Project Proposal: FPGA-Accelerated Medical Image Processing for Real-Time CT Image Filtering

CSEE4840 Embedded Systems, Spring 2026

Kristine Vergara (kev2128), Saha Dev Shanmugam (ss7654), Nitali Arora (na3202)

Introduction

Medical imaging systems such as CT scanners, X-Ray machines, and MRI systems generate images that are commonly stored in DICOM (Digital Imaging and Communications in Medicine) format. DICOM is a standardized protocol used for managing, storing, transmitting and displaying medical imaging data.

Radiologists often need to apply image processing techniques such as edge detection, contrast enhancement and sharpening to better visualize anatomical structures and pathological features. With the rise of smaller and more portable imaging devices, the demand for fast image processing systems capable of providing immediate visual feedback has grown significantly.

Convolution-based edge detection requires a large number of arithmetic operations to be performed for every pixel in the image. Although these operations can be implemented on a CPU, they are inherently parallel and spatially structured, making them well suited for FPGA implementation.

The objective of this project is to develop an FPGA-accelerated image processing pipeline that is capable of applying multiple filters to CT images and displaying the processed result in real time on a VGA monitor.

Embedded System Inputs

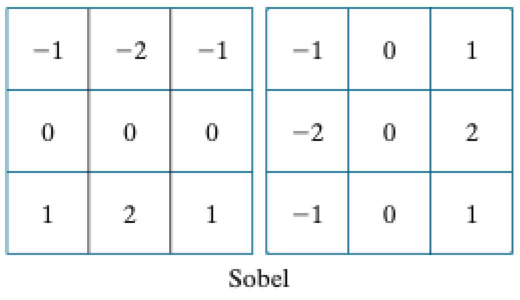
The DICOMs used are 512 x 512 pixel arrays with each pixel being represented by a 16 bit value. Keyboard inputs will be used to select filters and control zooming and panning operations.

Embedded System Outputs

The VGA monitor will be used to display the post processed image.

Algorithms

We selected three algorithms that serve three purposes in processing medical images.The following steps are used to detect and visualize edges in the image: first, the image intensity array pixels are convolved with the Sobel filter, and then the threshold filter is used to identify the edge.



−1	−2	−1
0	0	0
1	2	1

−1	0	1
−2	0	2
−1	0	1

Sobel

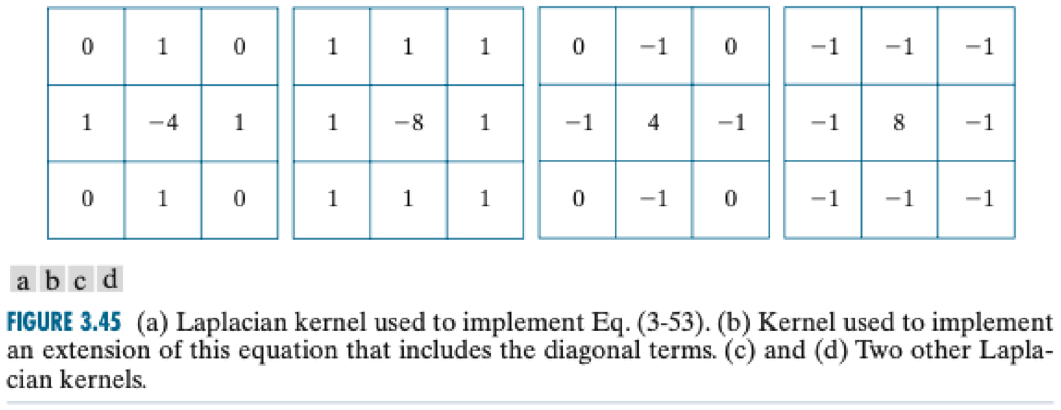
Fig.1: Sobel filter

To brighten the dark areas of an image, the power-law (gamma) transformation

$$s = cr^{\gamma}$$

is used to stretch out the dark intensity values.

To sharpen the image, the Laplacian filter is convolved with the image intensity array.



0	1	0
1	−4	1
0	1	0

1	1	1
1	−8	1
1	1	1

0	−1	0
−1	4	−1
0	−1	0

−1	−1	−1
−1	8	−1
−1	−1	−1

a b c d

FIGURE 3.45 (a) Laplacian kernel used to implement Eq. (3-53). (b) Kernel used to implement an extension of this equation that includes the diagonal terms. (c) and (d) Two other Laplacian kernels.

Fig.2: Laplacian filter

Hardware-Software Delineation

1. **User** copies the DICOM image file onto HPS memory.
2. **HPS** then parses the file, extracts the pixel dataset containing the Hounsfield Unit values. and transfers the pixel values to the FPGA.
3. **FPGA** compresses the image to a normal image intensity array, and sends data as an array to SDRAM storage as detailed below. Once the image is stored, the FPGA will retrieve the array line by line, holding 4 lines at a time in an M10K buffer. The FPGA will complete the filter processing shown above, depending on which algorithm the user selected (defaulting to Sobel), and send the filtered image array to the VGA monitor which displays the processed image output.

Data Compression

DICOM data is stored as "Hounsfield Units" which range from -1024 to 3071 HU, which contains 4096 possible values. Since $2^{12} = 4096$, the pixel bits can be truncated from 16 bits to 12 bits by removing the top 4 most significant bits. The resulting image storage size would be approximately 393 KB.

$$(512\, pixel\, rows \times 512\, pixels\, columns \times 12\, bits\, /\, pixel) \div 8\, bits/byte = 393,216\, bytes\, /image$$

Data Storage in SDRAM

An **SDRAM Controller** will send the image data to the SDRAM. The SDRAM controller, generated using the QSys system integration tool, will interface between the Avalon bus and the SDRAM chip. When the FPGA issues memory read or write requests, the controller translates these requests into the appropriate SDRAM control and data signals.

The diagrams below (from Altera Corporation's site) illustrate the role of the SDRAM controller within the system architecture and the specific signals used to send data to the SDRAM chip.

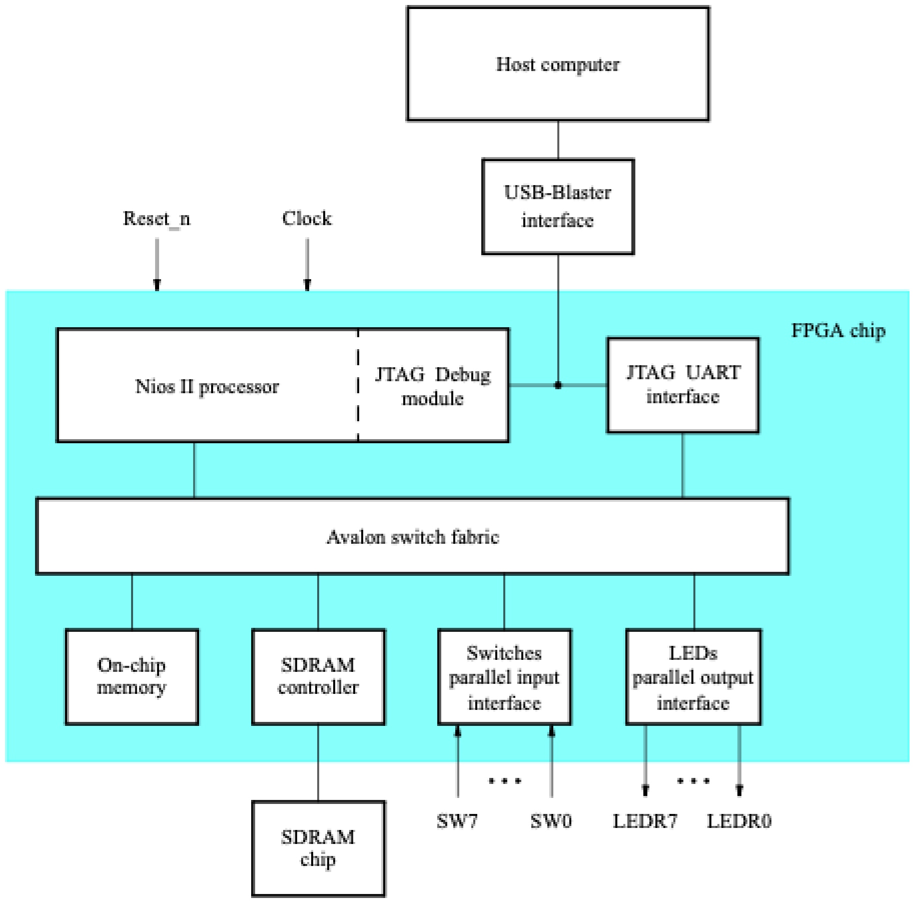


Fig.3: SDRAM controller architecture

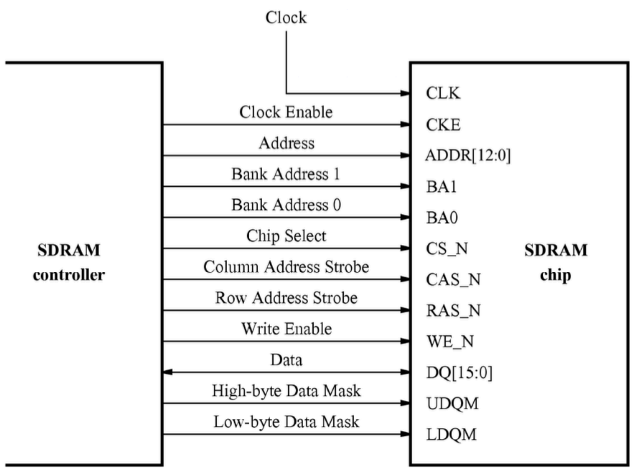


Fig.4: SDRAM signals

Milestones

Create a block diagram of the entire system detailing memory and timing for each component as the data flows through the data path	27 March
Create SDRAM Controller - This is the most challenging part of this project, so we will address this first. We will simulate a data array of random numbers to test it with, write out a FSM to plan proper behavior, and then using Qsys set up the SDRAM	15 April
Implement the image processing algorithms (Sobel edge detection, Laplacian sharpening and gamma correction) in Verilog	24 April
Compare FPGA output with MATLAB implementations of the same algorithms to verify results.	1 May
Implement filter selection, zooming and panning functionality	8 May

Reference

Gonzalez, R. C., & Woods, R. E. (2018). *Digital image processing* (4th ed.). Pearson.