Full Proposal

Image Enhancement Package

ECE 337

Team Member: Chichen Fu
Chang Liu
He Li
Yang Wang

Teaching Assistant: Chuan Tan

Lab Section: Thursday, 11:30 – 2:20 PM

Date: 10/19/2013

Signature: Chichen Fu
Chang Liu
He Li
Yang Wang

1. Executive Summary

Our team intends to design a image enhancement package (chip) which will process 24-bit bitmap image files with four filtering or editing options: brightness, contrast, grayscale, and B&W (black and white) adjustment. The 24-bit data will be stored in a SRAM and our datapath and its controller will read the data from the SRAM, then passed onto the ALU which all the filtering options take place. When the ALU finishes processing, the processed data will then be loaded back to the SRAM. The stored data can be possibly used for various purposes. It does contain certain degree of market value in the area of camera surveillance. Say a user needs to monitor a certain area which does not have a direct or limited light source geographically. By our brightness adjustment, the user will be able to see what the current situation or status of either a room or outside area; certain details like the status of hazard products can be also revealed for higher brightness. This design can also be applied for certain medical purposes. Some captured image files from medical device's cameras needs to be processed in order for the doctors or technicians to read off it. With image enhancements, it will be easier to process the data if the chip has already been integrated with the camera. The benefit of developing this chip can also been seen in terms of time and money. Time will be saved because company does not need to speed time on finding software solutions to do post-processing or analysis of the image file captured by the camera. When the chip is implemented with the cameras, the image can be processed almost instantaneously.

Our design differs from other design; the reason is that this image processing chip combines four functions into one big bundle. The users would choose what function he or she is willing to utilize. The four functions of this design intend to fit different user needs. This becomes crucial because most users nowadays want to pick a chip which is capable to perform more than one duty. Our chip falls into this category and customers are willing to purchase it to implement this chip in his or her system in order to do quick prototypes and demonstrations.

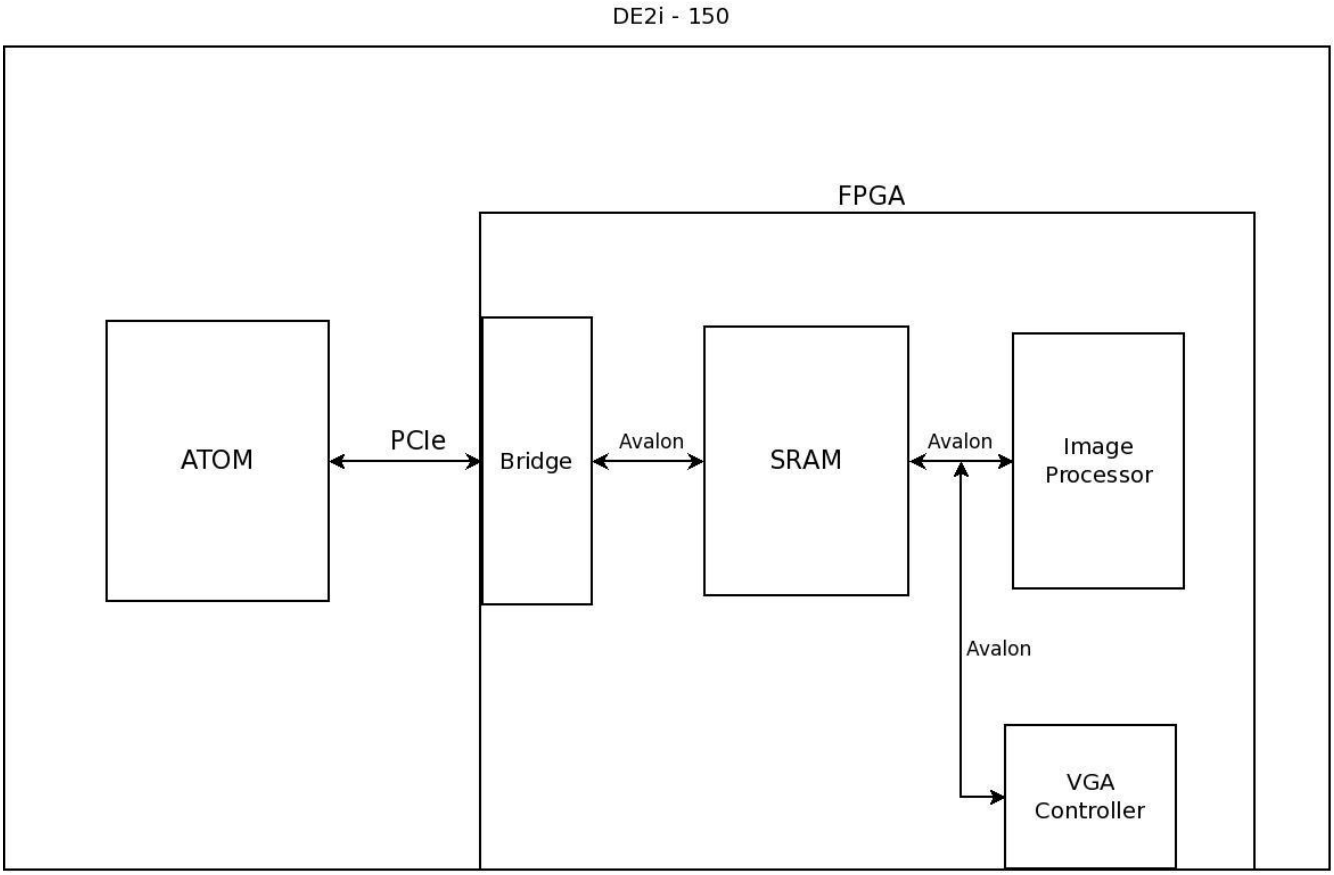
This design is appropriate for an ASIC implementation because first off, it requires quite low power consumption. It is always easier to put an low-power unit onto a existing device for either functional expansion or prototyping. Moreover, ASIC's high speed capabilities would satisfy our rather simple arithmetic operations. ASIC usually requires small area and therefore, it could be put into a camera device easily.

As described above, the 24-bit data will be first passed onto SRAM, then our top level controller, datapath incorporated with finite state machines and datapath controller will handle the overall flow of logic and arithmetic operations. After the data is processed, it will be finally written onto the SRAM again.

The rest of this proposal will guide you through our design specifications including the top level system usage diagram and operational characteristics. In this section, you will see further detailed input and output signal names and its descriptions. A list of functional blocks will also be provided with explanations. Requirements for the most critical design parameters will be discussed in this proposal as well.

2. Design Specifications

2.0 System Usage Diagram



The design is to enhance a 24 bit bitmap file with brightness, contrast, grayscale, and threshold operations. The commercial parts we will be using the DE2i-150 FPGA development kit. Our design will be programmed onto this FPGA. Our 24 bit image data will be first stored in the ATOM processor. Via PCI-express bus, the data will be sent to an intermediate bridge. Then, through Avalon bus, the data will be again sent to the SRAM. When our design, Image Processor Unit, sends out a signal which indicates the start of the processing. The unit will pull data from the SRAM and proceeds to make adjust with internal arithmetic and algorithm. When our chip has finished processing, it will store its data back to the SRAM via Avalon bus. The VGA controller will then pull the data from the SRAM and output the processed image data.

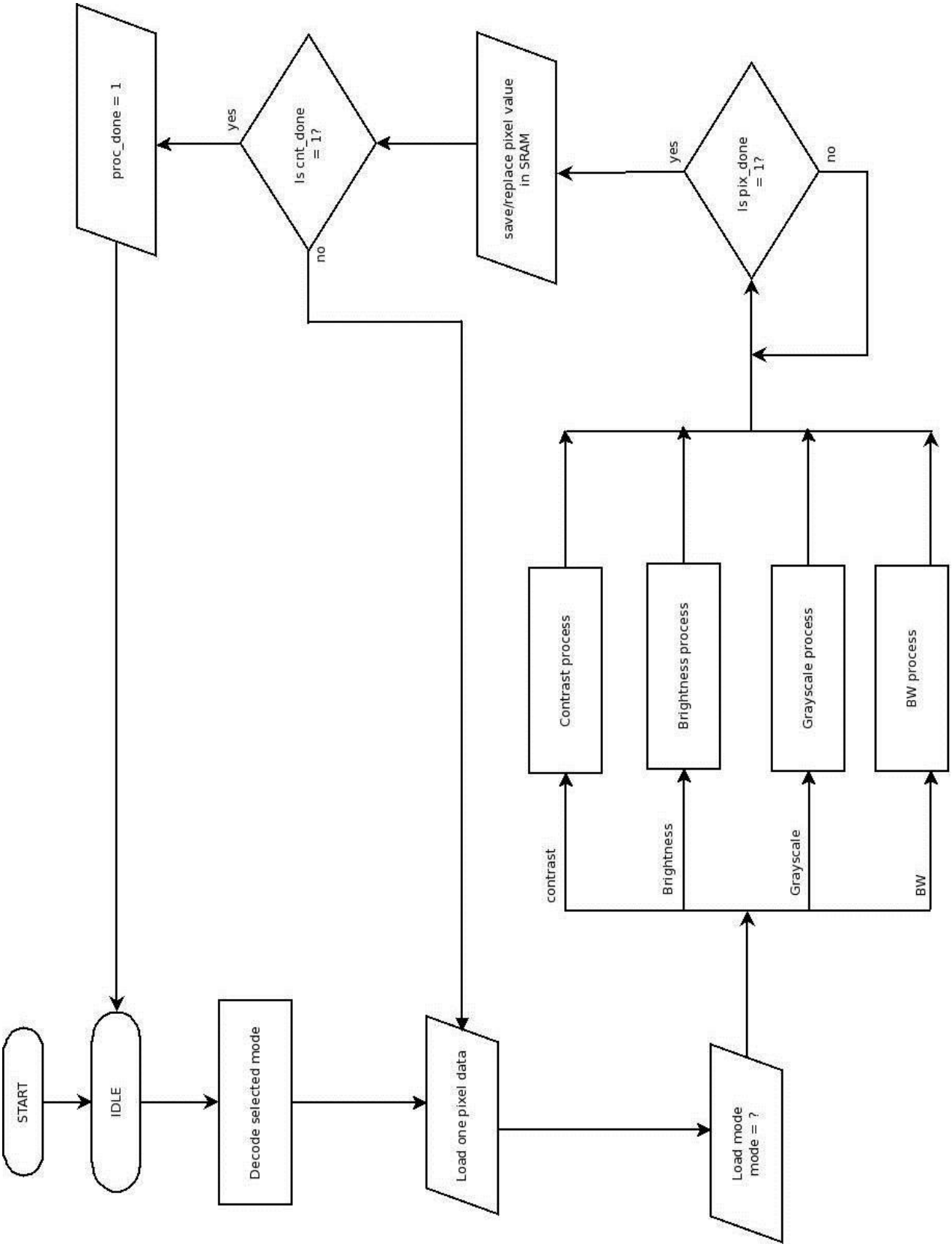
2.1 Operational Characteristics

Signal Name	Type In/Out/Bidir.	Number of bits	Description
clk	IN	1	200 MHz system clock signal with 50% duty cycle
n_rst	IN	1	Asynchronous active low triggered reset for all memory elements in the design
write_enable (Avalon)	IN	1	Write enable signal for both the SRAM and overall controller
read_enable (Avalon)	IN	1	Read enable signal for both the SRAM and overall controller
pixel_num	IN	16	Number of pixels specified by user's image file
pix_ready	IN	1	Pixel ready signal sending from the CPU simulator
mode_select	IN	2	The selection signal which enables the user to choose from brightness, contrast, grayscale and BW (black and white) operations (total of 4 operations)
threshold_value	IN	8	The specified threshold value which is utilized in the later arithmetic operations in ALU
add_sub_select	IN	1	Mode selection signal in between addition and subtraction operation
int_val	IN	8	The specified intensity control variable which is utilized in the later arithmetic operations in ALU for brightness and contrast adjustment
sample_data (Avalon)	BIDIR	32	The bidirectional data bus (Avalon) which carries the processed image data and communicates in between SRAM and processing unit (t will be concatenated with 8 extra zeros for the first 8 bits of data because our sample_data only occupies 24 bit of data)

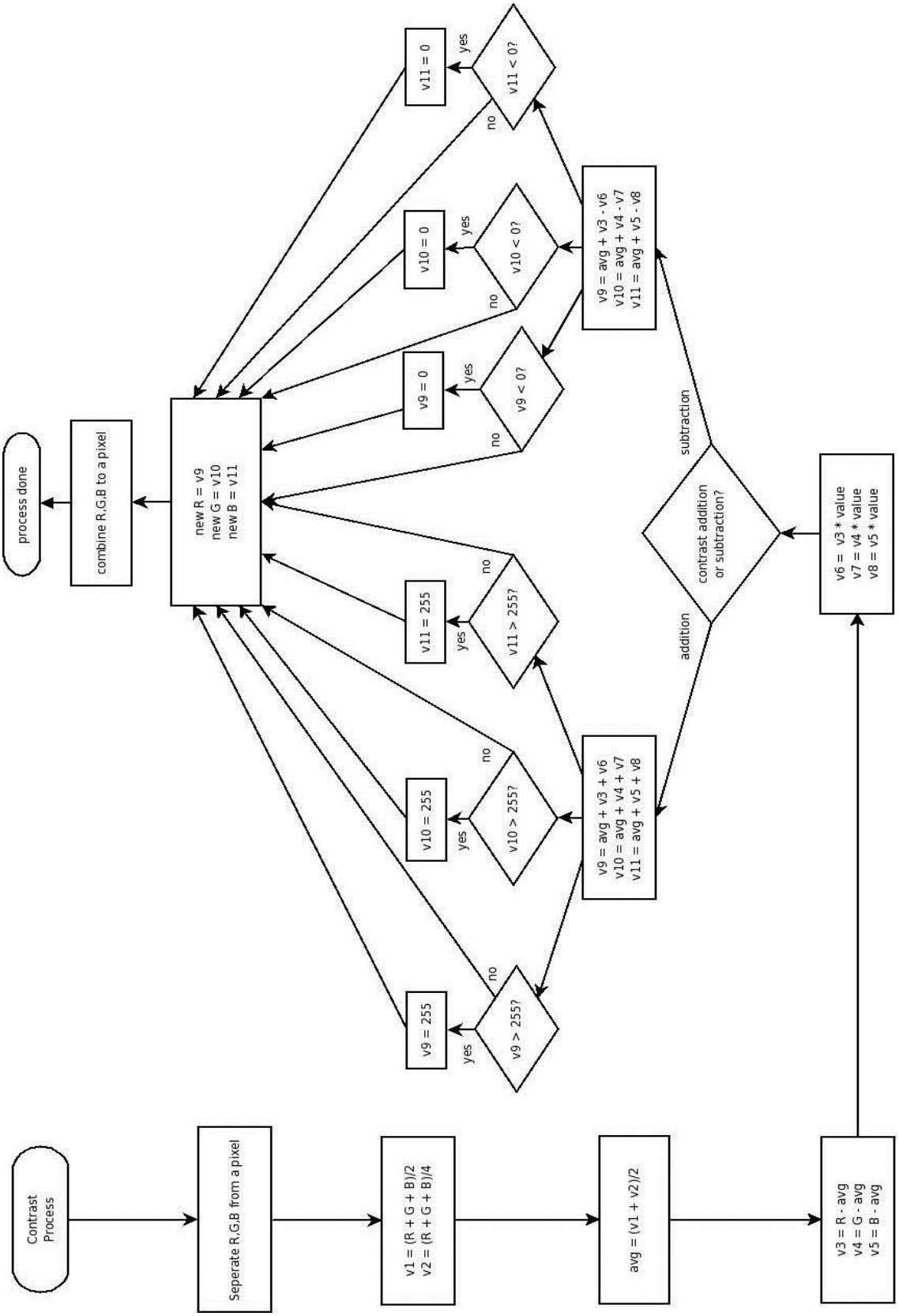
sram_addr (Avalon)	BIDIR	32	The SRAM data bus address
proc_done	OUT	1	The process done signal which indicates that the processing operation of all data has been finished
Error	OUT	1	The error sign which indicates that there is an overflow in the arithmetic operation

The commercial parts we will be using is the DE2i-150 FPGA development kit which utilizes the Avalon data transfer bus. The Avalon data transfer bus interconnects in between our chip and SRAM, and in between SRAM and bridge. The interface standard between ATOM and bridge will be using the PCI-express standard.

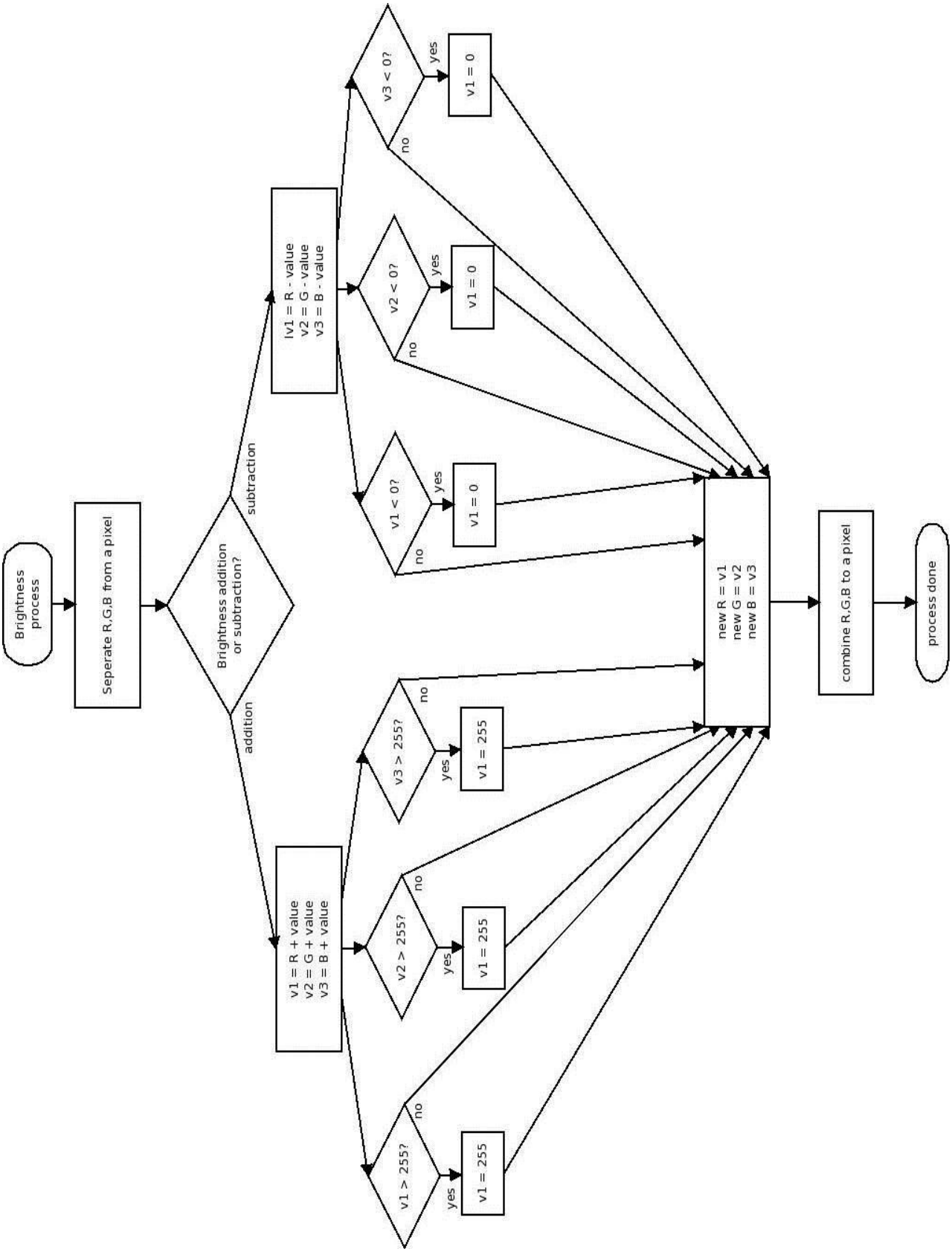
Flowchart of top level sequence of operations:



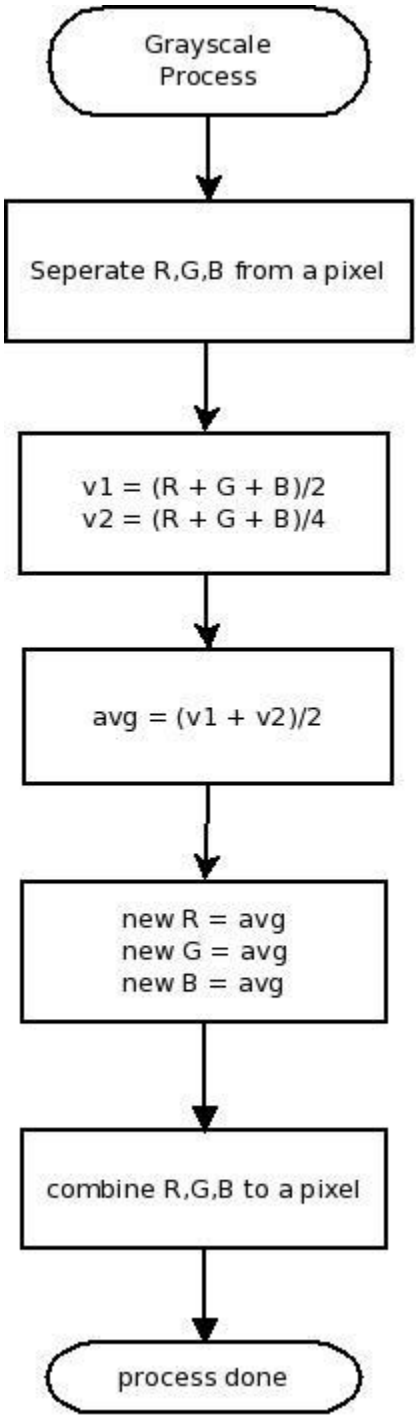
Flowchart of contrast control operations and algorithm:



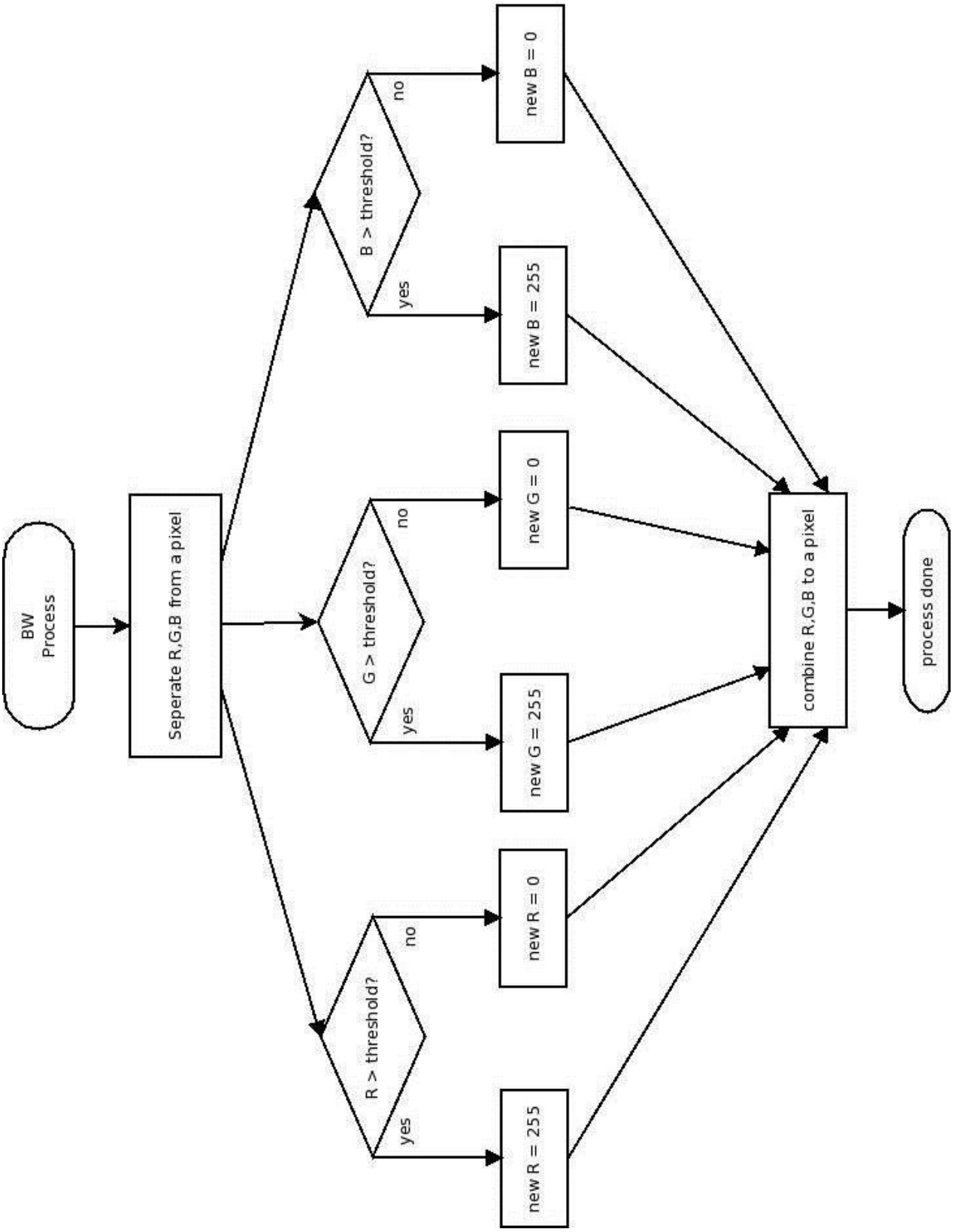
Flowchart of brightness control and algorithm:



Flowchart of grayscale control and algorithm:



Flowchart of Black and White Control and algorithm:



Functional Block Description:

- Top Level

This functional block has 4 sub-functions. It will load the 24 bit data from the SRAM one pixel a time. Then, with mode selection inputs, the image will be further processed by 4 operations. Four functions can be achieved by this image processor: contrast expansion, brightness manipulation, gray-scale, and threshold operation.

- Four sub-functions
  - Contrast Block

It can expand or reduce the contrast range of the image. The formula for this contrast algorithm is: R_out = avg + (R_in – avg) * (1 +/- int_val). “R_in” can be replaced with G_in and B_in and yields their correspondence outputs. “avg” is the average of RGB value of one pixel. “int_val” is the specified intensity adjustment level. Therefore, the dark pixels become darker and the bright pixels become brighter.

  - Brightness Block

It can increase or decrease the brightness of the BMP image by adding or subtracting a input value to all the RGB values of each pixel respectively.

  - Gray-scale Block

It can convert a color image to gray-scale image by averaging its R,G,B values of each pixels.

  - B&W Block

It can convert the image to a pure black and white image by an input threshold. All the R,G,B values greater than the threshold become black and all those smaller than the threshold become white.

2.2 Requirements

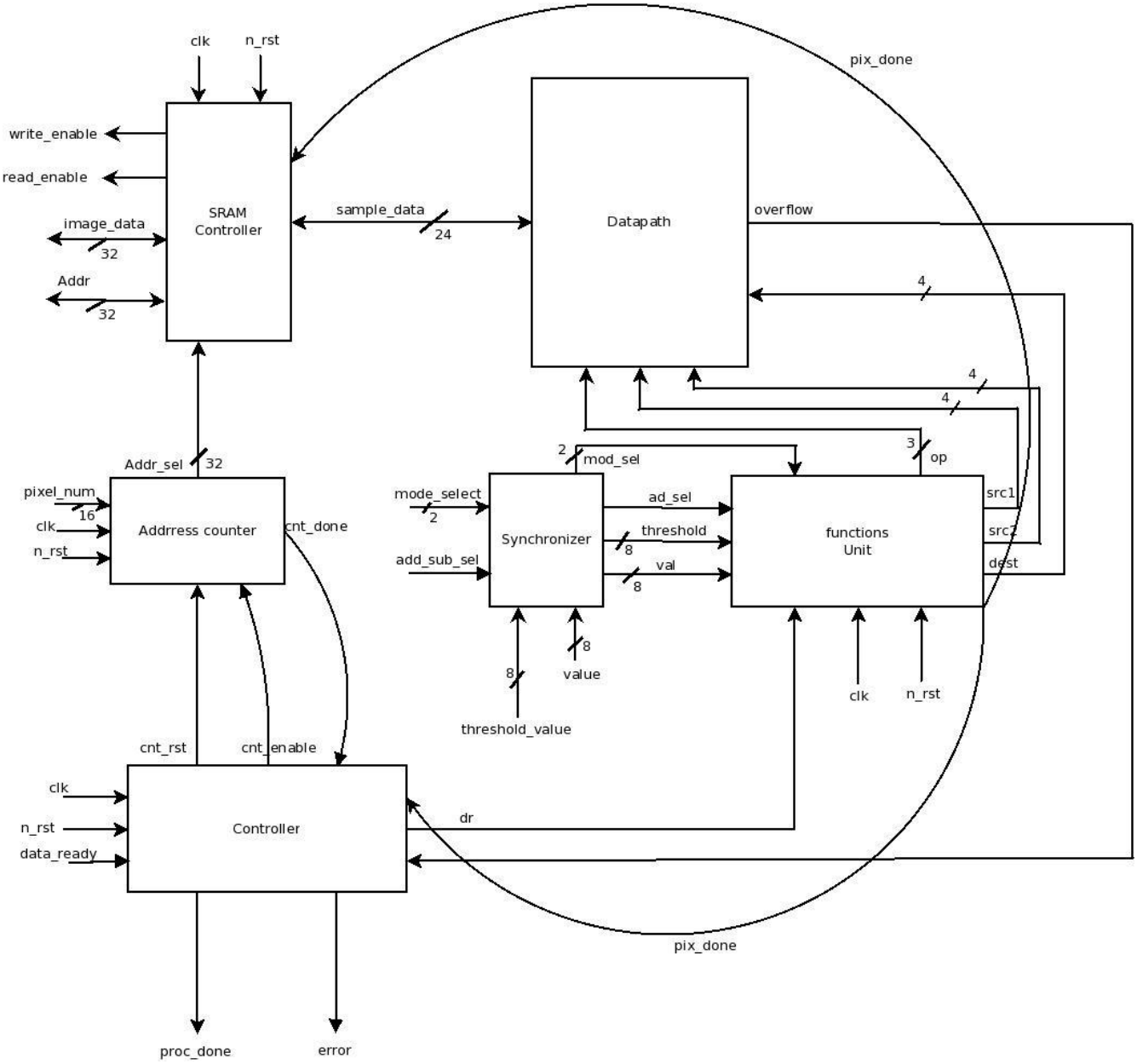
The main optimization objective of our design to is to maximize the processing speed of the incoming image data. This chip will be mainly used in a device in which a image processing unit is needed. Therefore, it needs a rather low power dissipation so that the device will not need extraneous power to have this chip running. Since our design incorporated the idea of using an SRAM, our chip allocation size might be a concern. This issue is significantly important if the consumer wants to put our design into a tiny medical device because it might not fit the overall design of such small-scale device. In addition, the speed cap of our design would also become problematic if the user intends to put this implementation into a high speed oriented camera. The reason is that it is highly possible that our speed capabilities cannot reach the actual processing requirement of a high speed camera. Therefore, our requirement for our design to fit is a platform, (can be either camera or any other image processing unit), which has some area tolerance as well as average speed specifications; it can mainly be implemented in a low-powered medical image devices or a simple camera which is in demand of a processing unit.

As stated above, our optimization objective is to maximize the speed. Since our design needs to process a good chunk of data, it is better for us to figure out the best way to apply our logic flow as well as the arithmetic operations. One of our strategies is to handle our data in parallel. For instance, when our design loads in the 24 bit data, our chip will further separate them into three 8 bit data lines representing the R, G, B data. Then, the ALU can handle 3 lines of data in parallel instead of handling them in series which is tedious and slow. In this method, the area will be increased for a little bit but won't tradeoff too much area.

Our design does require a slightly large area because of the area occupied by the SRAM. However, we don't think it would exceed the allocation of 1.5mm x 1.5 mm. Pin counts will be 106 with correspondence to each input and output of our design.

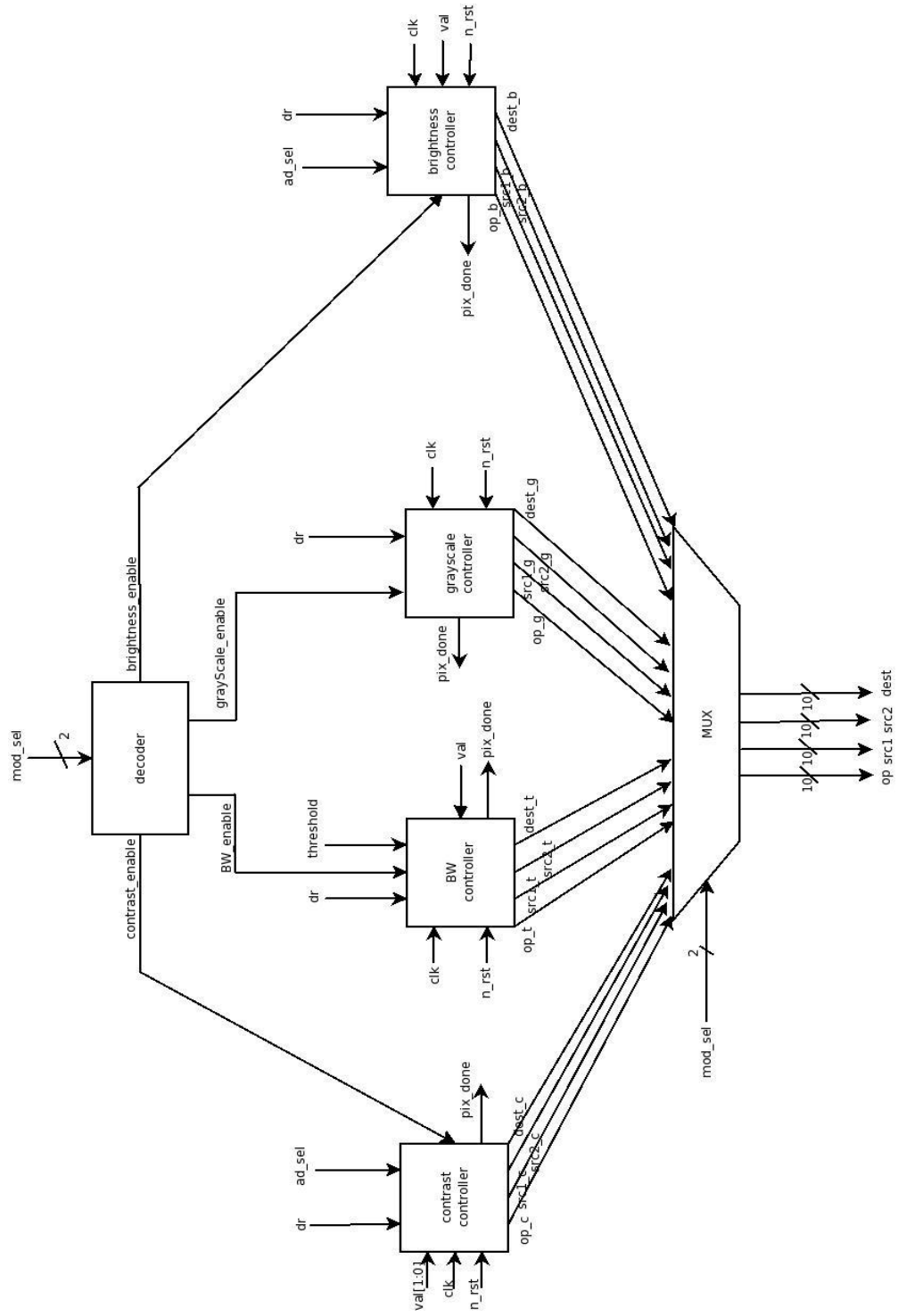
3. Initial Design

3.1Design Architecture

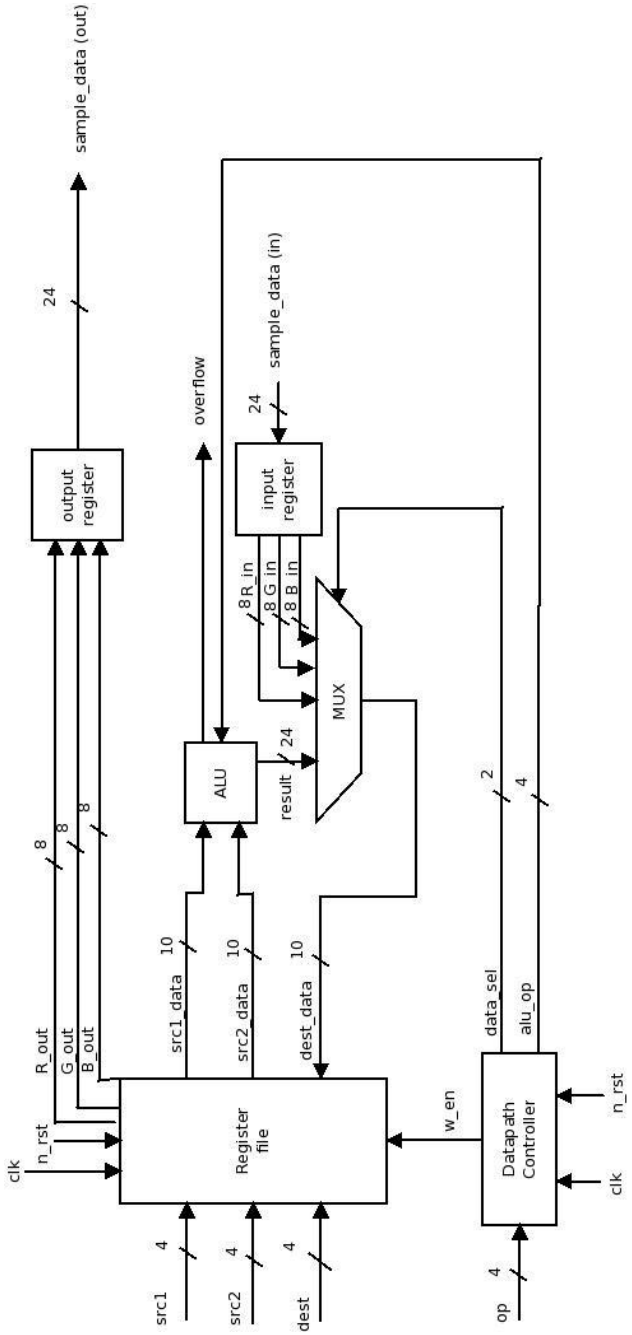


3.2 Functional Block Diagrams

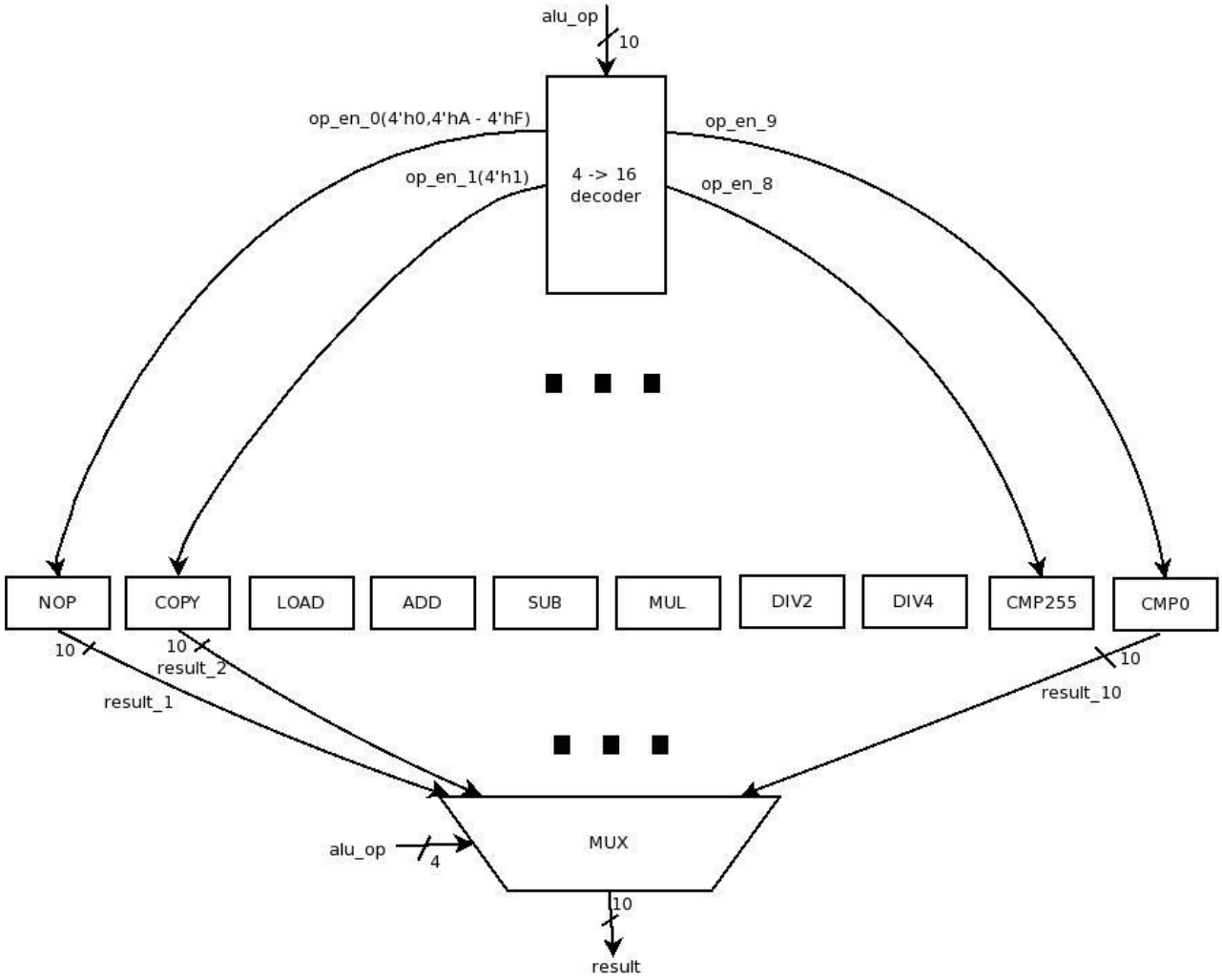
Functions Unit Block: the input mod_sel will be decoded and then enable the desired operation. After one operation is selected, one of the four controllers will output two operands, one op code, and one destination address.



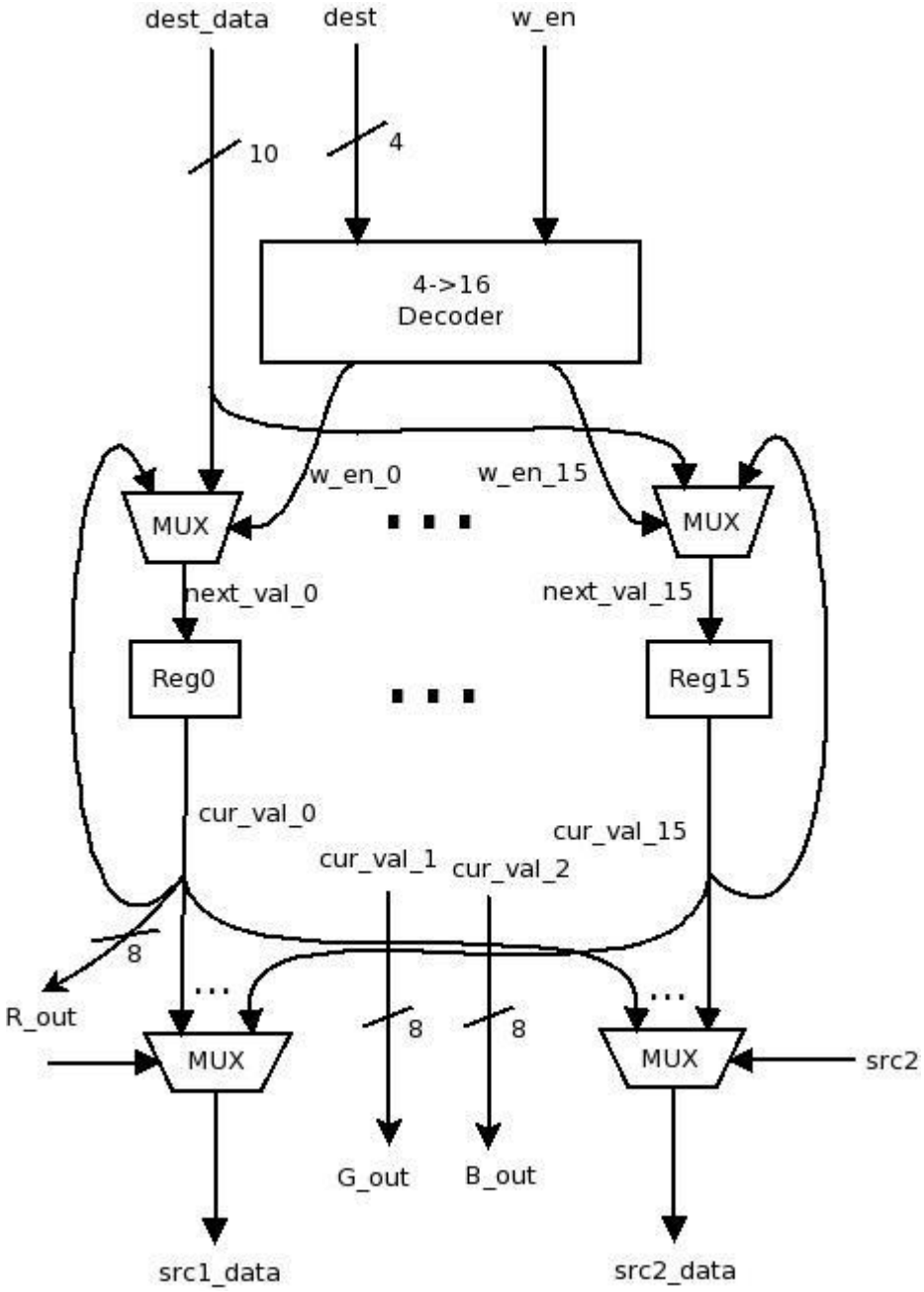
Datapath Block: the three operands from sample data and one op code is taken in as inputs. They are then processed in the ALU and stored in the register file. After the whole process is finished, register file will output the RGB data onto the bidirectional sample data bus.



ALU Block: the arithmetic logic unit which will handle all the arithmetic operations. For our design, there are a total of 10 operations in correspondence to 10 op codes.



Register File Block: the register file block will act as a “stack” and handle all the data storage.



4. Projected Timeline and Division of Tasks

Week	Overall Task	Duties
Week of Oct. 28Th	Figure out sub-functional modules like mux, decoder, etc. Figure out controller FSM. Design Budgeting.	Chang Liu: Figure out state transition diagram for controller Chichen Fu: Figure out state transition diagram for controller He Li: Figure out state transition diagram for controller Yang Wang: Figure out state transition diagram for controller Also, all team members need figure out and start coding mux, decoder, etc.
Week of Nov. 4Th	Figure out and code ALU, main controller, and functions unit. Testbench writing.	Chang Liu: Figure out and start coding ALU + some of the functions unit Chichen Fu: Figure out and start coding ALU + some of the functions unit He Li: Testbench writing Yang Wang: Start coding the main controller and some of the functions unit
Week of Nov. 11Th	Figure out and code datapath. Testbench writing.	Chang Liu: Coding and testing datapath Chichen Fu: Coding and testing datapath He Li: Testbench writing Yang Wang: Coding and testing datapath
Week of Nov. 18Th	Testing and debugging.	Chang Liu: Testing and debugging Chichen Fu: Testing and debugging He Li: Testbench writing Yang Wang: Testing and debugging
Week of Nov. 25Th	Testing and debugging. Final report and presentation.	Chang Liu: Testing and debugging Chichen Fu:Testing and debugging He Li: Testing and debugging Yang Wang: Testing and debugging

5. Success Criteria

5.1 Fixed criteria:

1. (2 points) Test benches exist for all top level components and the entire design. The test benches for the entire design can be demonstrated or documented to cover all of the functional requirements given in the design specific success criteria.
2. (4 points) Entire design synthesizes completely, without any inferred latches, timing arcs, and, sensitivity list warnings.
3. (2 points) Source and mapped version of the complete design behave the same for all test cases. The mapped version simulates without timing errors except at time zero.
4. (2 points) A complete IC layout is produced that passes all geometry and connectivity checks.
5. (2 points) The entire design complies with targets for area, pin count, throughput (if applicable), and clock rate. The final targets for these parameters will be determined by course staff based on your design review.

5.2 Design specific criteria:

1. (2 points) Demonstrate by simulation of VHDL test benches that the function of brightness control (+ and -) can be successfully perceived from the image output.
2. (2 points) Demonstrate by simulation of VHDL test benches that the function of contrast control (+ and -) can be successfully perceived from the image output.
3. (2 points) Demonstrate by simulation of VHDL test benches that the function of grays-scaling an image can be successfully perceived from the image output.
4. (2 points) Demonstrate by simulation of VHDL test benches that the function of B&W can be successfully perceived from the image output.