Aly Sultan

001057787

Professor Gunnar Schirner

Lab4

Questions

2.1.2 Assuming a timescale of 1 ns, a wait for of 1E8 would result in a 100 ms delay for each image. Unfortunately if you attempt to apply a finer grain waitfor at each row of the image, you will find that since the number of rows is 240, 1E8 will not result in a whole number when divided by 240. As a result, an additional adjustment wait for of 160 ns will need to be added after the rows of the image are sent. To prevent timing per row from drifting, a finer grained delay was applied to the entire sending loop to achieve the requested 100 ms delay with no adjustments.

2.2.2 If the stimulus stops sending, the qRx behavior will wait on the channel which will lead to a cascade of waits in the pipe and thus monitor will be starved. There will be no way to avoid this deadlock because qRx will never be released from the wait, unless stimulus starts sending empty rows right after the image.

2.2.3

The Convolution behavior will only start in the pipeline after 3 cycles, however, when that happens, the first row will be at the third input of the behavior and not the second. Hence, it will never be picked up. The solution is to send down an empty row before the image that acts as both padding and as garbage data until Conv wakes up and starts working.

2.2.5

```
RX 0: 00 01 02 03 04 05 06 07 08 09  (3)
RX 1: 10 11 12 13 14 15 16 17 18 19  (4)
RX 2: 20 21 22 23 24 25 26 27 28 29  (5)
RX 3: 30 31 32 33 34 35 36 37 38 39  (6)
RX 4: 40 41 42 43 44 45 46 47 48 49  (7)
```

2.2.6

Kernel = {{1,0,0},{0,0,0},{0,0,0}}

```
RX 0: 00 00 00 00 00 00 00 00 00 00  (3)
RX 1: 00 00 01 02 03 04 05 06 07 08  (4)
RX 2: 00 10 11 12 13 14 15 16 17 18  (5)
RX 3: 00 20 21 22 23 24 25 26 27 28  (6)
RX 4: 00 30 31 32 33 34 35 36 37 38  (7)
```

2.2.7

kernel = {{0,0,0},{0,0,0},{0,0,1}}

```
RX 0: 11 12 13 14 15 16 17 18 19 00  (3)
RX 1: 21 22 23 24 25 26 27 28 29 00  (4)
RX 2: 31 32 33 34 35 36 37 38 39 00  (5)
RX 3: 41 42 43 44 45 46 47 48 49 00  (6)
RX 4: 00 00 00 00 00 00 00 00 00 00  (7)
```

2.3.2
The output was distorted (visually) for 2 reasons, left and right padding was not present (I implemented that) and casting from float to unsigned char when the float value was negative or greater than 255 caused distortion. Unfortunately after fixing both of those issues, the results will certain compatible at the binary level, however they were visually compatible. This might be due to floating point truncation/ rounding/ observing boundaries.

3.3
My efforts were not successful when it came to matching the outputs at the binary level for 2.3.2. However, this is an issue that was also present in 2.2.5. I applied the identity convolution to the input Image using the non-count stimulus behavior (basically output = input), and I found that even though the actual output was identical at the binary level. Diff, still produced a difference that made absolutely no sense. It marked the first lines as different, and gave the following output:

```
-bash-4.2$ diff ../img/beachbus-edges.pgm ../img/beachbus.pgm
1,4c1,4
< P5
< # Created by IrfanView
< 320 240
< 255
---
> P5
> # Created by IrfanView
> 320 240
> 255
```

All these lines are identical. Additionally, when I check the output using difference package in sublime text, I get no reported difference.