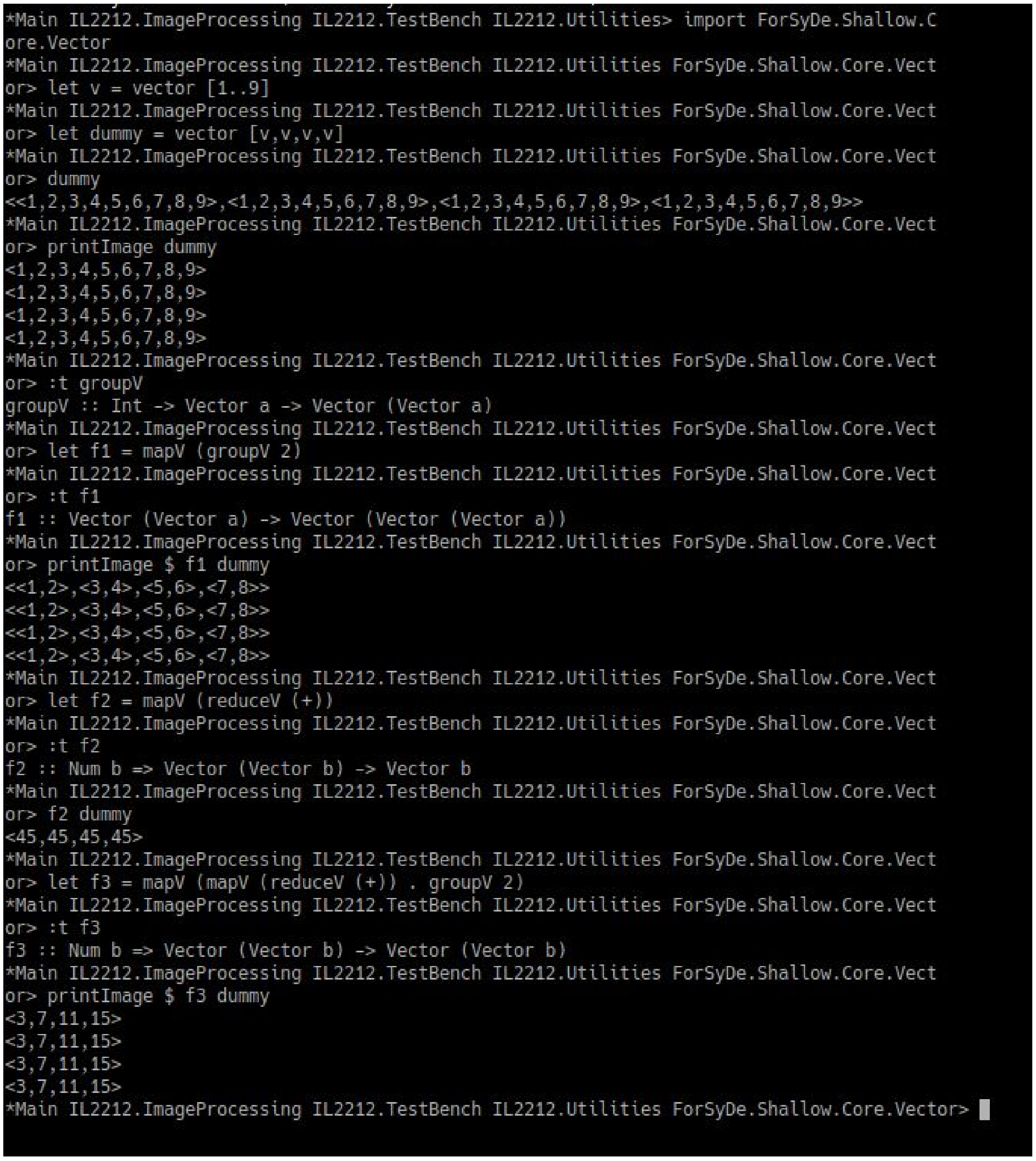
5.

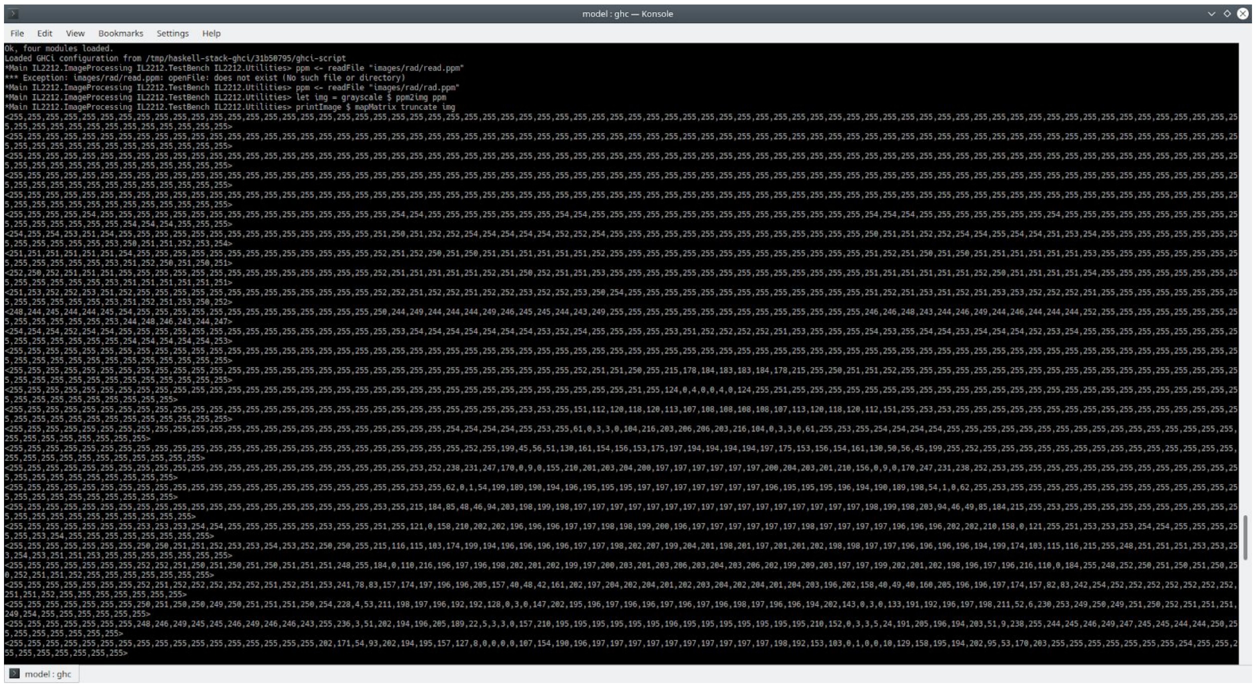
- Resized a test image using the resize function, also used a dummy image matrix and implemented the function´s source code on the image and retrieved a smaller image.



```
*Main IL2212.ImageProcessing IL2212.TestBench IL2212.Utilities> import ForSyDe.Shallow.C
ore.Vector
*Main IL2212.ImageProcessing IL2212.TestBench IL2212.Utilities ForSyDe.Shallow.Core.Vect
or> let v = vector [1..9]
*Main IL2212.ImageProcessing IL2212.TestBench IL2212.Utilities ForSyDe.Shallow.Core.Vect
or> let dummy = vector [v,v,v,v]
*Main IL2212.ImageProcessing IL2212.TestBench IL2212.Utilities ForSyDe.Shallow.Core.Vect
or> dummy
<<1,2,3,4,5,6,7,8,9>,<1,2,3,4,5,6,7,8,9>,<1,2,3,4,5,6,7,8,9>,<1,2,3,4,5,6,7,8,9>>
*Main IL2212.ImageProcessing IL2212.TestBench IL2212.Utilities ForSyDe.Shallow.Core.Vect
or> printImage dummy
<1,2,3,4,5,6,7,8,9>
<1,2,3,4,5,6,7,8,9>
<1,2,3,4,5,6,7,8,9>
<1,2,3,4,5,6,7,8,9>
*Main IL2212.ImageProcessing IL2212.TestBench IL2212.Utilities ForSyDe.Shallow.Core.Vect
or> :t groupV
groupV :: Int -> Vector a -> Vector (Vector a)
*Main IL2212.ImageProcessing IL2212.TestBench IL2212.Utilities ForSyDe.Shallow.Core.Vect
or> let f1 = mapV (groupV 2)
*Main IL2212.ImageProcessing IL2212.TestBench IL2212.Utilities ForSyDe.Shallow.Core.Vect
or> :t f1
f1 :: Vector (Vector a) -> Vector (Vector (Vector a))
*Main IL2212.ImageProcessing IL2212.TestBench IL2212.Utilities ForSyDe.Shallow.Core.Vect
or> printImage $ f1 dummy
<<1,2>,<3,4>,<5,6>,<7,8>>
<<1,2>,<3,4>,<5,6>,<7,8>>
<<1,2>,<3,4>,<5,6>,<7,8>>
<<1,2>,<3,4>,<5,6>,<7,8>>
*Main IL2212.ImageProcessing IL2212.TestBench IL2212.Utilities ForSyDe.Shallow.Core.Vect
or> let f2 = mapV (reduceV (+))
*Main IL2212.ImageProcessing IL2212.TestBench IL2212.Utilities ForSyDe.Shallow.Core.Vect
or> :t f2
f2 :: Num b => Vector (Vector b) -> Vector b
*Main IL2212.ImageProcessing IL2212.TestBench IL2212.Utilities ForSyDe.Shallow.Core.Vect
or> f2 dummy
<45,45,45,45>
*Main IL2212.ImageProcessing IL2212.TestBench IL2212.Utilities ForSyDe.Shallow.Core.Vect
or> let f3 = mapV (mapV (reduceV (+)) . groupV 2)
*Main IL2212.ImageProcessing IL2212.TestBench IL2212.Utilities ForSyDe.Shallow.Core.Vect
or> :t f3
f3 :: Num b => Vector (Vector b) -> Vector (Vector b)
*Main IL2212.ImageProcessing IL2212.TestBench IL2212.Utilities ForSyDe.Shallow.Core.Vect
or> printImage $ f3 dummy
<3,7,11,15>
<3,7,11,15>
<3,7,11,15>
<3,7,11,15>
*Main IL2212.ImageProcessing IL2212.TestBench IL2212.Utilities ForSyDe.Shallow.Core.Vector>
```

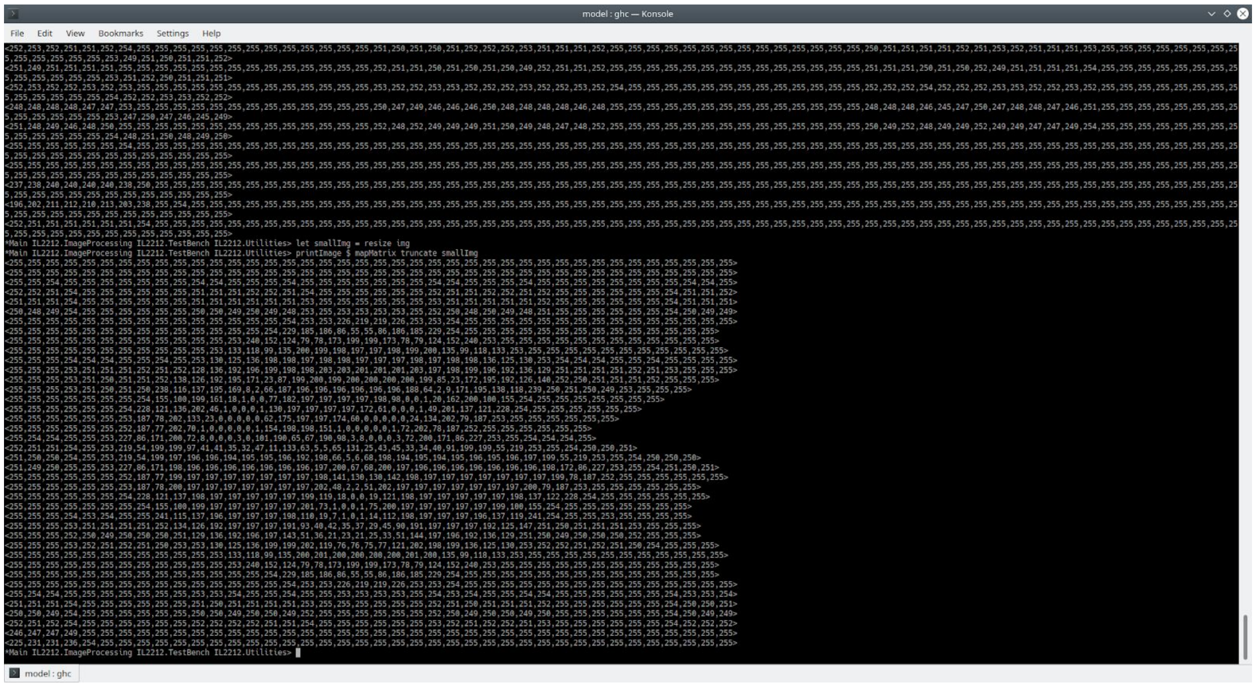
Also implemented the grayscale function on an input color image to retrieve an image with weighted gray values in the matrix instead of color values.

From this image



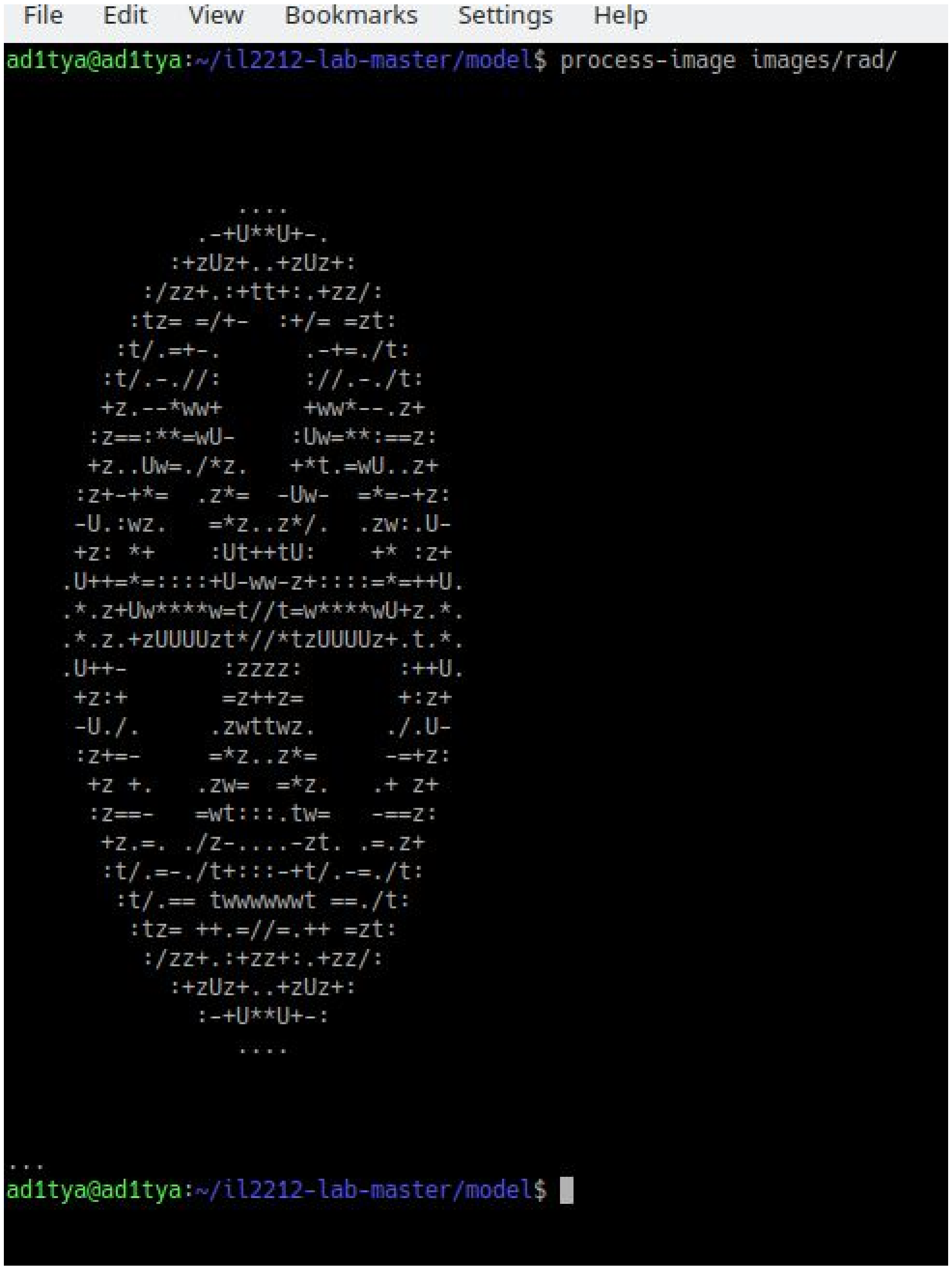
```
Ok, four modules loaded.
Loaded GHCi configuration from /tmp/haskell-stack-ghci/31b50795/ghci-script
*Main IL2212.ImageProcessing IL2212.TestBench IL2212.Utilities> ppm <- readFile "images/rad/read.ppm"
*** Exception: images/rad/read.ppm: openFile: does not exist (No such file or directory)
*Main IL2212.ImageProcessing IL2212.TestBench IL2212.Utilities> ppm <- readFile "images/rad/rad.ppm"
*Main IL2212.ImageProcessing IL2212.TestBench IL2212.Utilities> let img = grayscale $ ppm2img ppm
*Main IL2212.ImageProcessing IL2212.TestBench IL2212.Utilities> printImage $ mapMatrix truncate img
<255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,25
5,255,255,255,255,255,255,255,255,255,255,255,255>
<255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,25
5,255,255,255,255,255,255,255,255,255,255,255,255>
<255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,25
5,255,255,255,255,255,255,255,255,255,255,255,255>
<255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,25
5,255,255,255,255,255,255,255,255,255,255,255,255>
<255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,25
5,255,255,255,255,255,255,255,255,255,255,255,255>
<255,255,255,255,254,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,254,254,255,255,255,255,255,255,255,255,254,254,255,255,255,255,255,255,255,255,255,255,255,255,255,255,254,254,254,255,255,255,255,255,254,255,255,255,255,255,255,255,255,255,255,255,25
5,255,255,255,255,255,255,254,254,254,255,255,255>
<254,255,254,253,251,254,255,255,255,255,255,255,255,255,255,255,255,255,255,255,251,250,251,252,252,254,254,254,254,254,252,252,254,255,255,255,255,255,255,255,255,255,255,255,255,255,255,250,251,251,252,252,254,254,255,254,254,251,253,254,255,255,255,255,255,255,255,25
5,255,255,255,255,255,253,250,251,251,252,253,254>
<251,251,251,251,251,251,254,255,255,255,255,255,255,255,255,255,255,255,255,255,252,251,252,250,251,250,251,251,251,251,251,252,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,251,252,251,250,251,250,251,251,251,251,251,251,253,255,255,255,255,255,255,255,25
5,255,255,255,255,255,253,251,252,250,251,250,251>
<252,250,252,251,251,251,255,255,255,255,255,255,255,255,255,255,255,255,255,255,252,251,251,251,251,251,252,251,250,252,251,251,253,255,255,255,255,255,255,255,255,255,255,255,255,255,255,251,251,251,251,251,251,252,250,251,251,251,251,254,255,255,255,255,255,255,255,25
5,255,255,255,255,255,253,251,251,251,251,251,251>
<251,253,252,252,253,251,252,255,255,255,255,255,255,255,255,255,255,255,255,255,252,252,251,252,252,251,252,252,253,252,252,253,250,254,255,255,255,255,255,255,255,255,255,255,255,255,255,251,252,251,253,251,252,251,253,253,252,252,252,251,255,255,255,255,255,255,255,25
5,255,255,255,255,255,253,251,252,251,253,250,252>
<248,244,245,244,244,245,254,255,255,255,255,255,255,255,255,255,255,255,255,255,250,244,249,244,244,244,249,246,245,245,244,243,249,255,255,255,255,255,255,255,255,255,255,255,255,255,255,246,246,248,243,244,246,249,244,246,244,244,244,252,255,255,255,255,255,255,255,25
5,255,255,255,255,255,253,244,248,246,243,244,247>
<254,254,254,252,254,254,255,255,255,255,255,255,255,255,255,255,255,255,255,255,253,254,254,254,254,254,254,254,253,252,254,255,255,255,255,253,251,252,252,252,252,251,253,255,255,255,255,254,253,255,254,254,253,254,254,254,252,253,254,255,255,255,255,255,255,255,255,25
5,255,255,255,255,255,255,254,254,254,254,254,253>
<255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,25
5,255,255,255,255,255,255,255,255,255,255,255,255>
<255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,252,251,251,250,255,215,178,184,183,183,184,178,215,255,250,251,251,252,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,25
5,255,255,255,255,255,255,255,255,255,255,255,255>
<255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,251,255,124,0,4,0,0,4,0,124,255,251,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,25
5,255,255,255,255,255,255,255,255,255>
<255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,253,253,255,151,112,120,118,120,113,107,108,108,108,108,107,113,120,118,120,112,151,255,253,253,255,255,255,255,255,255,255,255,255,255,255,255,255,255,25
5,255,255,255,255,255,255,255,255,255,255,255,255>
<255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,254,254,254,254,255,253,255,61,0,3,3,0,104,216,203,206,206,203,216,104,0,3,3,0,61,255,253,255,254,254,254,254,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,
255,255,255,255,255,255,255,255>
<255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,252,255,199,45,56,51,130,161,154,156,153,175,197,194,194,194,194,197,175,153,156,154,161,130,50,56,45,199,255,252,255,255,255,255,255,255,255,255,255,255,255,255,255,
255,255,255,255,255,255,255,255,255,255,255>
<255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,253,252,238,231,247,170,0,9,0,155,210,201,203,204,200,197,197,197,197,197,197,200,204,203,201,210,156,0,9,0,170,247,231,238,252,253,255,255,255,255,255,255,255,255,255,25
5,255,255,255,255,255,255,255,255,255>
<255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,253,255,62,0,1,54,199,189,190,194,196,195,195,195,197,197,197,197,197,197,197,196,195,195,195,196,194,190,189,198,54,1,0,62,255,253,255,255,255,255,255,255,255,255,255,25
5,255,255,255,255,255,255,255,255,255>
<255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,253,255,215,184,85,48,46,94,203,198,199,198,197,197,197,197,197,197,197,197,197,197,197,197,198,199,198,203,94,46,49,85,184,215,255,253,255,255,255,255,255,255,255,255,25
5,255,255,255,255,255,255,255,255,255,255>
<255,255,255,255,255,255,255,253,253,253,254,254,255,255,255,255,255,253,255,255,251,255,121,0,158,210,202,202,196,196,196,197,197,198,198,199,200,196,197,197,197,197,197,197,198,197,197,197,197,196,196,196,202,202,210,158,0,121,255,251,253,253,253,254,254,255,255,255,25
5,255,253,254,255,255,255,255,255,255,255,255>
<255,255,255,255,255,255,255,250,250,251,251,252,253,253,254,253,252,250,250,255,215,116,115,103,174,199,194,196,196,196,196,197,197,198,202,207,199,204,201,198,201,197,201,201,202,198,198,197,197,196,196,196,196,194,199,174,103,115,116,215,255,248,251,251,251,253,253,25
3,254,253,251,251,253,255,255,255,255,255,255,255>
<255,255,255,255,255,255,255,252,252,251,250,251,250,251,250,251,251,251,248,255,184,0,110,216,196,197,196,198,202,201,202,199,197,200,203,201,203,206,203,204,203,206,202,199,209,203,197,197,199,202,201,202,198,196,197,196,216,110,0,184,255,248,252,250,251,250,251,250,25
0,252,251,251,252,255,255,255,255,255,255,255>
<255,255,255,255,255,255,255,252,251,252,252,252,252,252,251,252,251,253,241,78,83,157,174,197,196,196,205,157,40,48,42,161,202,197,204,202,204,201,202,203,204,202,204,201,204,203,196,202,158,40,49,40,160,205,196,196,197,174,157,82,83,242,254,252,252,252,252,252,252,252,
251,251,252,255,255,255,255,255,255,255>
<255,255,255,255,255,255,255,250,251,250,250,249,250,251,251,251,250,254,228,4,53,211,198,197,196,192,192,128,0,3,0,147,202,195,196,197,196,196,197,196,197,196,198,197,196,196,194,202,143,0,3,0,133,191,192,196,197,198,211,52,6,230,253,249,250,249,251,250,252,251,251,251,
249,254,255,255,255,255,255,255>
<255,255,255,255,255,255,255,248,246,249,245,245,246,249,246,246,243,255,236,3,51,202,194,196,205,189,22,5,3,3,0,157,210,195,195,195,195,195,195,196,195,195,195,195,195,195,195,210,152,0,3,3,5,24,191,205,196,194,203,51,9,238,255,244,245,246,249,247,245,245,244,244,250,25
5,255,255,255,255,255,255>
<255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,202,171,54,93,202,194,195,157,127,8,0,0,0,0,107,154,190,196,197,197,197,197,197,197,197,197,197,197,198,192,153,103,0,1,0,0,10,129,158,195,194,202,95,53,170,203,255,255,255,255,255,255,255,254,255,255,2
55,255,255,255,255,255,255>
```

To this image



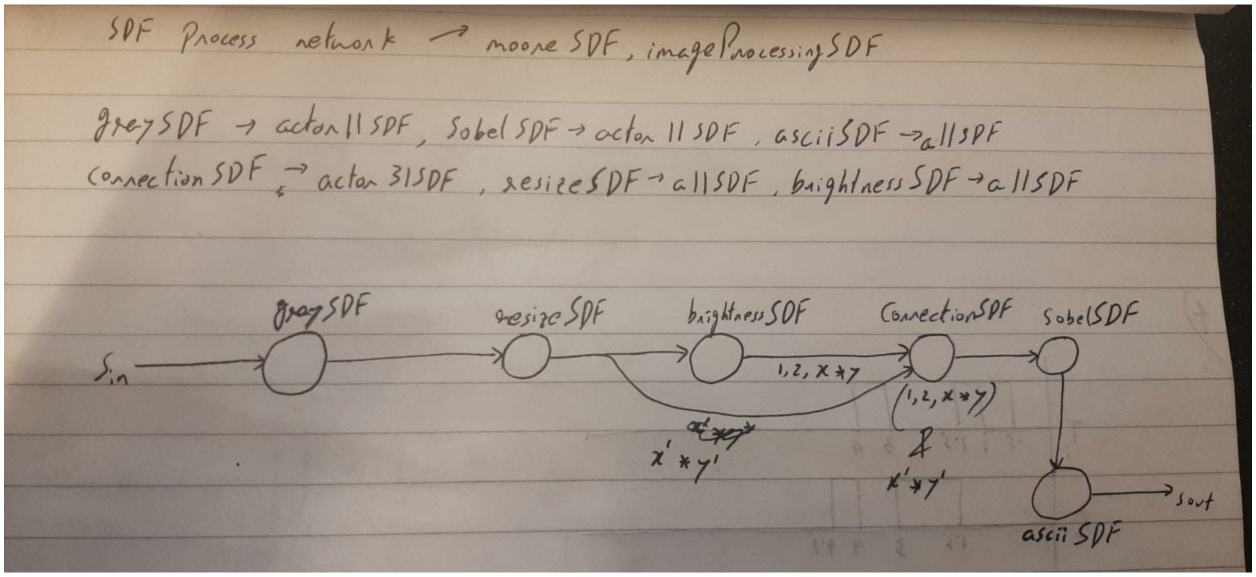
```
<252,253,252,251,251,252,254,255,255,255,255,255,255,255,255,255,255,255,255,255,251,250,251,250,251,252,252,252,253,251,251,251,252,255,255,255,255,255,255,255,255,255,255,255,255,255,255,250,251,251,251,251,252,251,253,252,251,251,251,253,255,255,255,255,255,255,255,25
5,255,255,255,255,255,253,249,251,250,251,251,252>
<251,249,251,251,251,251,255,255,255,255,255,255,255,255,255,255,255,255,255,255,252,251,251,250,251,250,251,250,249,252,251,251,252,255,255,255,255,255,255,255,255,255,255,255,255,255,255,251,251,251,250,251,250,252,249,251,251,251,251,254,255,255,255,255,255,255,255,25
5,255,255,255,255,255,253,251,252,250,251,251,251>
<252,253,252,252,253,252,253,255,255,255,255,255,255,255,255,255,255,255,255,255,253,252,252,253,253,252,252,252,253,252,252,253,252,254,255,255,255,255,255,255,255,255,255,255,255,255,255,252,252,252,254,252,252,252,253,253,252,252,253,252,255,255,255,255,255,255,255,25
5,255,255,255,255,255,254,252,252,253,253,252,252>
<248,248,248,248,247,247,253,255,255,255,255,255,255,255,255,255,255,255,255,255,250,247,249,246,246,246,250,248,248,248,248,246,248,255,255,255,255,255,255,255,255,255,255,255,255,255,255,248,248,248,246,245,247,250,247,248,248,247,246,251,255,255,255,255,255,255,255,25
5,255,255,255,255,255,253,247,250,247,246,245,249>
<251,248,249,246,248,250,255,255,255,255,255,255,255,255,255,255,255,255,255,255,252,248,252,249,249,249,251,250,249,248,247,248,252,255,255,255,255,255,255,255,255,255,255,255,255,255,255,250,249,252,248,249,249,252,249,249,247,247,249,254,255,255,255,255,255,255,255,25
5,255,255,255,255,255,254,248,251,250,248,249,250>
<255,255,255,255,255,255,254,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,25
5,255,255,255,255,255,255,255,255,255,255,255,255>
<255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,25
5,255,255,255,255,255,255,255,255,255,255,255,255>
<237,238,240,240,240,240,238,250,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,25
5,255,255,255,255,255,255,255,255,255,255,255,255>
<196,202,211,212,210,213,203,238,255,254,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,25
5,255,255,255,255,255,255,255,255,255,255,255,255>
<252,251,251,251,251,251,251,254,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,25
5,255,255,255,255,255,255,255,255,255,255,255,255>
*Main IL2212.ImageProcessing IL2212.TestBench IL2212.Utilities> let smallImg = resize img
*Main IL2212.ImageProcessing IL2212.TestBench IL2212.Utilities> printImage $ mapMatrix truncate smallImg
<255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255>
<255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255>
<255,255,254,255,255,255,255,255,255,255,254,254,255,255,255,254,255,255,255,255,255,255,255,254,254,255,255,255,254,255,255,255,255,255,255,255,255,254,254,255>
<252,252,251,254,255,255,255,255,255,255,251,251,251,252,252,251,254,255,255,255,255,255,255,252,251,251,252,252,251,252,255,255,255,255,255,255,254,251,251,252>
<251,251,251,254,255,255,255,255,255,255,251,251,251,251,251,251,253,255,255,255,255,255,255,253,251,251,251,251,251,252,255,255,255,255,255,255,254,251,251,251>
<250,248,249,254,255,255,255,255,255,255,250,250,249,250,249,248,253,255,253,253,253,253,255,252,250,248,250,249,248,251,255,255,255,255,255,255,254,250,249,249>
<255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,254,253,253,226,219,219,226,253,253,254,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255>
<255,255,255,255,255,255,255,255,255,255,255,255,255,255,254,229,185,186,86,55,55,86,186,185,229,254,255,255,255,255,255,255,255,255,255,255,255,255,255,255>
<255,255,255,255,255,255,255,255,255,255,255,255,253,240,152,124,79,78,173,199,199,173,78,79,124,152,240,253,255,255,255,255,255,255,255,255,255,255,255,255>
<255,255,255,255,255,255,255,255,255,255,255,253,133,118,99,135,200,199,198,197,197,198,199,200,135,99,118,133,253,255,255,255,255,255,255,255,255,255,255,255>
<255,255,255,254,254,254,255,255,254,255,253,130,125,136,198,198,197,198,198,197,197,197,198,197,198,198,136,125,130,253,254,254,254,255,255,254,255,255,255,255>
<255,255,255,253,251,251,251,252,251,252,128,136,192,196,199,198,198,203,203,201,201,201,203,197,198,199,196,192,136,129,251,251,251,251,252,251,253,255,255,255>
<255,255,255,253,251,250,251,251,252,138,126,192,195,171,23,87,199,200,199,200,200,200,200,199,85,23,172,195,192,126,140,252,250,251,251,251,252,255,255,255>
<255,255,255,253,251,250,251,250,238,116,137,195,169,8,2,66,187,196,196,196,196,196,196,188,64,2,9,171,195,138,118,239,250,251,250,249,253,255,255,255>
<255,255,255,255,255,255,255,254,155,100,199,161,18,1,0,0,77,182,197,197,197,197,198,98,0,0,1,20,162,200,100,155,254,255,255,255,255,255,255,255>
<255,255,255,255,255,255,254,228,121,136,202,46,1,0,0,0,1,130,197,197,197,197,172,61,0,0,0,1,49,201,137,121,228,254,255,255,255,255,255,255>
<255,255,255,255,255,255,253,187,78,202,133,23,0,0,0,0,0,62,175,197,197,174,60,0,0,0,0,0,24,134,202,79,187,253,255,255,255,255,255,255>
<255,255,255,255,255,255,252,187,77,202,70,1,0,0,0,0,0,1,154,198,198,151,1,0,0,0,0,0,1,72,202,78,187,252,255,255,255,255,255,255>
<255,254,254,255,255,253,227,86,171,200,72,8,0,0,0,3,0,101,190,65,67,190,98,3,8,0,0,0,3,72,200,171,86,227,253,255,254,254,254,255>
<252,251,251,254,255,253,219,54,199,199,97,41,41,35,32,47,11,133,63,5,5,65,131,25,43,45,33,34,40,91,199,199,55,219,253,255,254,250,250,251>
<251,250,250,254,255,253,219,54,199,197,196,196,194,195,195,196,192,198,66,5,6,68,198,194,195,194,195,196,195,196,197,199,55,219,253,255,254,250,250,250>
<251,249,250,255,255,253,227,86,171,198,196,196,196,196,196,196,196,197,200,67,68,200,197,196,196,196,196,196,196,196,198,172,86,227,253,255,254,251,250,251>
<255,255,255,255,255,255,252,187,77,199,197,197,197,197,197,197,197,198,141,130,130,142,198,197,197,197,197,197,197,197,199,78,187,252,255,255,255,255,255,255>
<255,255,255,255,255,255,253,187,78,200,197,197,197,197,197,197,197,202,48,2,2,51,202,197,197,197,197,197,197,197,200,79,187,253,255,255,255,255,255,255>
<255,255,255,255,255,255,254,228,121,137,198,197,197,197,197,197,199,119,18,0,0,19,121,198,197,197,197,197,197,198,137,122,228,254,255,255,255,255,255,255>
<255,255,255,255,255,255,255,254,155,100,199,197,197,197,197,197,201,73,1,0,0,1,75,200,197,197,197,197,197,199,100,155,254,255,255,255,255,255,255,255>
<255,255,255,254,253,254,255,255,241,115,137,196,197,197,197,198,110,19,7,1,0,1,14,112,198,197,197,197,196,137,119,241,254,255,255,253,255,255,255,255>
<255,255,255,253,251,251,251,251,252,134,126,192,197,197,197,191,93,40,42,35,37,29,45,90,191,197,197,197,192,125,147,251,250,251,251,251,253,255,255,255>
<255,255,255,252,250,249,250,250,250,251,129,136,192,196,197,143,51,36,21,23,21,25,33,51,144,197,196,192,136,129,251,250,249,250,250,250,252,255,255,255>
<255,255,255,253,252,251,252,251,250,253,253,130,125,136,199,199,202,119,76,76,75,77,121,202,198,199,136,125,130,253,252,252,251,252,251,250,254,255,255,255>
<255,255,255,255,255,255,255,255,255,255,255,253,133,118,99,135,200,201,200,200,200,200,201,200,135,99,118,133,253,255,255,255,255,255,255,255,255,255,255,255>
<255,255,255,255,255,255,255,255,255,255,255,255,253,240,152,124,79,78,173,199,199,173,78,79,124,152,240,253,255,255,255,255,255,255,255,255,255,255,255,255>
<255,255,255,255,255,255,255,255,255,255,255,255,255,255,254,229,185,186,86,55,55,86,186,185,229,254,255,255,255,255,255,255,255,255,255,255,255,255,255,255>
<255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,254,253,253,226,219,219,226,253,253,254,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255>
<255,254,254,255,255,255,255,255,255,255,253,253,254,255,255,254,255,255,253,253,253,253,255,254,253,254,255,255,254,254,255,255,255,255,255,255,254,253,253,254>
<251,251,251,254,255,255,255,255,255,255,251,250,251,251,251,251,253,255,255,255,255,255,255,252,251,250,251,251,251,252,255,255,255,255,255,255,254,250,250,251>
<250,250,249,254,255,255,255,255,255,255,250,250,249,250,250,249,252,255,255,255,255,255,255,252,250,249,250,250,249,250,255,255,255,255,255,255,254,250,249,249>
<252,251,252,254,255,255,255,255,255,255,252,252,252,252,251,251,254,255,255,255,255,255,255,253,252,251,252,252,251,253,255,255,255,255,255,255,254,252,252,252>
<246,247,247,249,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255>
<225,231,231,236,254,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255>
*Main IL2212.ImageProcessing IL2212.TestBench IL2212.Utilities> 
```

- Created a ppm image from a .png image, and processed the same. The image I've used is a nuclear hazard logo, the weird aspect ratio is resulted from me experimenting with the image resize part.



```
ad1tya@ad1tya:~/il2212-lab-master/model$ process-image images/rad/




                 ....
              .-+U**U+-.
            :+zUz+..+zUz+:
          :/zz+.:+tt+:.+zz/:
         :tz= =/+-  :+/= =zt:
        :t/.=+-.      .-+=./t:
       :t/.-.//:      ://.-./t:
       +z.--*ww+      +ww*--.z+
      :z==:**=wU-    :Uw=**:==z:
      +z..Uw=./*z.   +*t.=wU..z+
     :z+-+*=  .z*=  -Uw-  =*=-+z:
     -U.:wz.   =*z..z*/.  .zw:.U-
     +z: *+    :Ut++tU:    +* :z+
    .U++=*=::::+U-ww-z+::::=*=++U.
    .*.z+Uw****w=t//t=w****wU+z.*.
    .*.z.+zUUUUzt*//*tzUUUUz+.t.*.
    .U++-       :zzzz:       :++U.
     +z:+       =z++z=       +:z+
     -U./.     .zwttwz.     ./.U-
     :z+=-     =*z..z*=     -=+z:
      +z +.   .zw=  =*z.   .+ z+
      :z==-   =wt:::.tw=   -==z:
       +z.=. ./z-....-zt. .=.z+
       :t/.=-./t+:::-+t/.-=./t:
        :t/.== twwwwwwt ==./t:
         :tz= ++.=//=.++ =zt:
          :/zz+.:+zz+:.+zz/:
            :+zUz+..+zUz+:
              :-+U**U+-:
                 ....



...
ad1tya@ad1tya:~/il2212-lab-master/model$
```

- The SDF graph implemented in the project is illustrated below, please note that this graph doesn't include the actors involved in implementing the Moore state machine and the state detection and next state mechanisms, this SDF graph only illustrates the actors involved in the core image processing application. All the actors are of type actor11SDF, other than the correction SDF actor which is of type actor31SDF, ie, it has 3 inputs.


SDF Process network → moore SDF, imageProcessingSDF

greySDF → actor11SDF, Sobel SDF → actor 11 SDF, asciiSDF → a11SDF
CorrectionSDF → actor 31SDF, resizeSDF → a11SDF, brightnessSDF → a11SDF

- Description of the SDF actors:

  graySDF: It converts the input RGB matrix image into a grayscale matrix. It is done by multiplying the input samples with a weight value and adding them together, a process known as convolution.

  resizeSDF: It takes in the input grayscale image and resizes it into a smaller image by halving the x&y coordinates of the image.

  brightnessSDF: Brightness of the image is adjusted by finding the minimum and maximum brightness of the given image, this SDF only finds the minimum and maximum brightness.

  correctionSDF: Here the brightness is actually adjusted by taking the min and max brightness values and finding the difference between them and multiplying it with 2,4,8 or 16 based on their difference.

  sobelSDF: It uses an edge detection algorithm that detects the edges of the objects in the image based on strong intensity contrasts asciiSDF: This SDF takes the final image and generates ascii art using ascii characters instead of gray values in the image matrix, producing a replicated image with ascii characters instead of gray values.

e) Description of arcs (signal) are as follows:

The arc going into the graySDF has the data type int, thus ideally having a token size of 32 bits. It is important to note that the input to graySDF is not one integer, but an image of x*y integers. Assuming the image is of 8x8 pixels, the size of the token would be 2048 bits.

The arc coming out of the graySDF actor and going into the resizeSDF actor has a datatype of double, that is, 64 bits. The resizeSDF crops the image by taking in the x*y image and outputting an image of the coordinates (x/2)*(y/2), thus giving a token size of 1024 bits.

The arc coming out of the resizeSDF and going into the brightnessSDF has datatype Double, and the size is 1024 bits. The brightness SDF outputs the maximum and minimum brightness of the image as a list with 2 numbers of the double type, we can say that the token size is 128 bits.

The actor correctionSDF takes in 3 inputs, the list with 2 numbers, max and min from the brightnessSDF and the image from the resizeSDF. The output is a brightness corrected image of the same size as that of the output of resize actor, just corrected by brightness. The datatype is double, and it is a matrix, the size of the token being 1024 bits.

The arc leading into sobelSDF has the type Double and size 1024 bits, and the sobelSDF just uses an edge detection algorithm and outputs an image of the type double. Here also, the output of sobelSDF is an image of type Double and token size is 1024 bits. I believe that the sobel algorithm reduces the image size, but I'm not sure of by how much, I think it is (x/2)-2 * (y/2)-2 but I can't say for sure. In this case the token size would be 128 bits for an 8x8 image.

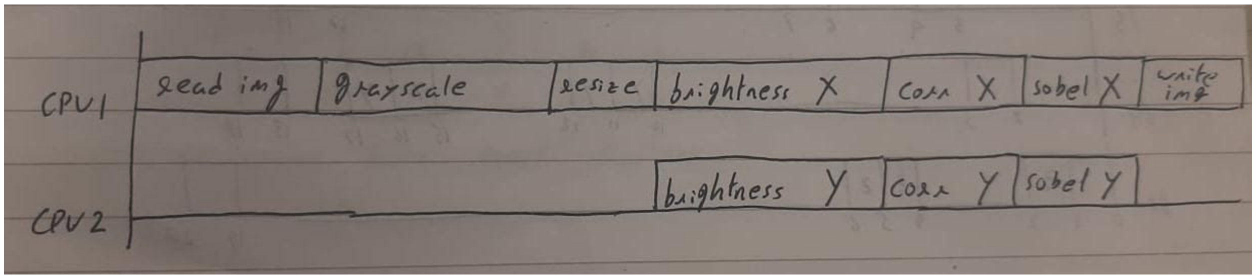
The arc coming out of the sobel SDF and going into the asciiSDF actor is an image of type double and token size 1024 bits. The asciiSDF actor implements the toAsciiArt algorithm on the image and outputs an image of the type char, making the token size 128 bits. If the sobel actor actually reduces the image size, then the output token size would be 32 bits.

The final output of the system is an image of datatype char, and the token size is 128 bits for an 8x8 image.

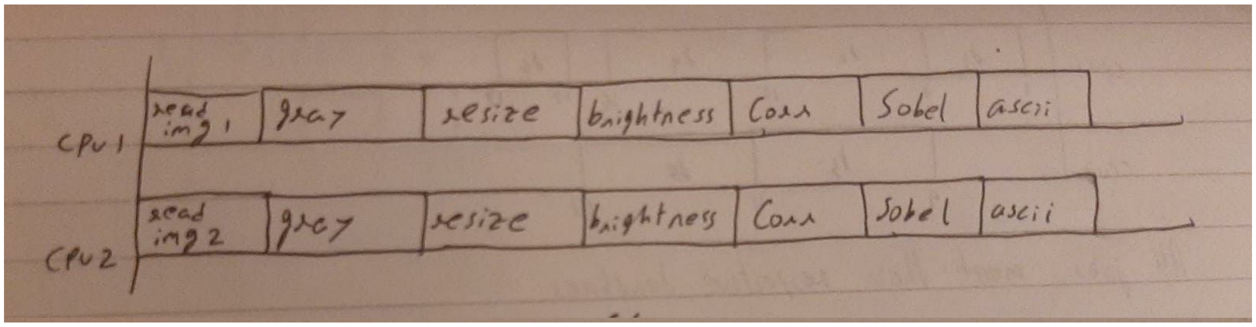
Other than these actors and arcs, the system consists of control arcs which are use the datatype control, and has a token size of 8 bits.

f) The entire dataflow of the network is sequential, and it be perfectly implemented on a single core system, be it bare metal or with an RTOS.

An interesting approach to parallelizing the flow would be to use two processors to process the x coordinates and y coordinates individually on each processor. The implementation would look something like this:



Another approach, and a more realistic one would be to input multiple images, and process them parallelly with the same sequence on multiple processors. Ie, the images are read, and then each processor runs the sequential algorithm on each image, so the number of images that can be processed is equal to the number of processors.



.

g) The formula to calculate the runtime of the application would simply be an addition of the runtime of each actors if we ignore the communication speeds. Say, the computation time of an actor is Ci, then the formula for runtime of the entire application is $\Sigma\, Ci$ where i goes from 1 to 6, as there are 6 SDF actors involved in the image processing aspect of the application. The actors involving mooreSDF and imageProcessingSDF is ignored here.

h) According to their estimated runtime, the actors ranked from slowest to fastest are :
graySDF, resizeSDF, sobelSDF, correctionSDF, brightnessSDF, asciiSDF
The computation time of each actor depend on the image size as well. However, the resize actor reduces the image size by half, so the actors following the resizeSDF have a much smaller computation time.
Say, for a 32x32 image, the runtime for resizeSDF is 1000 time units, then for graySDF it is 1200 time units. SobelSDF takes about 900 time units because of the huge amount of computation and runtime of the sobel algorithm, then correctionSDF takes about 600 time units, brightnessSDF takes 300 time units, and asciiSDF takes about 200 time units.
Using the formula, the total runtime of the application is 4200 time units.