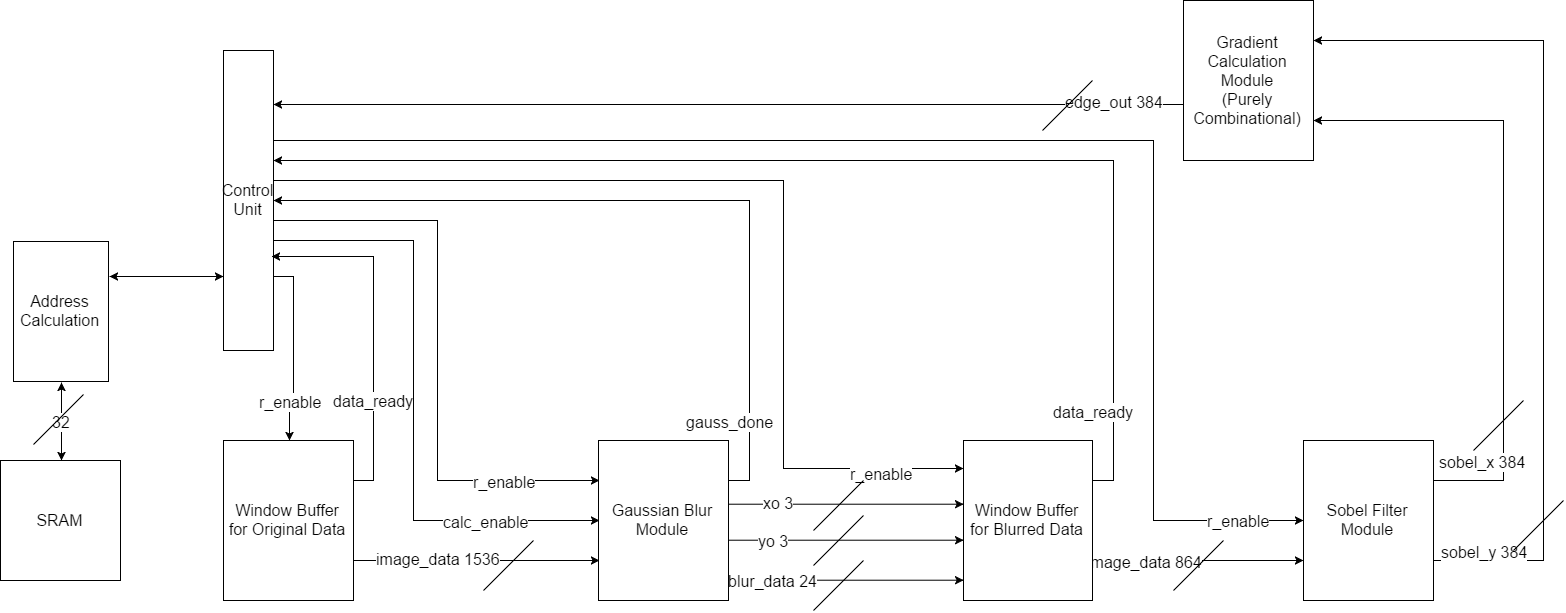
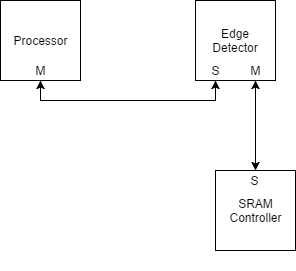
BMP Image Edge detection for Arm

Processors Verification Plan

Team members: Haobo Chen, Ni Kang, Haoming Zhang

Architecture





AHB-Lite Bus interface Module:Top-Level block that encapsulates data sending and reception via the AHB-Lite Bus. 256-bit transfer mode is used for our design. There are totally two interface module because we are using 2 buses(See picture above). The first bus communicates the processor and edge detector and the second bus communicates the edge detector with the SRAM controller.

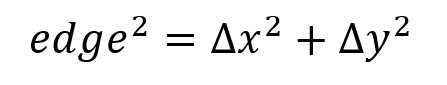
Address Calculation Block: Given the start address of the image in memory, the function of address calculation block is to create a correspondence between the index(offset from the start address) and the actual address in memory.

Window Buffer Block: The window buffer will be utilized to store the original and blurred data. But the implementation is identical. First the dir\_sel flag will be used to decide the behavior of the shifting operation. If dir\_sel is 11, it means that this is the first time the module gets executed. Otherwise, it indicate the direction of shifting operation. If r\_enable is asserted, the image data will be read in serially and be stored in the empty parts of the shifted buffer. A combination of new data and old data will form the next buffer\_data to be calculated. The done\_flag is asserted only when the internal counter reaches it’s rollover flag.

Gaussian Blur Block:The Gaussian Blur module will have two inputs from MCU: the ‘read\_enable’ signal to enable the counter (starts from 1) and the x\_i, y\_i values to determine the center pixel for calculation. The ‘calc\_enable’ signal is to get to know if the calculation can be processed. If it is set, the calculated value will be shifted into the next window buffer. After reading the 3 \* 3 pixel data from the previous buffer, a Gaussian Blur calculation will be implemented by adding the total 9 numbers together and then divide the sum by 9 to get the average pixel value. X\_o = x\_i -1; y\_o = y\_i - 1. These two determines the coordination on next window buffer. After calculation, the pixel will be transmitted to the next window buffer. And finally, when the x\_o or y\_o value reaches to the bottom-left index, return a gaussian\_done signal to MCU.

Sobel Filter Module:This is a purely combinational block, which will calculate two values for each pixel. The input of this module is a 6\*6 pixel array, and the output should be two arrays calculated by the middle 4\*4 pixels. One of the output arrays should be the horizontally calculated value of the 4 pixels, and the other one is the vertically calculated value of the 4 pixels. And the diagrams of the specific algorithm are shown above.

Gradient Calculation Module: After a group of gradients is found, the Pythagorean theorem is applied to the two gradient vectors. The calculated edge value is only necessary for comparing with the threshold, leave it as a squared value is acceptable.



If the edge value is larger than a threshold of our choice, it is recognized as a valid edge and will output FF (i.e. white pixel). Otherwise it will output 00 (i.e. black pixel).

By performing the operation above, an image with only major edges will be generated.

Fixed Criteria: (Total of 12 SC Points)

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. Failure to reach any of the targets will result a score of 1 out of 2 provided that you are within 50% on area, 10% on pin count, and 25% on throughput. Doing worse in any category will result in a score of 0 out of 2.

Design Specific Success Criteria: (Total of 8 SC Points)

1. (1 points) Demonstrate by simulation of a Verilog test bench that gaussian block works produce the correct blurred pixel given a 3x3 matrix directly from the window buffer.
2. (2 points) Demonstrate by simulation of a Verilog test bench that sobel filter works produce the correct 4x4 matrix of xy sets given a 6x6 blurred pixel matrix from the window buffer.
3. (1 points) Demonstrate by simulation of a Verilog test bench that gradient calculator works produce the correct squared valued based on the input xy set.
4. (2 points) Demonstrate by simulation of a Verilog test bench that the overall function produce the correct grayscale image with all the edges found.
5. (2 points) Demonstrate by simulation of a Verilog test bench that the address calculation block calculate the correct address so that data w/r can be implemented correctly. And it should make sure that the window buffer stores the correct 8\*8 pixels value.

Verification Plan Summary

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| What to Verify | Design Module(s) Involved | Verification Procedure Summary | DSSC(s) Proved | Use in Final Demo | Comments |
| Correctness of Window Buffer | Top-Level | Test if the window buffer can store the image data correctly each time it shifts in | DSSC1 | Yes | Use temporary registers to store the 8\*8 or 6\*6 pixels and test for corner cases in image |
| Correctness of Gaussian Blur | Top-Level | Initialize the counter and see if it can correctly blur the image and store them to the next window buffer | DSSC2 | Yes | Just check the output of the index and the calculated value |
| Correctness of Sobel filter | Top-Level | Give a 6\*6 pixel to the module and see if it can filter the middle 4\*4 pixels correctly | DSSC3 | Yes | Check the calculated values and check its correctness |
| Correctness of Gradient calculation | Top-Level | Give a 4\*4 pixel to the module and see if it can perform the calculation correctly and correspondingly generate a threshold value for it | DSSC4 | Yes | Check the calculated values and check its correctness and set an proper threshold |
| Correctness of address calculation | Top-Level | See if it corresponds to the correct memory address | DSSC6 | Yes | Check the correspondence |
| Shift register | Window Buffer | Test if it can shift the data correctly | DSSC1 | Only if  can’t show  using top  level | Use the shift register test bench that we created during lab. |
| counter | Window Buffer,  Gaussian Blur Block | Test if it can count and reset correctly | DSSC1&2 | Only if  can’t show  using top  level | Use the flex\_counter test bench that we created during labs |
| Sobel calculation block | Sobel filter Block | Give a 6\*6 pixel to the module and see if it can filter the middle 4\*4 pixels correctly | DSSC3 | Only if  can’t show  using top  level | Should be able to reuse most of test bench  For sobel Filter |
| Gaussian calculation block | Gaussian Blur Block | Give a 3\*3 pixel to the module and see if it can blur the middle pixel correctly | DSSC2 | Only if  can’t show  using top  level | Extract part of the test bench from the gaussian blur module and only test for the calculation part |
| Gradient calculation block | Gradient block | Give a 4\*4 pixel to the module and see if it can perform the calculation correctly and correspondingly generate a threshold value for it | DSSC4 | Only if  can’t show  using top  level | Should be able to reuse most of test bench  For gradient calculation |

Detailed Verification Test Breakouts

Demo Test

Correctness of Overall Chip Response

* Shown in Demo: Yes
* DSSC(s) Proved: 5
* Highest Level of Design Modules(s) involved:
  + Total Design/Chip
* Test bench Expectations/Requirements.
  + Load image data from the system
* No external or premade references are needed
* Clean image data need to be extracted from bmp file
* Main Verification Test Steps:
  + Load image data from the system and pass the clean image data into emulated RAM
  + Save image produced by edge detector to RAM
  + Save image from RAM to the system for further verification
  + Check that the produced edge is according to that of the given image

Correctness of Gaussian Blur

* Shown in Demo: Yes
* DSSC(s) Proved: 2
* Highest Level of Design Modules(s) involved:
  + Total Design/Chip
* Test bench Expectations/Requirements.
  + Load image from system
  + Feed clean image data and required signal into the module
  + Capture image from the module
  + Check required signals
* No external or premade references are needed
* No pre/post processing is needed
* Main Verification Test Steps:
  + Image data will be feed directly into the module
  + The image output will be captured and verified manually
  + Transaction signals will be checked by testbench

Correctness of Window Buffer

* Shown in Demo: Yes
* DSSC(s) Proved: 1
* Highest Level of Design Modules(s) involved:
  + Total Design/Chip
* Test bench Expectations/Requirements.
  + Load image from system
  + Feed clean image data and required signal into the module
  + Test the correctness of the data stored in the window buffer
  + Check required output signals
* No external or premade references are needed
* No pre/post processing is needed
* Main Verification Test Steps:
  + Image data will be feed directly into the module
  + The data stored in the window buffer will be tested by the test bench by comparing with the original image data
  + Transaction signals will be checked by testbench

Correctness of Sobel Filter

* Shown in Demo: Yes
* DSSC(s) Proved: 3
* Highest Level of Design Modules(s) involved:
  + Total Design/Chip
* Test bench Expectations/Requirements.
  + Load image from system
  + Feed clean image data and required signal into the module
  + Capture image from the module
  + Check required signals
* No external or premade references are needed
* No pre/post processing is needed
* Main Verification Test Steps:
  + Image data will be feed directly into the module
  + The image output will be captured and verified manually
  + Transaction signals will be checked by testbench

Correctness of Gradient Module

* Shown in Demo: Yes
* DSSC(s) Proved: 4
* Highest Level of Design Modules(s) involved:
  + Total Design/Chip
* Test bench Expectations/Requirements.
  + Feed xy data sets
  + Feed required signal into the module
  + Check output data
  + Check required signals
* No external or premade references are needed
* No pre/post processing is needed
* Main Verification Test Steps:
  + Test the sum of squared x and y with expected value

Backup and Sub-Module Tests

Correctness of Gaussian Blur

* Shown in Demo: Only if can’t show using top level
* DSSC(s) Proved: 2
* Highest Level of Design Modules(s) involved:
  + Gaussian Blur
* Test bench Expectations/Requirements.
  + Load image from system
  + Feed clean image data and required signal into the module
* No external or premade references are needed
* Clean image data need to be extracted from bmp file
* Main Verification Test Steps:
  + Image data will be feed directly into the module
  + The image output will be captured and verified manually
  + Transaction signals will be checked by testbench

Correctness of Window Buffer

* Shown in Demo: Only if can’t show using top level
* DSSC(s) Proved: 1
* Highest Level of Design Modules(s) involved:
  + Window Buffer
* Test bench Expectations/Requirements.
  + Feed 6x6 pixel matrix into the module
* No external or premade references are needed
* Clean image data need to be extracted from bmp file
* Main Verification Test Steps:
  + Image data will be feed directly into the module
  + The data stored in the window buffer will be tested by the test bench by comparing with the original image data

Correctness of Sobel Filter

* Shown in Demo: Yes
* DSSC(s) Proved: 3
* Highest Level of Design Modules(s) involved:
  + Sobel Filter
* Test bench Expectations/Requirements.
  + Feed 6x6 pixel matrix and required signal into the module
* No external or premade references are needed
* Clean image data need to be extracted from bmp file
* Main Verification Test Steps:
  + Image data will be feed directly into the module
  + The image output will be captured and verified manually

Correctness of Gradient Module

* Shown in Demo: Yes
* DSSC(s) Proved: 4
* Highest Level of Design Modules(s) involved:
  + Total Design/Chip
* Test bench Expectations/Requirements.
  + Feed xy data sets
  + Feed required signal into the module
* No external or premade references are needed
* No pre/post processing is needed
* Main Verification Test Steps:
  + Test the sum of squared x and y with expected value