DIGITAL SYSTEM DESIGN FINAL PROJECT REPORT
DEPARTMENT OF ELECTRICAL ENGINEERING
UNIVERSITAS INDONESIA

2D GRAPHICS ACCELERATOR

PERRY ANIMATE

GROUP 21

PERRY TJAHYA	**2406409965**
MUHAMMAD DHIYA'ULHAQ	**2406356782**
VINCENZO FABIAN TISILA	**2406354215**
SUTAN RENDY RIZALDI	**2406434626**

PREFACE

Laporan Proyek Akhir ini disusun sebagai pemenuhan persyaratan praktikum Desain Sistem Digital. Proyek ini berfokus pada perancangan dan implementasi akselerator grafis 2D, yang kami sebut sebagai 2D Graphics Accelerator, menggunakan perangkat keras deskriptif Very High-Speed Integrated Circuit Hardware Description Language (VHDL).

Dalam komputasi modern, pemrosesan grafis dasar sering kali membebani Central Processing Unit (CPU). Tujuan utama dari proyek ini adalah untuk merancang Graphics Processing Unit (GPU) yang didedikasikan untuk mengambil alih tugas-tugas rendering geometris dasar, seperti garis, lingkaran, persegi panjang, dan segitiga. Dengan demikian, beban komputasi dapat dialihkan dari CPU, yang pada akhirnya meningkatkan efisiensi dan performa sistem secara keseluruhan.

Kami mengucapkan terima kasih kepada semua pihak yang telah memberikan bimbingan dan dukungan selama proses perancangan dan implementasi proyek ini.

Depok, December 7, 2025

Group 21

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND

Pada komputasi modern kita sering terkendala untuk pemrosesan grafis ketika mengandalkan CPU General Purpose. Arsitektur pada CPU lebih berfokus untuk logika dan eksekusi sekuensial, sehingga akan mengakibatkan inefisiensi ketika memproses volume data pixel yang besar secara berulang. inefisiensi ini akan membebani CPU utama, yang seharusnya berfokus pada logika aplikasi sehingga dapat menyebabkan penurunan performa sistem secara keseluruhan.

2D Graphics Accelerator Perry Animate kita jadikan sebagai ide dengan tujuan untuk merancang sebuah Graphics Processing Unit (GPU) yang dapat mengambil alih terkait tugas-tugas komputasi grafis dasar seperti, rectangle, circle line, dan triangle untuk membebaskan CPU dari beban rendering pixel.

1.2 PROJECT DESCRIPTION

2D Graphics Accelerator Perry Animate ini merupakan implementasi dari arsitektur Graphics Processing Unit (GPU) akselerator dua dimensi yang dirancang dari awal. Deskripsi singkatnya merujuk pada: Implementasi GPU Microprogrammed pada Simulasi Vivado. Ini berarti bahwa inti dari sistem grafis ini dibangun dengan VHDL dan menggunakan mekanisme kontrol berbasis memori (microprogramming) untuk mengelola set instruksi grafisnya, dan seluruh proses verifikasi fungsionalitasnya dilakukan dalam simulasi Behavioral Xilinx Vivado.

1.3 OBJECTIVES

The objectives of this project are as follows:

1. Merancang arsitektur GPU 2D menggunakan VHDL
2. Mengimplementasikan algoritma menggambar grafis (Rectangle, Triangle, Circle, Line) ke dalam logika digital.
3. Menerapkan arsitektur Microprogrammed Control Unit untuk fleksibilitas set instruksi

1.4 ROLES AND RESPONSIBILITIES

The roles and responsibilities assigned to the group members are as follows:

Roles	Responsibilities	Person
Ide	- Mencari ide	- Vincenzo Fabian Tisila
Implementasi	- Mencari bagaimana cara implementasinya	- Vincenzo Fabian Tisila - M. Dhiya - Perry Tjahya
GPU	- Menggabungkan semua komponen menjadi 1 komponen GPU.	- M. Dhiya
CU	- Membuat microprogrammed CU.	- M. Dhiya
Datapath	- Menggabungkan algoritma rendering menjadi 1 komponen datapath.	- Vincenzo Fabian Tisila
VRAM	- Membuat komponen VRAM.	- M. Dhiya
Algoritma rendering	- Membuat algoritma rendering segitiga dan garis. - Membuat algoritma rendering lingkaran dan persegi. - Membuat algoritma rendering text.	- Perry Tjahya - Vincenzo Fabian Tisila - M. Dhiya
Testbench	- Membuat testbench animasi gambar bentuk. - Membuat testbench animasi text.	- Sutan Rendy Rizaldi - M. Dhiya
PPT	- Membuat PPT presentasi	- M. Dhiya
Laporan	- Membuat laporan	- Sutan Rendy Rizaldi - M. Dhiya - Vincenzo Fabian Tisila
Readme github	- Membuat readme github	- Perry Tjahya

Table 1. Roles and Responsibilities

CHAPTER 2

IMPLEMENTATION

2.1 EQUIPMENT

The tools that are going to be used in this project are as follows:

- Platform Vivado
- Arsitektur Kontrol Microprogrammed (berbasis ROM)

2.2 IMPLEMENTATION

Implementasi akselerator grafis 2D Perry Animate ini dilakukan dalam lingkungan Simulasi Behavioral Xilinx Vivado menggunakan VHDL. Sistem ini dirancang untuk beroperasi pada resolusi kanvas spesifik 512 x 256 pixel dengan kedalaman warna 24-bit RGB (True Color). Input ke sistem berupa Instruksi Makro 64-bit Fixed Length yang merepresentasikan perintah grafis dari Testbench (CPU). Hasil akhir pemrosesan disimpan ke VRAM sebelum dikeluarkan sebagai File Bitmap (.bmp) melalui Testbench.

Terdapat tiga komponen utama dari arstitektur ni yaitu Testbench, GPU Core, dan VRAM. GPU Core dibagi menjadi Control Unit dan Datapath Unit. Datapath Unit menampung mesin-mesin khusus (engine) untuk menggambar bentuk dasar seperti Rectangle Engine, Triangle Engine, Circle Engine, dan Line Engine. Untuk mengendalikan operasi ini kita memanfaatkan arsitektur Microprogrammed Control Unit.

Arsitektur ini menggantikan logika Hardwired FSM yang kaku dengan tabel memori (Control Store) yang disimpan dalam Microcode ROM. Proses kerjanya melibatkan Mapper yang menerjemahkan Opcode instruksi menjadi alamat awal di ROM, dan Sequencer yang menentukan alamat memori berikutnya (Next Address Logic) untuk mengeksekusi pola sinyal control.

GPU menggunakan format instruksi "fix" 64-bit, di mana bit [63:60] dialokasikan untuk Opcode dan bit [59:00] untuk Payload (Koordinat/Warna/Ukuran). Sistem ini mendukung tujuh instruksi dasar, yaitu Finish/Print (0000), Set Color (0001), Clear (0010), dan instruksi menggambar seperti Draw Rectangle (0011), Draw Triangle (0100), Draw Line (0101), dan Draw Circle (0110). Semua hasil penggambaran disimpan dalam VRAM, yang secara konseptual adalah array.

CHAPTER 3

TESTING AND ANALYSIS

3.1 TESTING

Pengujian fungsionalitas akselerator grafis 2D dilakukan sepenuhnya dalam lingkungan Simulasi Behavioral Xilinx Vivado menggunakan VHDL Testbenches (tb_gpu.vhd dan tb_test_char.vhd). Proses pengujian meliputi:

1. **Pemberian Instruksi:** Testbench berfungsi sebagai pengganti CPU, mengirimkan serangkaian Macro Instruction 64-bit yang merepresentasikan perintah grafis (Set Color, Clear, Draw Rectangle, Draw Line, Draw Circle, Draw Triangle, Draw Char) ke unit GPU.
2. **Verifikasi Fungsional:** Setelah GPU menyelesaikan eksekusi perintah (ditandai dengan sinyal done), Testbench akan mengaktifkan "Dump Mode". Dalam mode ini, GPU menghentikan operasi menggambar dan mengizinkan Testbench untuk membaca seluruh konten Video RAM (VRAM) secara berurutan.
3. **Visualisasi Hasil:** Data VRAM yang telah dibaca kemudian diekspor oleh Testbench menjadi berkas gambar berformat Bitmap (.bmp). Berkas .bmp ini digunakan untuk verifikasi visual guna memastikan bahwa setiap perintah menggambar telah dieksekusi dengan benar dan menghasilkan output grafis yang akurat sesuai spesifikasi (misalnya, letak koordinat, pilihan warna, serta akurasi algoritma Bresenham untuk garis dan lingkaran).

3.2 RESULT

Hasil pengujian menunjukkan bahwa 2D Graphics Accelerator Perry Animate berhasil merender semua primitif grafis yang dirancang. Verifikasi visual dari berkas .bmp dan animasi hasil simulasi (seperti yang ditunjukkan pada Gambar 2) mengonfirmasi:

- **Keberhasilan Rendering Primitif:** Semua mesin penggambaran (Line Engine, Circle Engine, Rectangle Engine, Triangle Engine, Char Engine) berfungsi sesuai dengan algoritma yang diimplementasikan (misalnya, Algoritma Bresenham untuk garis, Bounding Box & Edge Function untuk segitiga).
- **Sistem Kontrol Mikroprogram:** Microcoded Control Unit berhasil menginterpretasikan Opcode instruksi dan mengelola handshaking (sinyal start, busy, done) serta wait states untuk menyinkronkan operasi Datapath Unit.

- **Dukungan Warna:** Sistem berhasil mengaplikasikan warna 24-bit (True Color) pada setiap piksel yang digambar dan berhasil menjalankan instruksi CLEAR dan SET_COLOR.

Fig 2. Testing Result

3.3 ANALYSIS

Analisis terhadap implementasi proyek menyoroti dua aspek utama, yaitu arsitektur kontrol dan efisiensi algoritma.

a. **Efisiensi Arsitektur Kontrol:** Penerapan Microprogrammed Control Unit terbukti jauh lebih fleksibel dan mudah diperluas dibandingkan dengan implementasi Hardwired Finite State Machine (FSM) yang kaku. Dengan memisahkan logika kontrol menjadi tabel memori (Control Store) dalam Microcode ROM, penambahan

instruksi baru (Opcode baru) di masa depan hanya memerlukan pembaruan microcode tanpa perlu mendesain ulang logika perangkat keras inti.

b. **Optimasi Pemrosesan Grafis:** Penggunaan algoritma yang dioptimalkan untuk perangkat keras, seperti Algoritma Bresenham untuk garis dan lingkaran, memungkinkan pemrosesan grafis dilakukan hanya dengan aritmatika bilangan bulat (penjumlahan, pengurangan, pergeseran bit). Hal ini secara signifikan menghindari operasi floating-point yang mahal secara komputasi dan memastikan kinerja tinggi yang efisien. Dengan mengimplementasikan mesin-mesin khusus ini dalam Datapath Unit, GPU berhasil mengambil alih beban pemrosesan piksel secara massal dari CPU.

CHAPTER 4

CONCLUSION

Proyek 2D Graphics Accelerator telah berhasil merancang dan mengimplementasikan arsitektur Graphics Processing Unit (GPU) 2D menggunakan VHDL dan Simulasi Behavioral Xilinx Vivado.

Kesimpulan utama dari proyek ini adalah:

1. Semua objektif yang ditetapkan, yaitu perancangan arsitektur GPU 2D, implementasi algoritma menggambar grafis dasar (Rectangle, Triangle, Circle, Line), dan penerapan arsitektur Microprogrammed Control Unit, telah berhasil dipenuhi.
2. Arsitektur Microprogrammed Control Unit memberikan fleksibilitas dalam set instruksi dan potensi pengembangan di masa depan. Algoritma grafis berbasis integer yang diimplementasikan dalam Datapath Unit berhasil mengakselerasi tugas rendering 2D.

Saran untuk pengembangan lebih lanjut:

1. Meningkatkan resolusi kanvas dan kedalaman VRAM untuk mendukung canvas yang lebih besar.
2. Menambahkan engine penggambaran yang lebih kompleks, seperti kurva Bézier atau texture mapping dasar.
3. Mengembangkan antarmuka fisik (misalnya, ke papan FPGA) untuk verifikasi kinerja pada perangkat keras yang sebenarnya.

REFERENCES

[1]

Care4you, “Microprogrammed Control Unit Computer Organization and Architecture - Care4you,” *Care4you*, May 19, 2021. https://care4you.in/microprogrammed-control-unit/ (accessed Dec. 07, 2025).

[2]

J. J. Jensen, “BMP file bitmap image read using TEXTIO,” *VHDLwhiz*, Nov. 13, 2019. https://vhdlwhiz.com/read-bmp-file/

[3]

GeeksforGeeks, “Bresenham’s Line Generation Algorithm,” *GeeksforGeeks*, Feb. 16, 2017. https://www.geeksforgeeks.org/dsa/bresenhams-line-generation-algorithm/

[4]

GeeksforGeeks, “Bresenham’s circle drawing algorithm,” *GeeksforGeeks*, Jul. 14, 2017. https://www.geeksforgeeks.org/c/bresenhams-circle-drawing-algorithm/

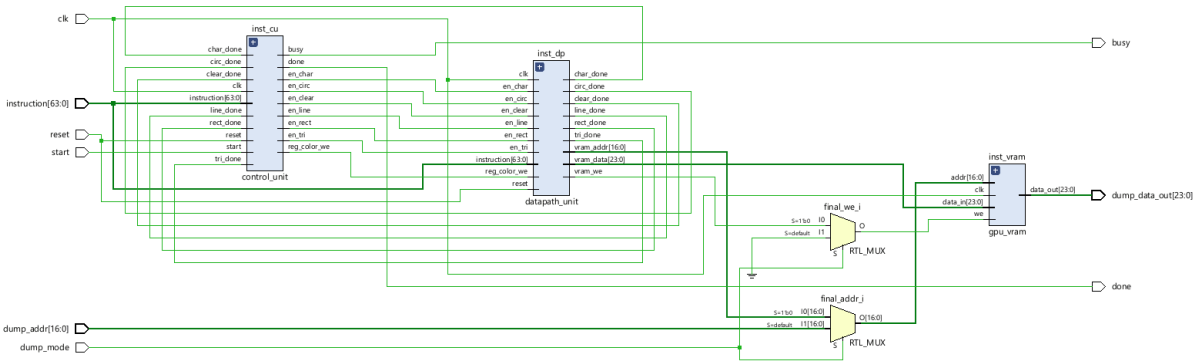
[5]

K. Fatahalian and D. James, “Drawing a Triangle (+ the basics of sampling and anti-aliasing),” 2022. Accessed: Dec. 07, 2025. [Online]. Available: https://gfxcourses.stanford.edu/cs248/winter22content/media/drawtriangle/02_drawtriangle.pdf

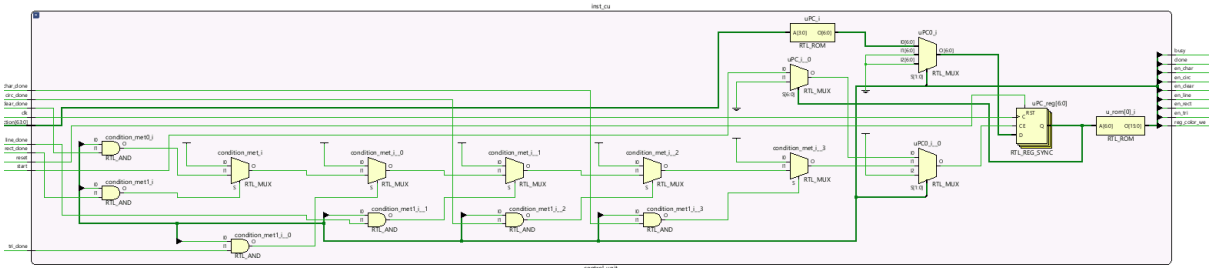
APPENDICES

Appendix A: Project Schematic

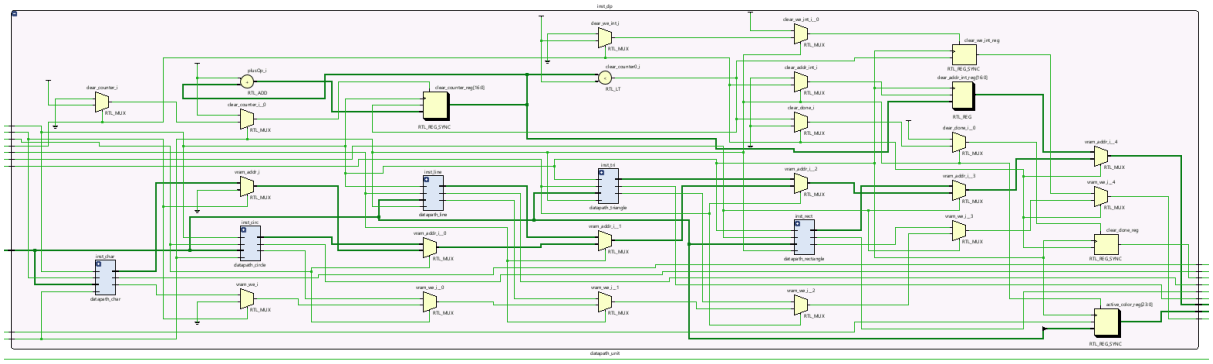
GPU



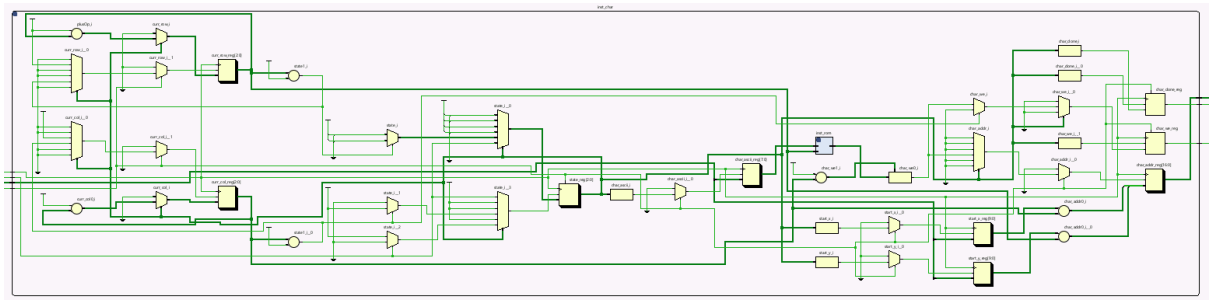
CU



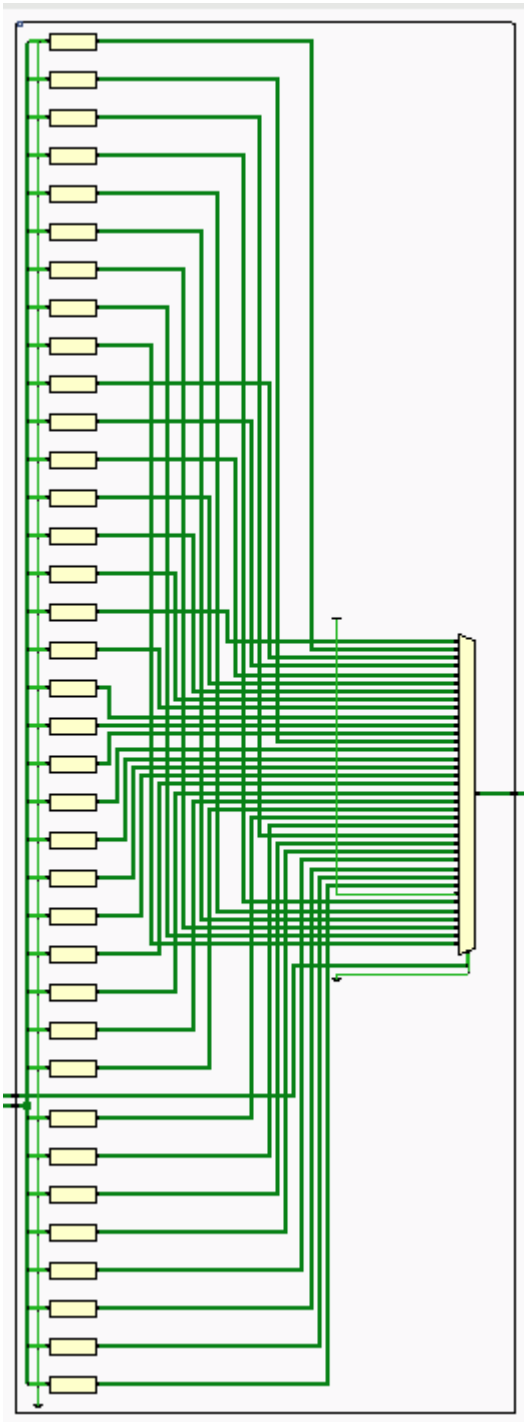
Datapath



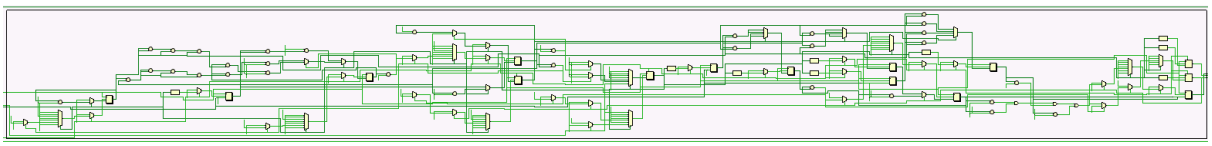
Character



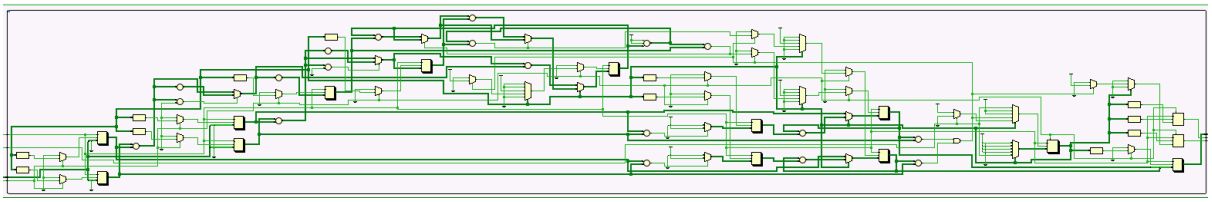
Character ROM



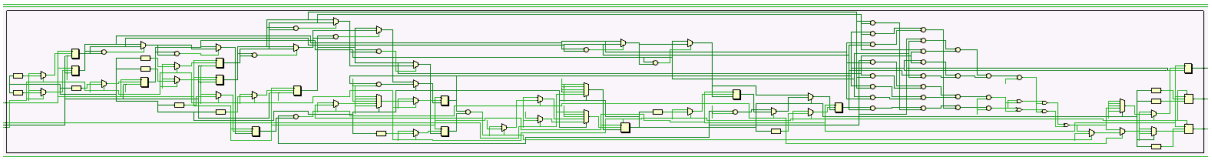
Circle



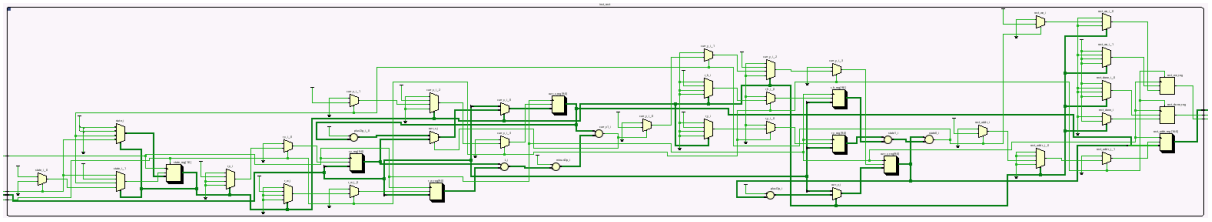
Line



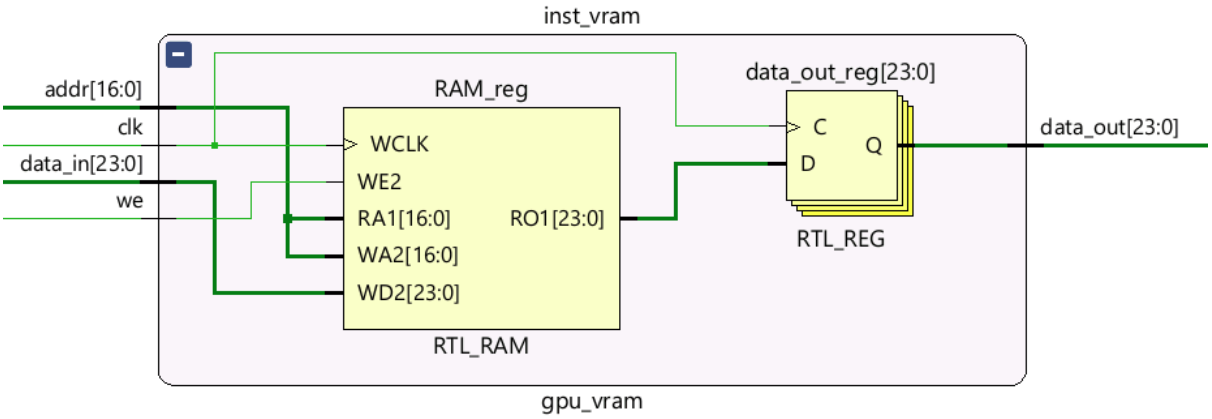
Triangle



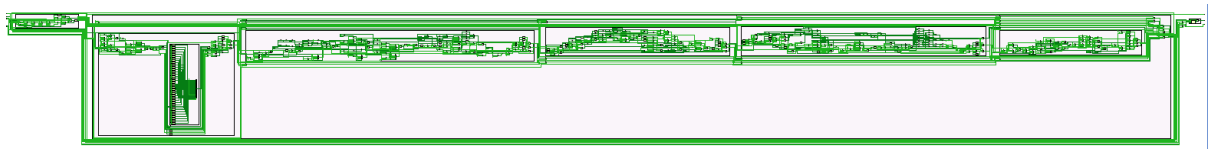
Rectangle



GPU VRAM



Full schematic



Appendix B: Documentation

Put the documentation (photos) during the making of the project