**6 x 6 BMP Edge Detector**

**Phase 1 Proposal**

Submission Date: 2/11/2018

Thursday 14:30 Lab (Lab Section 11)

TA: Reena Elangovan

**Prepared by:**

Ni Kang

Haoming Zhang

Haobo Chen

**1. Executive Summary**

Edge detection is a basic problem for image processing and computer visualization, whose aim is to denote the points which change obviously in digital image. The obvious changes in image properties usually reflect the property’s important events and changes, which includes: discontinuity in depth, discontinuity on surface, the changes in matter properties and the brightness change in a specific scenario. Image edge detection drastically decreases the number of data and eliminates the irrelevant information which keeps the important properties of the image structure. There are a lot of ways to be used in edge detection, and they can be divided into two parts. The first part is to find the maximum and minimum value in the first derivation of the image. The second part is to find the edge through find the second derivation of the image. This result in a large amount of arithmetic and repeated processing, which, makes an ASIC design a perfect fit for the purpose. In addition, having a SoC design produced specifically for them function described above will drastically diminish the workload for processing images.

The design will utilize Intel Avalon Memory Mapping Interface(Avalon-MM), which is capable for both reading and writing. The Avalon-MM is required for communication between systems in the design. Since edge detection requires large scale computation, Avalon-MM itself won’t be able to store all necessary data. All data will be stored in registers in order for edge detection. The edge detector will also have two dedicated core, one for image filtering, the other is for edge detection.

The successful design of the proposed edge detector will require the following resources:

* Avalon Memory Mapped Interface Datasheet
* Reference Standard Cell Simulation Library for Mapped Design Verification
* Reference Standard Cell Technology Library for Final Design Layout Verification
* Verilog HDL Simulation and Design Synthesis Tool Chain

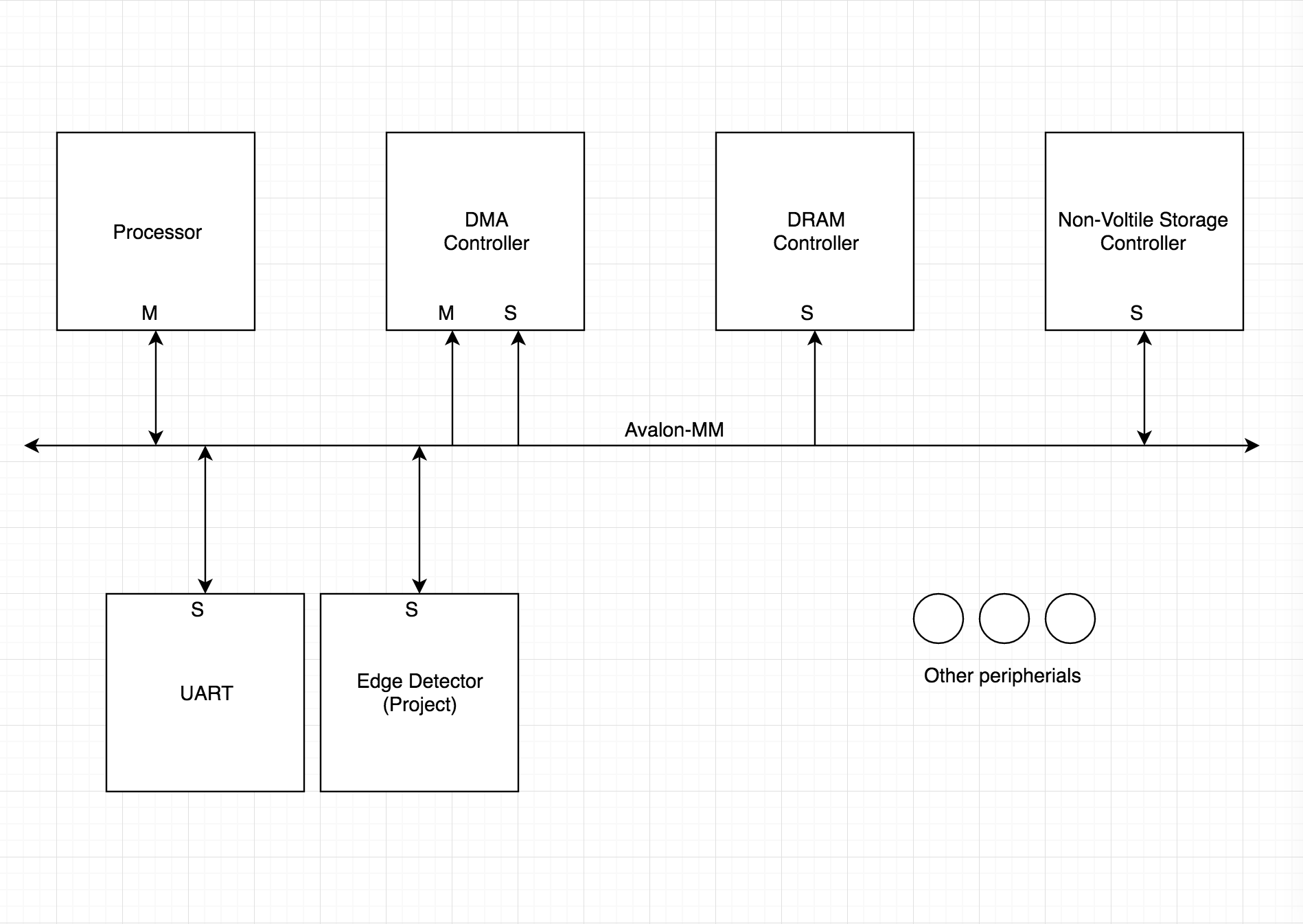
The following documents will describe:

* Super high level block diagram of edge detection
* Edge detector address mapping
* Design pinout
* Design architecture

**2. Design Specifications**

***2.1. System Usage***

***2.1.1. System Usage Diagram***

******

*Figure 1: Example System Usage Diagram for Edge Detector*

An example system illustrating the intended use of this edge detector is depicted above in Figure 1. In this

system there is a main processor where any relevant software would be executed, volatile main memory,

non-volatile storage, an UART module designed to serially input the image, our edge detector, and Direct Memory Access (DMA) controller for handling large data movement between slaves. The key operational ideas are that the software running on the processor would perform the following steps:

1. Configure the operational settings directly on the edge detector

2. Schedule a DMA transfer to populate the filter values

3. Directly send the image data to the edge detector

4. Wait until edge detector’s completion status bit is asserted

5. Read the calculation result for use

***2.1.2. Implemented Standard(s) and Algorithm(s)***

**Edge Detector**

18-bit Integer Filter Values

Configurable for signed or unsigned value usage

Default configuration is signed

36 \* 3-byte Integer Result Magnitude values

Configurable for signed or unsigned value usage

Default configuration is signed

6\*6 bmp image file needs 6\*20 bytes

**Avalon Memory Mapped Interface (Avalon-MM) Standard Slave**

1392-bit data bus

Read and Write Transfers

Burst Transfers Supported

Pipelined Transfers

10-bit, 522 word-address namespace (0x000-> 0x304)

***2.1.3. Edge Detector Namespace Address Mapping***

*Table 1: Namespace Address Mapping Table*

|  |  |  |  |
| --- | --- | --- | --- |
| Slave Word | Address Read / Write | Data Size (Byte) | Description |
| 0x000 - 0x0FF | R/W | 3 | Input image locations |
| 0x100 - 0x10F | R/W | 2 | Input filter matrix data |
| 0x200 - 0x2FF | W | 3 | Result image location |
| 0x300 - 0x301 | R | 4 | 64 bit accumulator |
| 0x302 | R/W | 1 | Configuration Register 0: Number of samples for calculation |
| 0x303 | R | 1 | Status Register:  Bit 0: filter done (1 -> true, 0 -> false)  Bit 1: result done (1 -> true, 0 -> false) |
| 0x304 | R/W | 1 | Control Register:  Bit 0: clear prior sample (1 -> clear, 0 -> keep)  Bit 1: clear prior filter (1 -> clear, 0 -> keep) |

***2.2. Design Pinout***

*Table 2: Miscellaneous Pinout Table*

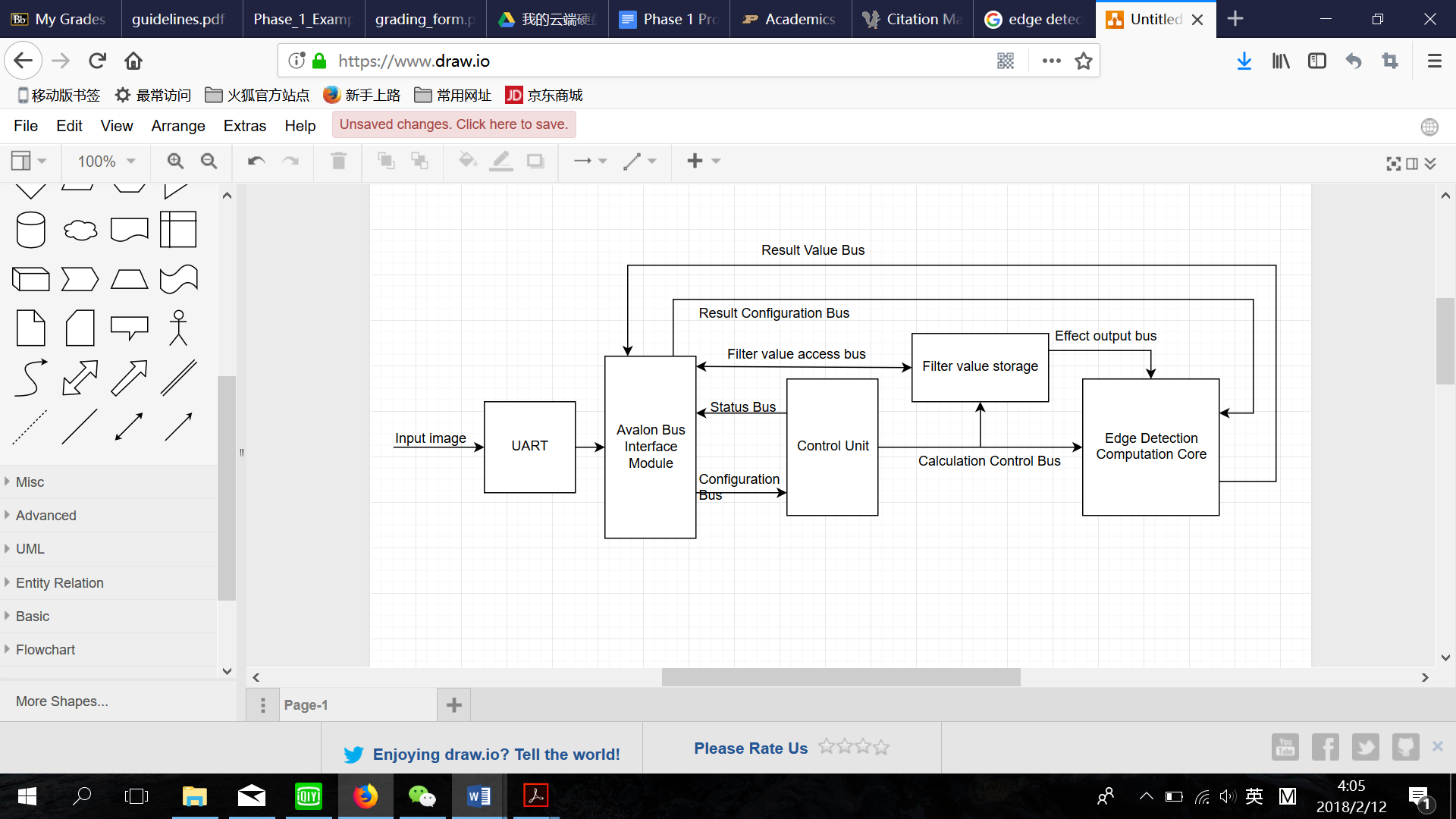
|  |  |  |  |
| --- | --- | --- | --- |
| Signal Name | Direction | Number of bits | Description |
| vcc | PWR | / | Power Pin |
| gnd | GND | / | Ground Pin |
| clk | IN | 1 | System clock(100MHz) |
| n\_rst | IN | 1 | Asynchronous Reset. (Active Low) |

*Table 3: Avalon Memory Mapped Slave Interface Pins*

|  |  |  |  |
| --- | --- | --- | --- |
| Signal Name | Direction | Number of bits | Description |
| header\_detected | IN | 1 | Input image header read. If the input image is less than 3x3 or larger than 6x6, set it to 0. Else, set it to 1. |
| image\_data | IN | 20\*6\*8 | Store the full 6\*6 image data |
| filter | IN | 18 | Sobel filter value |
| process\_done | OUT | 1 | Set to high when process finished |
| image\_out | OUT | 20\*6\*8 | Output image data |
| data\_ready | IN | 1 | Set to high when data is ready to be processed |

**3. Design Implementation**

***3.1. Design Architecture***

******

*Figure 2: Edge Detection Architecture Diagram*

The intended implementation architecture is depicted above in Figure 2. A module handling the

implementation of the Avalon Memory Mapped slave interface is used to bridge between the

external SoC environment and the internal data and status locations. There will be a UART module which is used to serially input the input image data. There will be one value storage module, which is for the filter value set. It will be implemented using the underlying stream register design that will be defined in subsequent documents. A edge detection computation core module will house all of the arithmetic related portion of the design for processing a stage of the calculation. These two components (filter storage, and computation core) will be stepped through the full algorithmic edge detection calculation for new image data by the main control unit, which will be implemented as a Finite-State-Machine (FSM) with subordinate counters for tracking the progression through the needed stages.

**Reference Cited:**

“Popular Links:” *Intel PSG*, 08-May-2017. [Online]. Available: https://www.altera.com/documentation/nik1412467993397.html. [Accessed: 11-Feb-2018].