

**Nome:**

Allan Baldissin

Danilo Alves

Marcela Tiemi Shinzato

**nUSP:**

8657904

10408390

10276953

## **SSC0740 - Sistemas Embarcados**

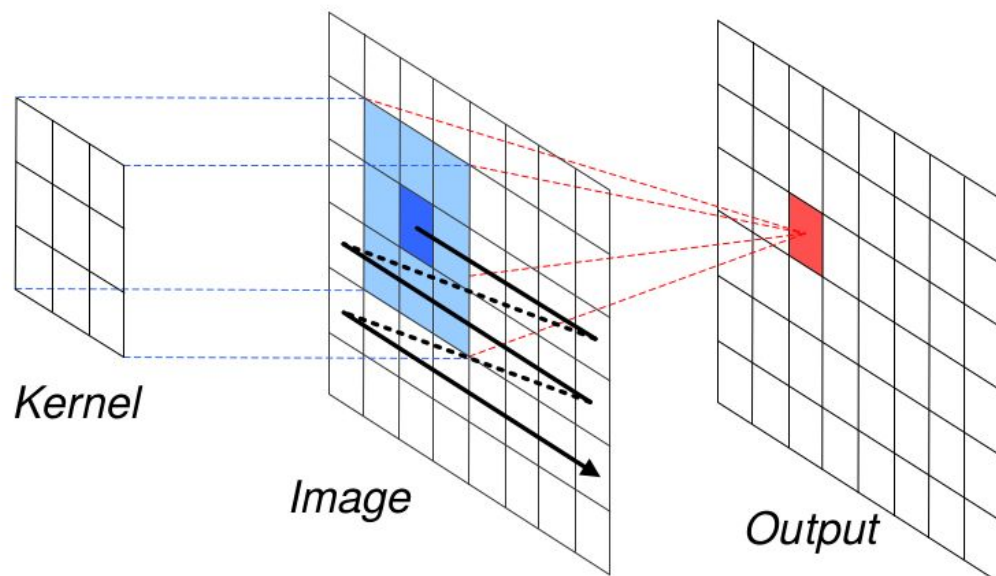
**Professor: Vanderlei Bonato**

### **Detecc o de bordas por Sobel**

O filtro de Sobel   uma opera o utilizada para em processamento de imagem para detec o de descontinuidades de uma imagem, detectando bordas horizontais e verticais.

  calculado gradiente da intensidade da imagem em cada ponto, dando a dire o da maior varia o de claro para escuro e a quantidade de varia o nessa dire o. O valor do gradiente   aplicado nos pixels e, como essas grandes mudan as entre claro-escuro indicam as bordas da imagem,   f cil fazer a detec o delas.

Neste trabalho foi implementado o algoritmo de Sobel em Verilog, usando o compilador Icarus Verilog, vers o 10.3. O algoritmo consiste em descrever um Kernel 3x3 e aplicar os devidos c culos de Gradiente em cada pixel, transformando a imagem.



Foi desenvolvido um testbench que recebe um arquivo de imagem em binário cru (utilizamos a extensão *.bin*). Para criar esse arquivo, utilizamos o Octave e criamos um código que transforma uma imagem.png 100x100 em um arquivo binário:

```
intobin.m ✕
1 function intobin(imageName, binName)
2     image = imread(imageName);
3     grayImg = rgb2gray(image);
4     fileID = fopen(binName, "w");
5     fwrite(fileID, grayImg);
6     fclose(fileID);
7 endfunction
8
```

O testbench carrega o arquivo gerado e o percorre píxel a píxel enviando para o módulo de Sobel a 8-vizinhança para o cálculo do pixel resultante, armazenando o resultado na memória, para posteriormente ser gravado em um novo arquivo.

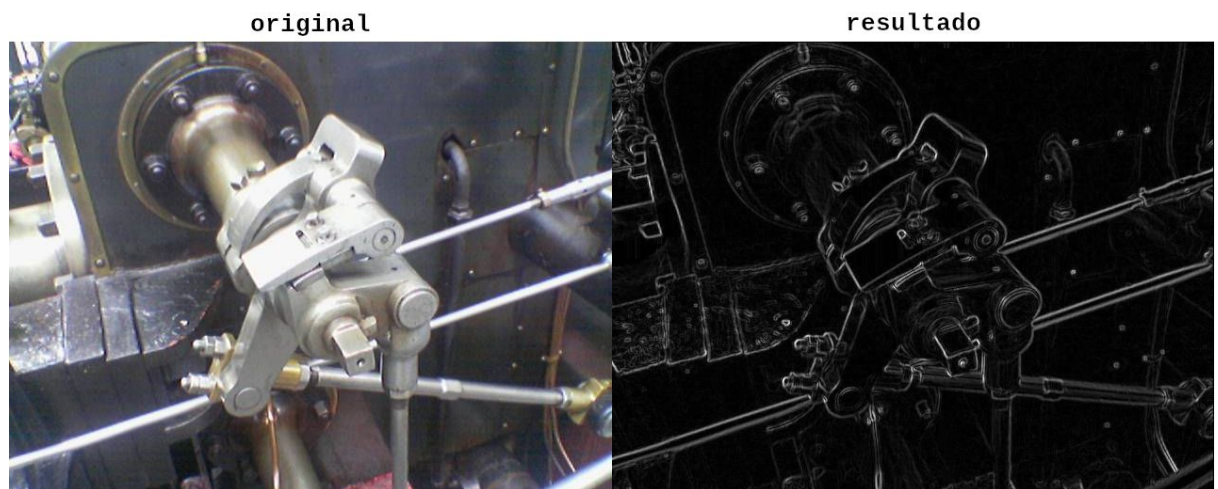
Assim, após a execução do testbench, temos um novo arquivo binário gerado que é novamente convertido usando o Octave para transformá-lo em uma imagem:

```
binToim.m ✕
1 function data = binToim(binName, imageName)
2     fileID = fopen(binName, "r");
3     data = fread(fileID, [100 100], 'int=>double');
4     fclose(fileID);
5     maxValue = max(data(:));
6     data = data / maxValue;
7     imwrite(data, imageName);
8 endfunction
9
```

Resultado:



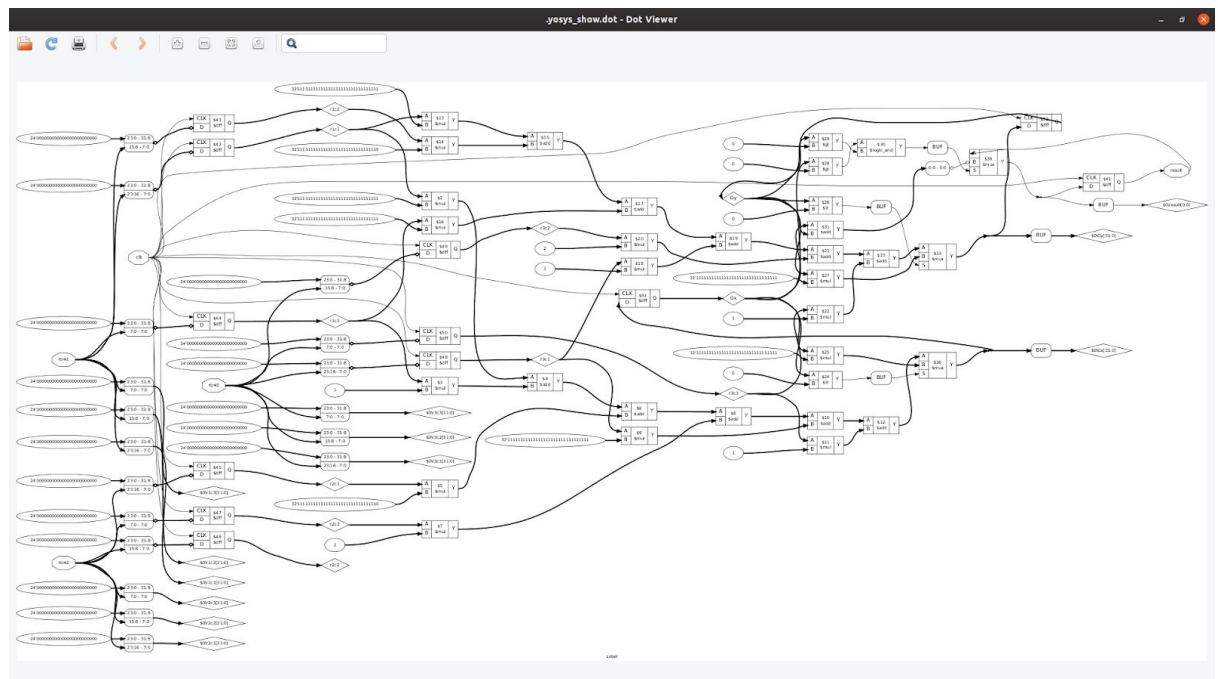
Um outro teste foi realizado com uma imagem de tamanho 640x480, apresentando o seguinte resultado:



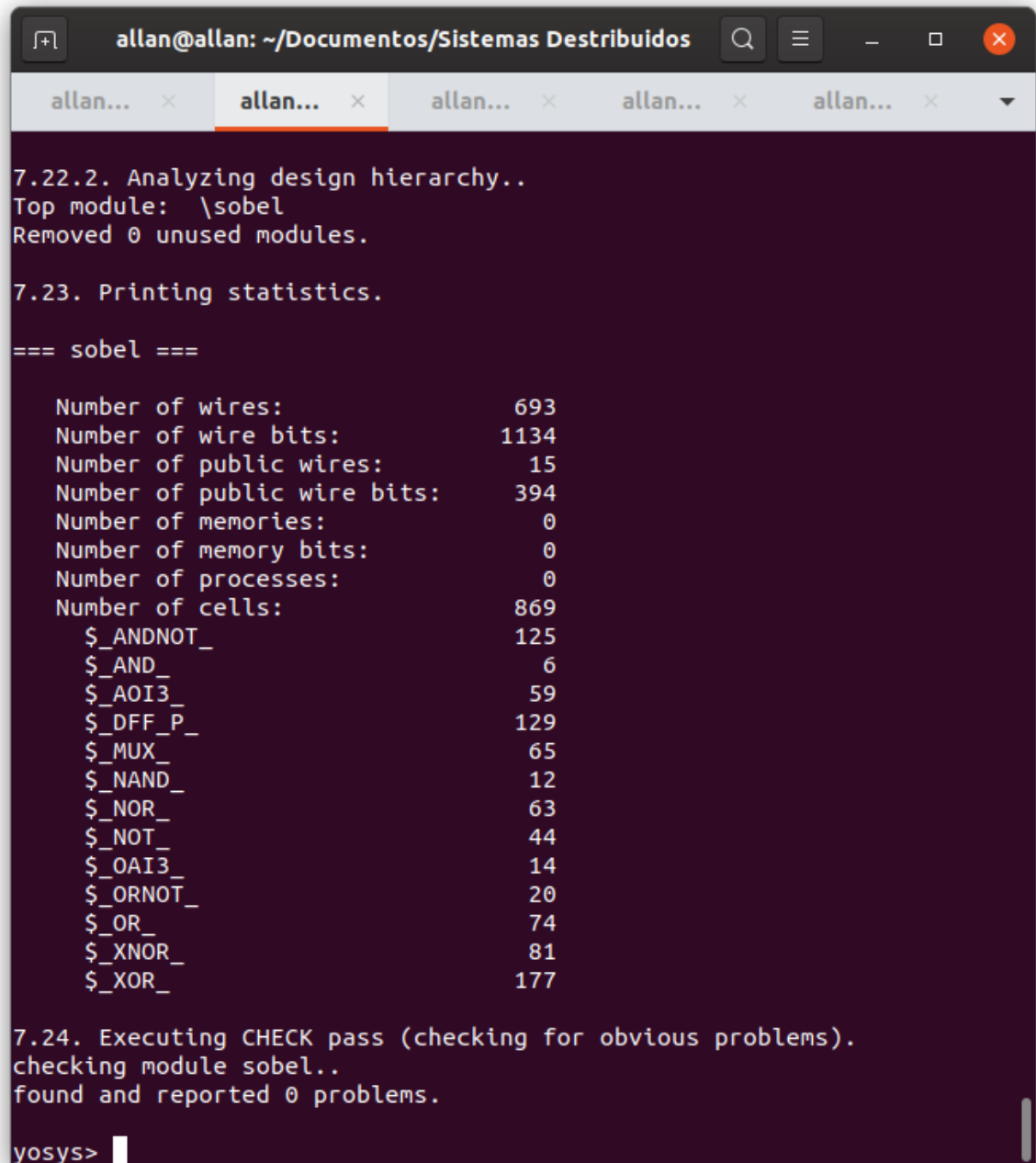
A descrição do circuito e síntese do hardware foi feita utilizando a ferramenta Yosys, versão 0.8. É possível visualizar todo o comportamento do algoritmo e seus componentes:

```
# read design  
read_verilog sobel.v
```

```
# showing result  
show -colors -width
```



```
# generic synthesis  
synth -top sobel
```



The screenshot shows a terminal window with a dark background and light-colored text. The window title is "allan@allan: ~/Documentos/Sistemas Distribuidos". The terminal output displays the results of a Yosys synthesis process for a module named "sobel".

```
allan@allan: ~/Documentos/Sistemas Distribuidos  
7.22.2. Analyzing design hierarchy..  
Top module: \sobel  
Removed 0 unused modules.  
  
7.23. Printing statistics.  
  
=== sobel ===  
  
Number of wires:          693  
Number of wire bits:      1134  
Number of public wires:   15  
Number of public wire bits: 394  
Number of memories:       0  
Number of memory bits:    0  
Number of processes:      0  
Number of cells:          869  
  $_ANDNOT_                125  
  $_AND_                    6  
  $_AOI3_                   59  
  $_DFF_P_                 129  
  $_MUX_                    65  
  $_NAND_                   12  
  $_NOR_                    63  
  $_NOT_                     44  
  $_OAI3_                   14  
  $_ORNOT_                  20  
  $_OR_                      74  
  $_XNOR_                    81  
  $_XOR_                     177  
  
7.24. Executing CHECK pass (checking for obvious problems).  
checking module sobel..  
found and reported 0 problems.  
  
yosys> 
```