



FPGA System Specification

Revision 1 — Sørensen, Jonsterhaug, September 22, 2024

Contents

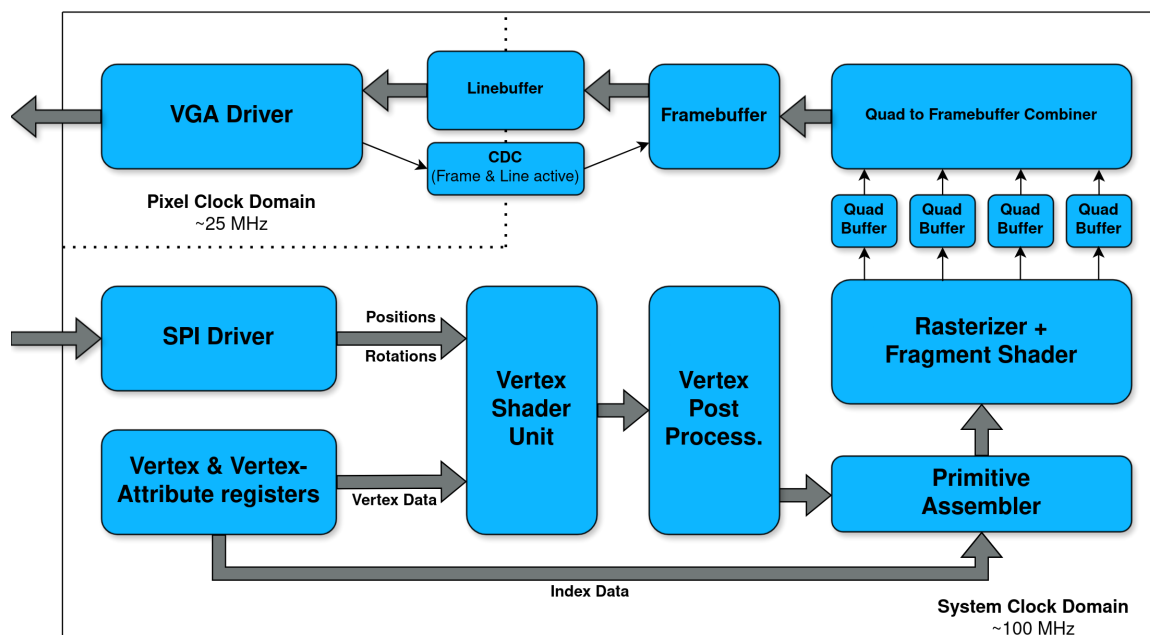
1	System Overview	2
2	FPGA-MCU Communication	3
2.1	SPI Driver	3
2.2	Data Protocol	3
3	Framebuffer	4
4	Display Driver	4
4.1	VGA Signals	4
5	Mesh and Attribute Storage	4
6	Render Pipeline	4
6.1	Vertex Shader	4
6.2	Vertex Post-Processor	4
6.3	Primitive Assembler	4
6.4	Rasterizer & Fragment Shader	4

1 System Overview

The system consists of four main parts:

1. The communications system between the FPGA and the MCU
2. The Framebuffer
3. The Display Driver
4. The Render Pipeline

The following is a diagram of the entire system, its parts, and the dataflow between modules.



The system has two main clock domains: the pixel clock domain for controlling the display at around 25 MHz, and the system clock domain for the rest of the system at around 100 MHz.

The render pipeline is the main part of the system. It's what does the graphics processing, taking in mesh data and outputting pixels.

2 FPGA-MCU Communication

2.1 SPI Driver

2.2 Data Protocol

The data protocol for the communication between the MCU and the FPGA is shown in figure 1.

[[FIGURE SHOWING PROTOCOL]]

Diagrams/data_protocol_diagram.png

Figure 1: Data protocol

First byte represents the number of entities that is to be rendered to the screen, called *NUM_ENTETIES*. The following 3 bytes represents the camera yaw and pitch angles in relation to the y and x axis respectively. The three bytes are split into two 12-bit fixed-point numbers of type Q1.11, where the MSB is the sign bit. This means that each angle represents a number in the range $[-1, 0.99951171875]$, which maps to the range $[-\pi, \pi]$.

After that follows *NUM_ENTETIES* accounts of entity data to be rendered, where the first entity is the player. The entity data encoding is as follows:

1. **Byte 1 – 8:** The 10 MSB bits are flags for each entity (TBD), then follows the x, y and z position of the entity, each of which are 18-bit fixed-point numbers on the form Q7.11.
2. **Byte 9 – 11:** Entity rotation in pitch, yaw, and roll, each represented with an 8-bit fixed-point number in a Q1.7 format (again MSB is sign).

The flag bits can be decoded as follows:

Diagrams/flag_bit_decoding.png

Figure 2: Flag bit decoding

3 Framebuffer

4 Display Driver

4.1 VGA Signals

5 Mesh and Attribute Storage

6 Render Pipeline

6.1 Vertex Shader

6.2 Vertex Post-Processor

6.3 Primitive Assembler

6.4 Rasterizer & Fragment Shader