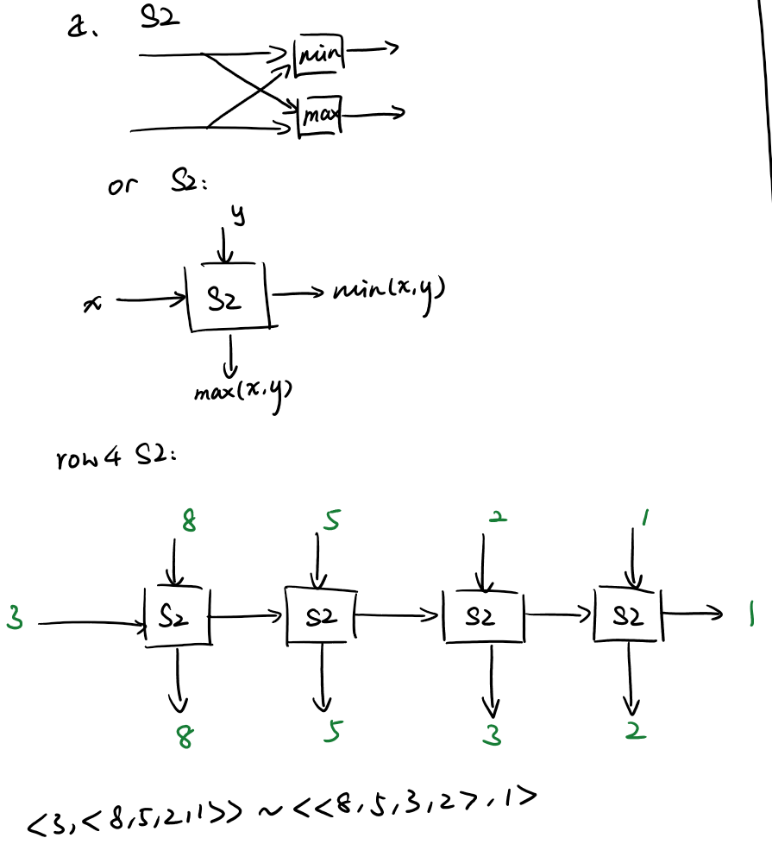
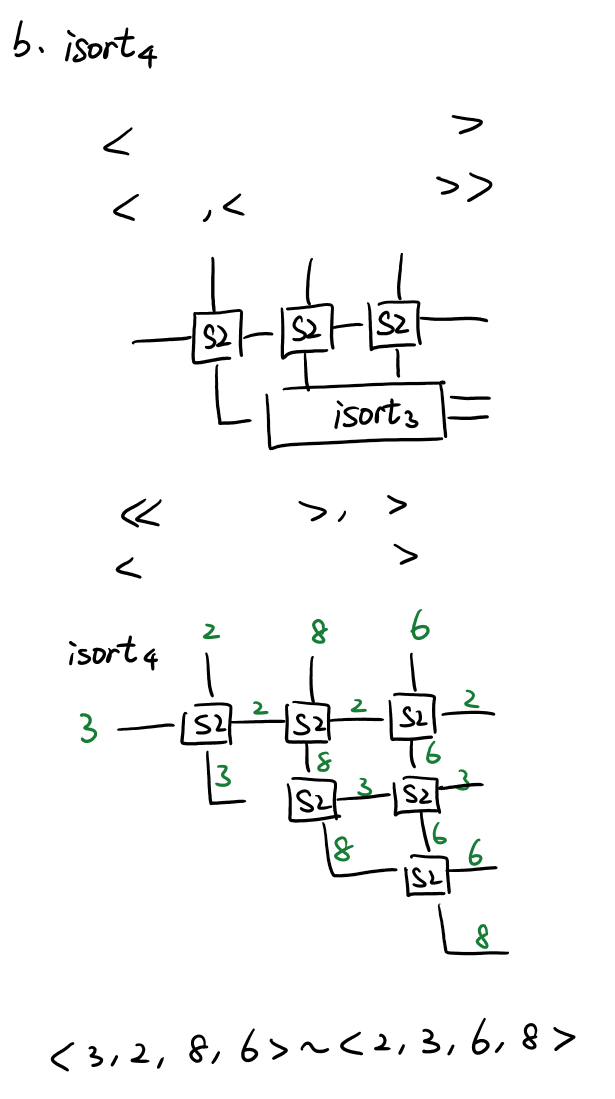
I have put some of my solutions here. I am not sure about the correctness of them. Please feel free to leave any comment and complete the solution. Thank! 😊

Question 1

a.

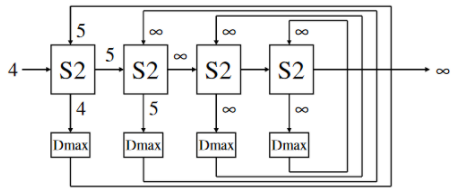


b.



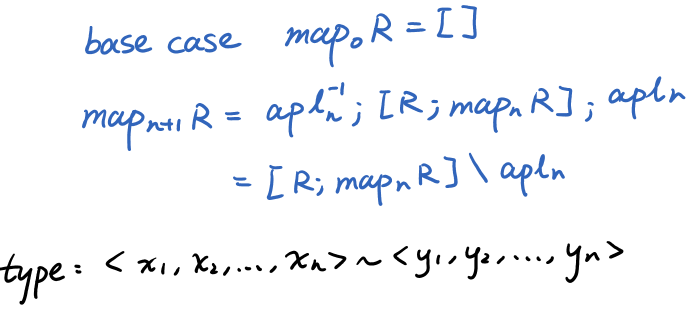
c.

You want to create this state machine – every cell remembers the largest value it has seen so far. As values stream in, the output will be the smallest value out of the *n* previous ones. When you fill all registers and want to get their values out, stream in n-times and that will push out everything that is in registers.

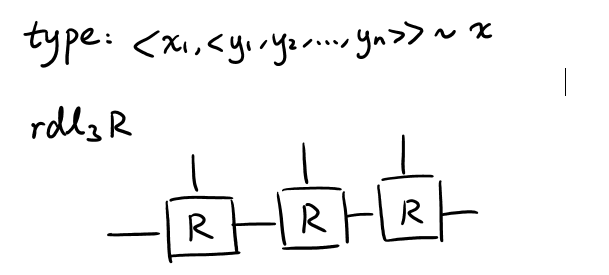


Question 2

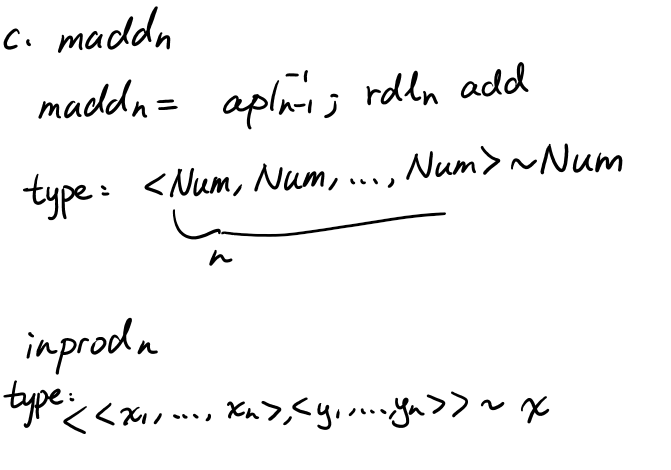
a.



b.



c.



For inprod n, I imagine something using zip and mac? Nah, don't bother. Thing below is good.

inprod n = zip n ; map n mult ; madd n.

d.

Grouped reduce left is relatively straightforward, just need to think of the types.

grdl m n R = snd (group m n); rdl m (rdl n R).

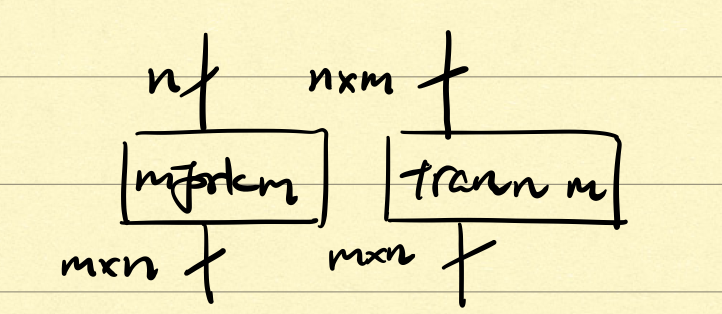
Partial pipelining means clustering. From above we have a thing with type <X, Y\_(m\*n)> ~ Z, now it’s just the matter of adding the delays as you would with outlines. It's a bit more complicated because now n stages need have the same number of Ds on the top. That can be solved with a triangle array of maps.

pgrdl m n = snd (group m n; /\ m (map n D)); rdl m ((rdl n R); D); D^~(m\*n + 1).

e.

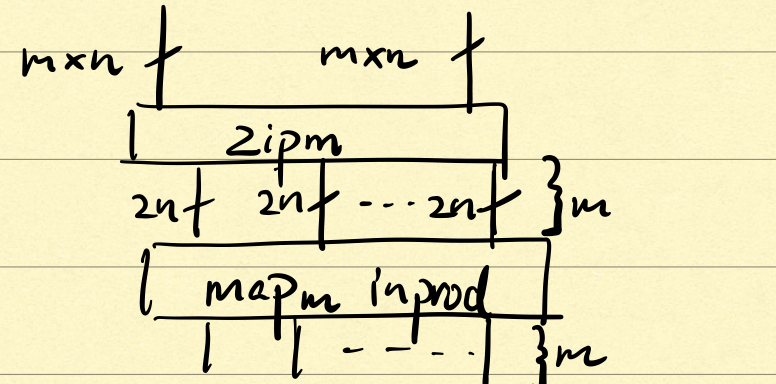
FC1 m n: <<Num>\_n, <<Num>\_m>\_n> ~ <<<Num>\_n>\_m, <<Num>\_n>\_m>

FC1 m n = [mfork m, tran n m].



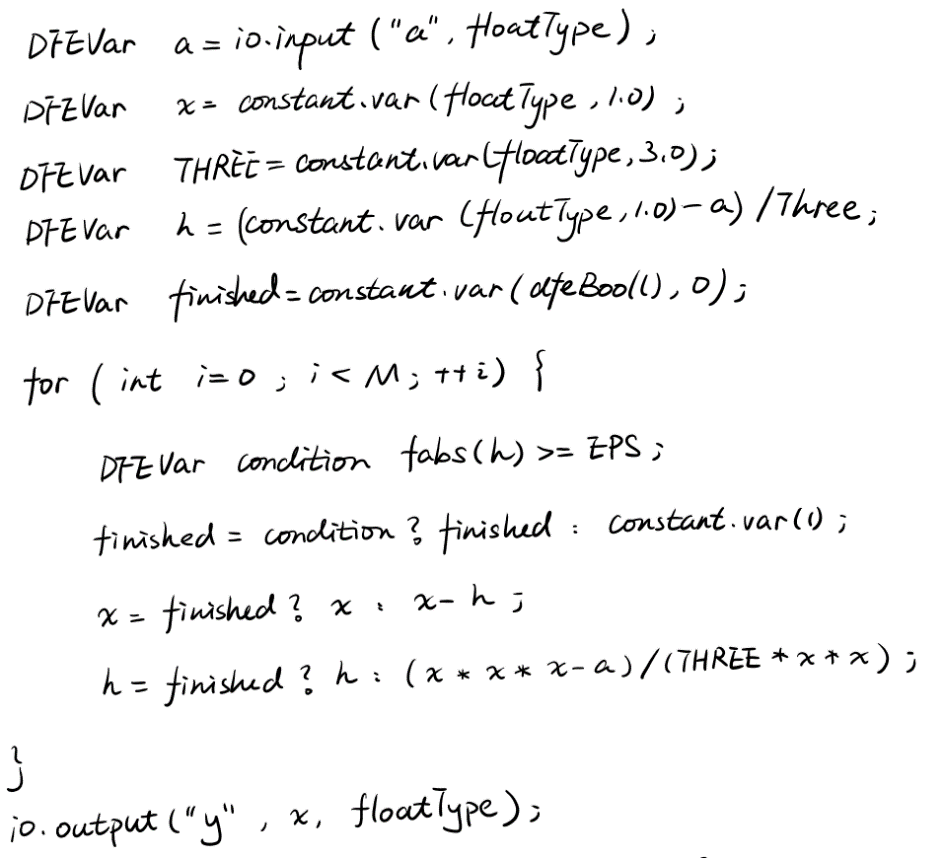
FC2 m n: <<<Num>\_n>\_m, <<Num>\_n>\_m> ~ <Num>\_m

FC2 m n = zip m; map m (inprod n).

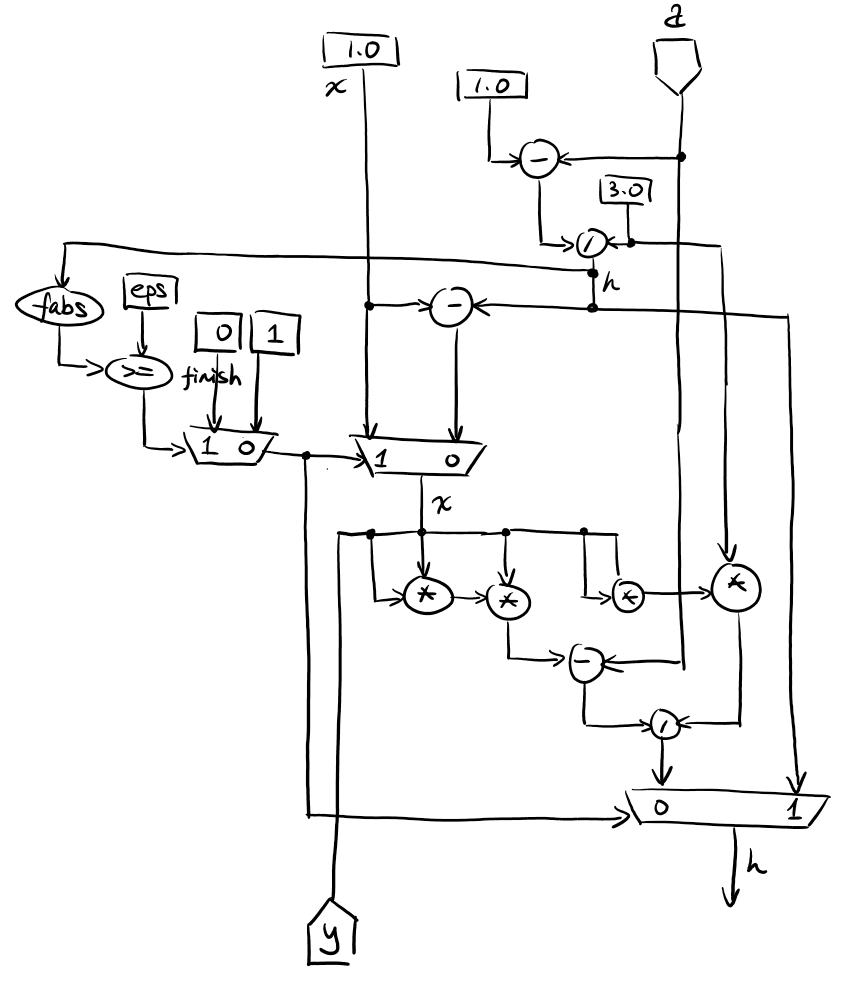


Question 3

a.



b.



c.

max(4N/B, N\*Tc)

[No idea about this]

d.

We need to set the maximum number of iterations. If we know this number (might not always be the case), it can be way higher than the average number of iterations, which means that most of the time we would be doing useless work. It can also happen that the unrolled loop won’t fit on the chip.

Question 4

[NON-EXAMINABLE]