ISTANBUL TECHNICAL UNIVERSITY

Digital System Design Applications

Experiment 8
Image Processing System

Preliminary

Students should have information about 2-d convolution. Also they should know implementing Block RAM's on the FPGA.

Objectives

In this experiment, it is expected to you implement an image processing system. The system reads the image from Block RAM as a stream. Then a 2-d Highpass filter is applied to the image. Finally, the result image is written on another Block RAM. In Fig. 1, image processing system and the expected results are given.

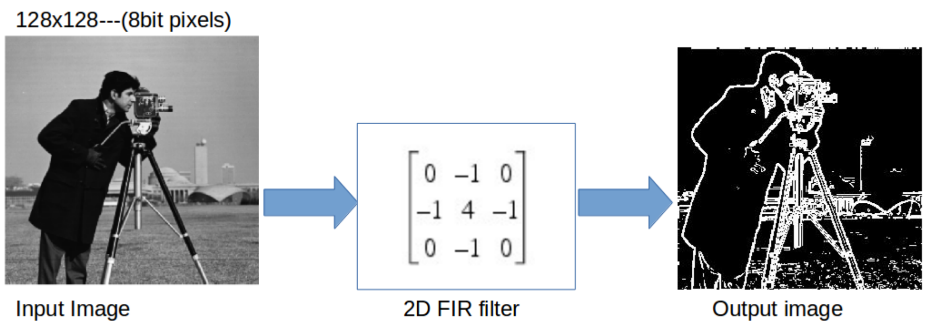
128x128---(8bit pixels)
$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$
Input Image
2D FIR filter
Output image

Figure 1: image processing system and the expected results are given.

End of the experiment, students will be able to

- Implement a circuit with Line buffers used for reading the image as stream and generating the sliding window for convolution.
- Using pipeline method, implement an arithmetic unit for multiplication and sum operations for convolution .

Requirements

Students are expected to be able to

- add Block RAM IPcore to the project
- implement multiplier and summer

References:

1. Nexys2 Reference Manual
2. Spartan-3E Libraries Guide for HDL Designs
3. Constraints Guide
4. http://www.songho.ca/dsp/convolution/convolution.html

Line Buffer Architecture

1. Create a new ISE project with the settings below:
   - Family: **Spartan 3E**
   - Device: **XC3S500E**
   - Package: **FG320**
   - Speed: **-4**
   - Synthesis Tool: **XST (VHDL\Verilog)**
   - Simulator: **ISim (VHDL\Verilog)**
   - Preferred Language: **Verilog**
2. Add a new Verilog module named **imageProcess** to your project. Clear all lines in your module.
3. Add a Block RAM IP to your project. Set the data width as 8 bits and the depth as $130 * 130 = 16900$. Load the initial data(image pixels), from the file named **image.coe** given in Ninova system.
4. Implement the all system given in Fig. 2.
5. Write a post PAR simulation code to check if the window values (w1,w2,w3,...,w9) are generated correctly.
6. Add the timing analysis after PAR to your reports. What is the maximum clock frequency that you can apply the circuit?

Adder Tree

1. Create a new module named **adderTree**.
2. Implement the adder tree given in Fig. 3 using Pipeline method and ***,+** operators. Notice that, for each pipeline stage, you need the store the operation results at proper registers.
3. Write a post PAR simulation code to check whether your adder tree gives true results.
4. Add the timing analysis after PAR to your reports. What is the maximum clock frequency that you can apply the circuit?

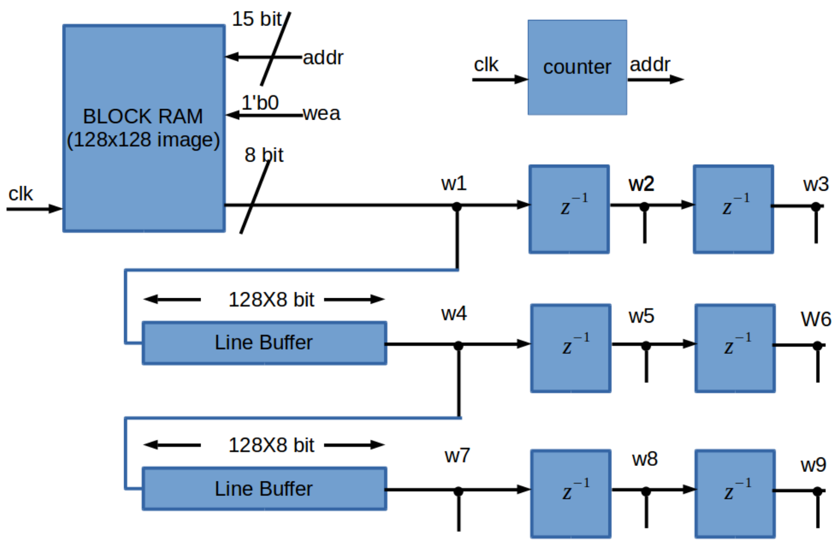

Figure 2: Line Buffer Architecture.

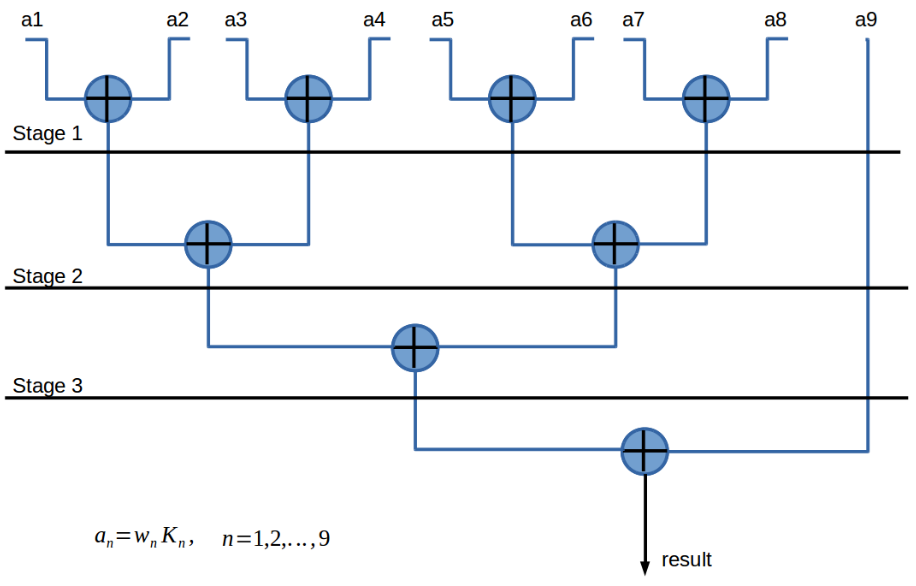

Figure 3: Adder Tree.

Complete System

1. Define the 16 bit wires named a1,a2,...a9.
2. connect the evaluation of $w_n K_n$ operations to a_n, $n = 1,2,\ldots,9$. Take the K_n as given in Fig. 1.
3. Propose a strategy to avoid signed integer operations.

4. Instanstiate the **adderTree** module in the **imageProcess** module. Connect the wires a_n to input port of the adderTree.

5. Instanstiate another Block RAM in the **imageProcess** module. Connect the output **Result** from adderTree to **dina** port of the new Block RAM. Furthermore, define an address wire or register for the BlockRam and control the adreess port for a proper writing.

6. Implement the design. Add the timing analysis after PAR to your reports. What is the maximum clock frequency that you can apply the circuit?

7. Write a post PAR simulation code to check whether your system works correctly. Your test code should write the convolution results to a file.

8. Check the data in the file generated by the test code.

Experiment Report

Each group member is going to prepare his/her own report. Reports should include:

- Screenshots and results asked above
- Your comments on results

Submission includes reports and project files.