Simulatoropgaven (A5, CompSys 2025)

David Marchant, Finn Schiermer Andersen

This assignment is due January 7th at 22:00. Note that this is a Wednesday rather than the regular Sunday deadline.

Task Description

This is a simple assignment, intended as some simple course revision before the exam. It only awards 2 points for exam qualification and does not count towards any of the particular topics (e.g. operating systems, networking, or computer architecture).

Your task is to answer the following questions as you would on an exam. Each of them should be answerable in at most a few paragraphs of text.

Any diagrams can be made on a computer, or drawn on a piece of paper and scanned in, just make sure that they are legible. You can also provide accompanying text explaining your diagram if necessary, though this is not required for a sufficiently clear diagram.

Task 1 - What even is a computer

Draw a diagram showing the hardware setup used of a typical modern computer. Your diagram should show components such as the CPU, registers, memory, disk, network adaptors, busses, as well as any other components you can fit in.

Task 2 - What even is memory

Consider the following table. It shows an area of the heap, with the values of the data at that location. Each word is only of size 2 bytes. In order to allocate blocks of data on this heap we use a block format with a header and footer, both of which are identical.

The values for the header and footer will tell us a about the block. The least significant bit (e.g. rightmost) will always have a value of 0 if the block is free, and 1 if it is allocated. The 2nd least significant bit will have a value of 0 if the previous block was free, and 1 if the previous block was allocated.

The remaining 14 bits are used to give the size of the block in bytes, including the header and footer. The minimum size of any block payload is 2 bytes.

Note that all values are given in hex.

For instance, starting at the bottom of the table at block `7ab8` we have the value `001B` which converts to the bits `0000 0000 0001 1011`. As bit 0 is 1 it is an allocated block, bit 1 is 0 so the previous block (off the bottom of the table) is allocated, and the remaing 14 show a size of 6 bytes.

Block Address	Value
7ada	00a7
7ad8	0072
7ad6	029b
7ad4	29a4
7ad2	ee2a
7ad0	5a3b
7ace	8ff3
7acc	1ca0
7aca	db31
7ac8	bb82
7ac6	009a
7ac4	03e8
7ac2	873a
7ac0	a524
7abe	0072
7abc	001B
7aba	2f14
7ab8	001B

Show the values stored in the heap after we run each of the following commands consecutively: - `ptrM = malloc(3)` - `ptrC = calloc(1)` - `free(ptrM)`

You should assume that the allocation system uses immediate coallescing, and that the minimum changes are made for each operation.

Task 3 - What even is the network

The protocol in A3 was a little basic, even if it didn't feel so at the time. It had no accounting for peers leaving the network. Propose some modifications to the protocol so that we can check if peers are still in the network. Note that we cannot assume that any peer leaving the network will always send a polite goodbye (though you may wish to include this when you can) as they might crash unexpectedly. Your modifications should be able to detect that peers have left the network within a certain amount of time, and inform the rest of the network that the peer has been lost.

Your answer should include any changes to the protocol made, including what messages types you are adding/changing/removing, what message contents is changing, and in what order messages or any other operations are performed.

Task 4 - What even is a program

There once was a 16-bit computer working on 16 bit numbers. It had 16 16-bit registers, r0 always zero. Memory was word-addressed and the word-size was 16 bit. Instructions came in 2 formats known as rrr and rri:

```
format:     size:       Layout:
rrr         16 bit      oooocccccbbbbaaaa
rri         32 bit      0000vvvvbbbbaaaa iiiiiiiiiiiiiiii
```

Here, ‘a’, ‘b’ and ‘c’ were used to select one of the 16 registers. ‘i’ was a 16 bit constant. Finally, ‘o’ and ‘v’ indicated the operation:

```
    hex oooo    meaning

    0   0000    reserved, indicates rri format.
    1   0001    a = b + c
    2   0010    a = b - c
    3   0011    a = b and c
    4   0100    a = b or c
    5   0101    a = b xor c
    6   0110    a = mem[b + c]      // This is a load, 'mem' means memory access
    7   0111    mem[b + c] = a
    8   1000    a = jalr (b + c)    // Ra = PC + 1, PC = Rb+Rc

    hex vvvv    meaning

    0   0000    reserved
    1   0001    a = b + i
    2   0010    reserved
    3   0011    a = b and i
    4   0100    a = b or i
    5   0101    a = b xor i
    6   0110    a = mem[b + i]
    7   0111    mem[b + i] = a
    8   1000    a = jalr (b + i)    // Ra = PC + 1, PC = Rb+i
    9   1001    if (a == b) PC = i
    a   1010    if (a != b) PC = i
    b   1011    if (a < b) PC = i
    c   1100    if (a <= b) PC = i
```

Assemble the following instruction sequence (to hex representation):

```
Entry:
    R2 = R1 + R0
Loop:
    R3 = mem[R2+R0]
    R2 = R2 + 1
    mem[R4+R0] = R3
    R4 = R4 + 1
    if (R3 != R0) goto Loop
    R1 = R2 - R1
    R0 = Jalr (R14 + R0)
```

Assume that the sequence starts at address `1234`.

Task 5

Write a function in ‘C’ corresponding to the instruction sequence given in task 4.

Submission format

Your should submit a pdf document containing your answers to the above tasks. Make sure to include an AI declaration just as you have for the other assignments. If you can fit everything in one pdf, then you only need to submit that document. If you have multiple files then they should all be submitted as a single compressed directory.