | 1

Anonymous Mouse 4 months ago Also for {'\0'}, outer = 1 at line 4 meaning str[outer] indexes out of bounds.
helpful! | 1

Resolved Unresolved

Sarah Bhaskaran 4 months ago
[SP-F]:Q1bii
To get 128, are they counting positive zero and negative zero as two different numbers? I feel like for this question +0 and -0 would be equivalent since it's not floating point. Then to calculate the number of representable values I added 1 (for zero) + 63 * 2 (for each of the nonzero numbers representable by the last

three bits, times two because they can be positive or negative. I got 127.

(ii) Suppose rather than using a bias notation, we decide to do the following.

For each base 4 number, we will reserve the most significant digit to strictly be used as a sign bit. A digit value of 1 will indicate a negative number, and a digit value of 0 will indicate a positive number. Any other values will result in an invalid number. For instance:

$$0003_4 = +3 \qquad 1003_4 = -3 \qquad 2003_4 = Invalid$$

How many valid representation can we represent with a 4 digit base 4 number using this scheme?

Solution: 2*4*4*4 = 128

helpful! | 0

Daniel Zhang 4 months ago Notice the question is asking for number of "valid representation"s, not real numbers. Because 0 has two valid representations (-0 and +0), it is counted twice.

helpful! | 0

Resolved Unresolved

Anonymous Calc 4 months ago

[SU-F]:Q1

I had a question regarding Final Exam from Summer 2018 for Question 1. I had a hard time "seeing" how the scenario of the fork thread came to be. In other words, I didn't understand when (*sp->in_use >0) was NOT true and the function ends up in the for-loop process for making the fork. (I highlighted parts of the code indicated in green).

The exam gave us a diagram of how the push function is implemented. However, when a fork occurs, it shows that in_use for the block sp points to is still 1. So how is it possible for the code to run the part indicated by /* We are making a fork */ if it doesn't by-pass the if-statement regarding (*sp->in_use >0). Because based on the picture, (*sp->in_use >0) is always true

```
void push (char *func_name, StackNode **sp) {
      StackNode *temp = (StackNode *) malloc(sizeof(StackNode));
      temp->frame = malloc( frame_size(func_name) );
      temp->prev = *sp;
      temp->func_name = func_name;
      temp->threads = 0;
      temp->in_use = 1;
      if (*sp == NULL) {
            *sp = temp;
            return;
      }
      if ((*sp)->in_use > 0) {
            (*sp)->threads += 1;
            (*sp)->in_use -= 1;
      } else {    /* We are making a fork. */
            for (StackNode *trace = *sp; trace != NULL; trace = trace->prev) {
                  trace->threads += 1;
            }
      }
      *sp = temp;
}
```

helpful! | 0

Nikhil Pimpalkhare 4 months ago This question is out of scope for this midterm. The reason you can enter the else block is that the process of "forking" creates a second "thread", which you'll learn is an independent sequence of instructions that are being evaluated, so the first would decrement the "in_use" value of your frame to 0.

Actually don't worry about it though, this is content that will be covered during Week 13.

helpful! | 0

Resolved Unresolved

Anonymous Scale 4 months ago

[SP-MT1]:Q1a

Why is 129 N/A? I thought if a number is too big its wraps around and goes to negative numbers. Ex. 129 would go to -127. This is what we did in HW1 so I am very confused.

(a) Translate the following decimal numbers into 8-bit two's complement and unsigned binary representation in the table below. If a translation is not possible, please write "N/A". **Write your final answer in hexadecimal format.**

Solution:

Decimal Number	Two's Complement	Unsigned Number
10	0x0A	0x0A
129	None	0x81
-12	0xF4	None

helpful! | 0

Jie Qiu 4 months ago In hw, you were essentially doing computations in binary representation, and the overflowed decimal value was never represented in binary.

helpful! | 0

Resolved Unresolved

Anonymous Scale 4 months ago
[SP-MT1]:Q6c

For converting the RISC-V to C code, how can you tell that the return value is the sum of the input array? Where in the code can you see that? What are general strategies to solve problems like these?

helpful! | 0

Jie Qiu 4 months ago The return value is always stored in **a0**. Right before we called `jr ra` to return from this function, we run `mv a0, s0`, which basically just means we are returning the value in **s0**. **s0** is the sum of the input array because it was initialized in the beginning as 0, and as we looped through the input array, we added every element in the input array to **s0** by `add s0, s0, t1.`
Generally, when I try to solve problems like these, I would first try to understand the riscv code, especially the functionality of each register, then everything else should easily follow.

helpful! | 1

Resolved Unresolved

Anonymous Calc 2 4 months ago
[SP-MT1]:Q5

How does dividing up points for this to calculate our score even work?

There is an a, b, c, d1-d4, and 15 points total? I have no idea how this would be divided.

helpful! | 0

Lewis Zhang 4 months ago not in scope anyways.

helpful! | 0

Lewis Zhang 4 months ago nvm it might be in scope, remarking as unresolved

helpful! | 0

Daniel Li 4 months ago Q5 is not in scope

helpful! | 0

Resolved Unresolved

Anonymous Atom 2 4 months ago
[SP-MT1]:Q5
Is Question 5 on Spring 2018 MT 1 in scope? It talks about converting RISC-V to machine code as well as opcodes.

helpful! | 0

Anonymous Calc 4 months ago 5 and 6 are not in scope

helpful! | 0

Anonymous Calc 4 months ago Nothing on Instructions

helpful! | 0

Resolved Unresolved

Anonymous Helix 2 4 months ago
[SU-MT1]:Q2 (2018)

For the second part, why would comment 6 be possibly legal and comment 8 be always legal?
I thought the possible error in both would be that if there was not enough memory left for the malloc and calloc calls, then copy_message and print_msg would not work correctly?

helpful! | 0

Jie Qiu 4 months ago Here we assumed malloc and calloc always succeed.
6 is possibly legal because you malloc'd space. When the line `x[strlen (x)] = '/0'` is executed, `strlen` will go through your memory space character by character until it hits a null terminator. Since we didn't copy over the null terminator from `msg`, we don't know if we are actually going to hit a null terminator.
8 is always legal because you calloc'd space for `x`, so even though we didn't copy over the null terminator, it is already there. Thus, printing it out would always be legal.
helpful! | 1

Resolved Unresolved

[REDACTED] 4 months ago
[SP-MT1]:Q6

Is loop going to be run automatically aftere MAGIC is run or there has to be some "j loop" or "jal ra loop" to run loop. But I don't see any jump to loop in the code.
helpful! | 0

Jie Qiu 4 months ago It's run automatically. Instructions are executed in sequential order.
helpful! | 0

Resolved Unresolved

Anonymous Gear 2 4 months ago
Is Q5 in the scope of MT1? If so, how do we approach part a?
helpful! | 0

Daniel Li 4 months ago This question is not in scope. Please format your questions as given by the example in the post
helpful! | 0

Resolved Unresolved

Anonymous Mouse 2 4 months ago
[SP-MT1]:Q6

In which line of the RISC-V code is result written to since I don't see that it's referred to anywhere besides "add t2, t2, a2"

helpful! | 0

Daniel Li 4 months ago The sw s0, 0(t2) instruction is where we actually store values into result. Whenever we add t2 to a2, we are essentially finding the index in our array through pointer arithmetic and then storing the value in s0 into that address in memory (i.e an index in our array)

helpful! | 0

Resolved Unresolved

Anonymous Calc 4 months ago

[SU-MT1]:Q3

I had a question regarding Summer 2018 Midterm 1 Question 3 Part 2. This is about the first part of the C-code and RISC-V.

This is about initalizing a pointer to a unsigned array on the stack. I get that the stack pointer needed to be decremented in order to save space for the array. But is t4 register "using" this space? Is address of t4 register is where the new stack pointer is but the array was not "saved" so to speak.

Why was the stack pointer just decremented and nothing else for the array on the stack?

```
foo:
        slli  t6 a0 2
        sub   sp sp t6
        mv    t4 sp
        sw    zero 0(t4)
        addi  t1 zero 1
```

```
Translate the RISC-V Assembly on
the left into C code to complete
the function foo:

unsigned foo(unsigned n) {
  unsigned arr[n];
  unsigned total = 0;
```

helpful! | 0

Anonymous Calc 4 months ago Wait the stack grows downwards

helpful! | 0

Resolved Unresolved

[REDACTED] 4 months ago

[SP-MT1]:Q2b

Solution:

```
list_node * flatten(tree_node *root) {
    if (root == NULL)
        return NULL;
    if (root->str != NULL) {
        list_node *curr = malloc(sizeof(list_node));
        curr->str = malloc(strlen(root->str)+1));
        strcpy(curr->str, root->str);
        return curr;
    }
    list_node *left_list = flatten(root->left);
    list_node *right_list = flatten(root->right);
    list_node *rend = find_end(right_list);
    if (rend == NULL) {
        return left_list;
    } else {
        rend->next = left_list;
        return right_list;
    }
}
```

in the last line, shouldn't it return rend and not right_list? We haven't malloced any memory for rend so setting its next to left_list is of no consequence unless we return it?

helpful! | 0

Jie Qiu 4 months ago **rend** is the last node of the **right_list**. By setting `rend->next = left_list`, you are changing the right list by linking it with the left list.

helpful! | 1

[REDACTED] 4 months ago Wouldn't we lose the root node then? Since rend's next is immediately the left branch, not the root.

Also, why do we return right_list and not rend?

helpful! | 0

Daniel Zhang 1 month ago You won't lose root because root is the value that's being passed in and is never edited in this function. Right_list is returned because we want to return a pointer to the start of the linked list.

helpful! | 0

Resolved Unresolved

Anonymous Scale 4 months ago
[SP-MT1]:Q3ai

Why is song1 in the heap? Since it is a ptr shouldn't it be in the stack (*song1 would be in the heap however?)

helpful! | 0

Evan Sum 4 months ago Hello! So you are somewhat correct when you say that the ptr is LOCATED on the stack and that if we did *song1 we would then be LOCATED on the heap.

However, since it's asking for the VALUE that song1 has, you can think of it like this jank image below.

memory

|_____|
|_____|
|_ptr to some heap address_| this address is in the stack and is song1
|_____|
|_____|

In this example, &song1 would be in the stack. But song1 contains the VALUE of something in the heap. It asks for what VALUE each variable would evaluate to, and a ptr contains the actual address of its destination. That means "ptr to some heap address" is actually 0xHEAP0000 address.
helpful! | 0

Resolved Unresolved

Anonymous Scale 4 months ago
[FA-MT1]:Q2a

How is this code returning 1 if N is a power of 2? For example if N = 00001 count would be incremented by 1 since N & 1 = 1. Then if we do N >> 1, N will only consist of 0's so from here on out so counter = 1?

```
int mystery (unsigned int N) {
        unsigned int counter = 0;
        while (N > 0) {
                counter += N & 1;
                N = N >> 1;
        }
        return counter == 1;
}
```

Q2a) What does the **mystery** return? (Select ONE)

- ○ The number of 1s in the binary representation of **N**
- ○ 1 if **N** is odd, otherwise 0
- ● **1 if N is a power of 2, otherwise 0 [it shifts N to the right, storing in counter all the 1s it sees. If it's exactly 1, meaning the only 1 is the MSB (most significant bit), then it's a power of 2]**
- ○ 1 if the binary representation of **N** is all 1s, otherwise 0
- ○ 1 if the binary representation of **N** has any 1s, otherwise 0

Q2b) Give
unsigned
GetNFrom
int myst
...could **my**
mystery?
○ Yes
○ It depe
● **No [b**

helpful! | 0

Jie Qiu 4 months ago 1 is the zeroth power of 2. The idea is that powers of 2 will only have 1 bit as 1 and the rest all zeros.

helpful! | 1

◉ Resolved ○ Unresolved

Anonymous Scale 2 1 month ago

[FA-MT2]:Q1f

Is there a difference to using srai vs srli since the RISC-V green sheet seems to show me that they do exactly the same thing

helpful! | 0

Daniel Li 1 month ago srai will sign extend but srli will not

helpful! | 0

◉ Resolved ○ Unresolved

Anonymous Beaker 2 1 month ago

[SU-F]:Q3

Suppose that the label `Q1` is at address 0x4000 0000. If the label `end` is at address 0x40**XY Z**800, what are all the possible values for **X**, **Y**, and **Z** such that `j end` can be resolved in the assembler? Formulate your answer in the form `[A - B]` where `A` and `B` are both hexadecimal digits.

X: 0
Y: [0 - F]
Z: [0 - F]

What is the reason X cannot be 1? My logic is that, j resolves to jal which is a UJ type instruction, so offset is 20 bits *extended by a 0* = 21 bits, therefore XYZ800 (24 bits) must have the top 3 bits zero, not top 4 bits?

helpful! | 0

Charles Hong 1 month ago

EDIT: My previous answer was incorrect, sorry! Here is the correct explanation:

Remember, we can jump backwards too! RISC-V immediates use the Two's Complement number representation. This affects the range of addresses we can jump to.

helpful! | 1

Resolved Unresolved

Anonymous Beaker 2 1 month ago
[SP-F]:Q10(d)

(d) Given a message of length `n` characters, how many instructions are needed after loop unrolling? Express your answer in terms of `n`, such as `3n + 4`. In addition, what is the speed up when `n` is approaching infinity in comparison to the **original non-optimized function** `obfuscate`? Count pseudo-instructions as 1 instruction. You do not need to simplify your expressions.

# of Instructions: $(7 + (n/8) * 10 + 1 + (n\%8) * 6 + 1)$ Speedup: 7.5X

So, asymptotically we have that optimized function takes $\frac{5}{4}n$ instructions and the original takes $6n$ instructions. Then, why is the speedup not $\frac{6}{\frac{5}{4}} = 4.8$ and is rather $6 \cdot \frac{5}{4} = 7.5$?

helpful! | 2

[REDACTED] 1 month ago I'm wondering the same.

helpful! | 0

Victor Sun 1 month ago Yes, you are correct this appears to be a mistake.

helpful! | 1

Resolved Unresolved

Anonymous Atom 2 1 month ago
[SP-F]:Q5(a)

Why is the maximum hold time 5 ps? I thought we are supposed to add the clk to q because hold time <= clk-to-q + shortest CL, should the maximum not be 5 + 3 = 8 ps?

helpful! | 0

Anonymous Beaker 2 1 month ago Inputs A and B are not registers, so they "take on their new values **exactly** at the rising edge of every clock cycle". Therefore the input to register 1 will be calculated, through the adder block (5ps), 5ps after the clock rising edge.

If inputs A and B were registers, you would be correct.

~ An instructor (Victor Sun) thinks this is a good comment ~

helpful! | 1

Resolved Unresolved

Anonymous Poet 3 1 month ago

[SP-F]:Q13A

"You have a computer that, well, stinks. It goes down on average 6 times a day and it takes 1 hour to get working again. What is the current system's availability?" Answer: 0.8

I'm fairly confident the answer here should be 0.75. The solution says the MTTF is 4 hr and thus availability is 4 / (4 + 1). I argue that failing an average of 6 times a day implies MTBF = 4hr => MTTF = 3hr => Availability = 3 / (3 + 1) = 0.75.

helpful! | 0

Victor Sun 1 month ago Perhaps this question was not worded the best... I think they were trying to get at 4 hour mttf and the main idea to get here is that availability = mttf/ (mttf+mttr)

helpful! | 0

Resolved Unresolved

Anonymous Atom 1 month ago

[SP-F]:Q6a

Why don't we need to stall after line 6? It's adding to t2, and the next line is storing that new t2 value to t0. Without stalling, wouldn't it store the old value of t2?

[SP-F]:Q6c

I don't understand why we need to flush the pipeline (5 stalls) with the read-write. Can someone help explain this?

helpful! | 0

Victor Sun 1 month ago 6a. You are correct you need stall after line 6 this is a mistake.

6c. 2 stalls after Line 1 b/c of data hazard with t0 in line 3, 3 stalls after line 3 b/c of data hazard with t0 in line 4. With write/read we could reduce number of stalls for each of these by 1 by writing and reading in same cycle. I don't think the answers included the stalls for the beq instruction since its on line 5

helpful! | 0

Anonymous Atom 1 month ago Does this mean we need a stall after line 7 as well? Because line 7 is only 2 instructions away, not enough time to get the updated value, even if its write-read right?

helpful! | 0

Charles Hong 1 month ago If we insert stall(s) after instruction 6, we are stalling every instruction after 6 by that amount, since the 5-stage pipeline we've been looking at is in-order, meaning instruction 8

must come after instruction 7. In the pipeline in the question, if instruction 7 is stalled enough to use the result of instruction 6, that must be the case for instruction 8 as well.

So, we don't have to add any additional stalls after instruction 7.

helpful! | 0

Anonymous Atom 3 1 month ago I do not understand why we need a stall on line 1? Why does slli and or on different registers require a stall?

helpful! | 0

Anonymous Helix 3 1 month ago It's actually the instruction on line 3 that has a dependency on the operation from line 1, I think.

helpful! | 0

Resolved Unresolved

Anonymous Calc 3 1 month ago

[SP-F]: Q7d(iii)

Why is the number of access for loop 1, five? why not 4?

(iii) Overall AMAT of Loop 1 2 (an expression of REDUCED fractions is alright. You may use "T1" as the Loop 1 AMAT and "T2" as the Loop 2 AMAT in your calculation of this value):

Solution: 20/21 * T1 + 1/21 * T2
The first loop has 5 accesses per step and $2^{13}/4$ total steps. The second loop has 2 accesses per step and $2^{13}/32$ total steps. This gets that the first loop does $5 * 2^{11}$ accesses while the second loop does $2 * 2^8$ or 2^9 accesses. The weighted average of AMAT then becomes: $T1 * (5 * 2^{11})/(5 * 2^{11} + 2^9) + T2 * (2^9)/(5 * 2^{11} + 2^9) = T1 * (5 * 2^2)/(5 * 2^2 + 1) + T2 * 1/(5 * 2^2 + 1) = T1 * 20/21 + T2 * 1/21.$

helpful! | 0

Anonymous Beaker 2 1 month ago When we do x += smth we do a read and a write for x. This is why the second loop has 2 accesses (1 read 1 write) and the first has 5 (4 read 1 write).

helpful! | 1

Resolved Unresolved

[REDACTED] 1 month ago

[SP-MT2]:Q4

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15	C16
ori s1 x0 0xf	F	D	E	M	W											
andi s2 x0 0		F	D	E	M	W										
beq s1 s2 exit			F	D	D	D	D	E	M	W						
lw s1 0xc(s0)				F	F	F	F	D	E	M	W					
xor s1 s1 s2								F	D	D	D	D	E	M	W	
lw s1 0xc(s0)									F	F	F	F	D	E	M	W

Why is it in the 5 stage pipeline the next instruction stays in that stage? Is that the same as a Stall?

Instructions	Cycles													
	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14
ori s1 x0 0xf	A	B												
andi s2 x0 0		A	B											
beq s1 s2 exit			A	*	B									
lw s1 0xc(s0)					A	B								
xor s1 s1 s2						A	*	B						
lw s1 0xc(s0)								A	B					

For the 2 stage pipeline we stall because we want the previous instruction to finish? Im also wondering if both pipelined CPUs are stalling?

helpful! | 0

Victor Sun 1 month ago Yes, essentially repeating a stage is one way of representing a stall, a * would be another way. Both CPUs contain stalls because of data ependencies.

helpful! | 0

Resolved Unresolved

Anonymous Scale 2 1 month ago

[SP-MT2]:Q7(f)

How did they get 3/8 for the miss rate within the L2 accesses?

helpful! | 0

Victor Sun 1 month ago I think this is for F not MT2 but 3/8 is hit rate not miss rate, then they do 1-hitrate to get miss rate. In this case hit rate is 3/8 because total array size is 2^15 and L2 cache size is 2^14 so it will store half the elements, but 1/8 element will be hit in L1 before it reaches l2 so 1/2-1/8= 3/8 elements hit in L2

helpful! | 0

Resolved Unresolved

Anonymous Atom 1 month ago
[SP-F]: Q5

Why do none of the hold time and clock period calculations add in the clock-to-q time? In the fall 2019 question about circuit timing (question 5), I believe clock-to-q time was considered when getting the answers. Is there a reason why it isn't for spring 2018?

helpful! | 0

Anonymous Atom 1 month ago I should clarify: clock-to-q is not used in parts a through c but it is used in d...

helpful! | 0

Charles Hong 1 month ago

There is no clock-to-q for those questions as the paths we are considering start at Inputs A, B, C which change exactly at the rising edge of the clock cycle. We don't know what they are physically, but they change exactly at the rising edge, so they behave differently from registers, which change their outputs a short time (which we call clock-to-q delay) after each rising edge.

Normally, when we consider clock-to-q time for something like a critical path delay calculation, we are considering the delay after the rising edge for which it takes the register at the start of the path to output a new value it read at the rising edge.

helpful! | 0

Anonymous Atom 1 month ago Ah so if it mentions that vlues change on the rising edge, then we should not take clock to q into consideration?

helpful! | 0

Anonymous Helix 3 1 month ago Correct.

helpful! | 0

Resolved Unresolved

Anonymous Calc 2 1 month ago
Can someone explain Fall 2018 M2c to me? I don't understand how that line of code gets you to multiply by 2, as opposed to shifting.

helpful! | 0

Resolved Unresolved

Anonymous Poet 3 1 month ago
[FA-F]: QF-1b

○**True** ○**False**	If we have a TLB which contains a number of entries equal to `MEMORY_SIZE / PAGE_SIZE`, every TLB miss will also be a page fault.

Solution says True.

But I thought of two reasons why it could be False.
a) What if the TLB and the OS have different eviction policies (for example TLB uses random, OS uses LRU for evicting pages from main memory)

b) What if switch to a different process then come back? The TLB will be cleared but the pages could still be in main memory?

helpful! | 0

Resolved Unresolved @366_f39

Anonymous Comp 3 1 month ago

[FA-F] QF-4f,g,h

I am lost with these 3 questions.

f) Why is the performance for T=3 equivalent to T=4?

g) How do we determine that TLB is the chokehold of this process?

h) Is there a resource I could look at to help me understand the mechanical process of solving h?

helpful! | 0

Anonymous Comp 3 1 month ago More specifically, for h, I am not able to get the Physical Page Numbers. For row 1, how does a VPN of 0x1 map to physical page number of 0x12?

helpful! | 0

Pavel Asparouhov 1 month ago I was stuck on this for a while, this helped out a lot. https://www.youtube.com/watch?v=8kBPRrHOTwg
Basically L1 is in memory from 0x00 to 0x0C. 0x1 has a first two bits of 00, so you got to L1 and grab the 0th entry, 0x20. This simply tells you the location of L2, which is from 0x20 to 0x2c. The second two bits of 0x1 are 01, so you grab the first entry from the specified L2 which in this case is 0x24 which holds 0x12

helpful! | 0

Resolved Unresolved

Anonymous Gear 3 1 month ago

[SP-F]: Q4D

How is the offset calculated? I tried using the relative address and subtracted it from 0x80 but that was way off.

helpful! | 0