Akcelerator graficzny fixed-pipeline oparty na FPGA

Jakub Janeczko

Instytut Informatyki, Uniwersytet Wrocławski

Promotor: Dr. Marek Materzok

6 lutego 2026

Cel projektu i problem

- Problem: sprzętowe przyspieszenie renderingu 3D w systemach wbudowanych.
- Cel: zaprojektować i zaimplementować fixed-pipeline GPU zgodny z wybranym podzbiorem OpenGL ES 1.1 Common-Lite.
- Założenia: arytmetyka stałoprzecinkowa, integracja z SoC, możliwość testowania i demonstracji.

Motywacja i kontekst

- FPGA łączy elastyczność z deterministycznym czasem działania.
- Fixed-pipeline pozwala precyzyjnie kontrolować zasoby i opóźnienia.
- Projekt jako punkt wyjścia do dalszych rozszerzeń (teksturowanie, shadery).

Zakres funkcjonalny

Zaimplementowane elementy potoku:

- Transformacje wierzchołków (model–view–projection).
- Rasteryzacja trójkątów z interpolacją perspektywiczną.
- Oświetlenie (ambient, diffuse) w wierzchołkach.
- Testy głębokości i szablonu.
- Mieszanie kolorów (alpha blending).

Technologie — dlaczego te?

- **FPGA Intel Cyclone V (DE1-SoC)**: elastyczność, możliwość iteracji architektury, integracja z HPS.
- **Amaranth HDL**: generatywność w Pythonie, szybka iteracja, symulacja i eksport do SystemVerilog.
- **OpenGL ES 1.1 Common-Lite**: klasyczny fixed-pipeline, naturalny punkt odniesienia i arytmetyka fixed-point.
- **SoC + Linux**: łatwy dostęp do CSR i uruchamianie aplikacji demonstracyjnych.

Wymagania i założenia

- Docelowa platforma: Intel Cyclone V (DE1-SoC).
- Podzbiór OpenGL ES 1.1 Common-Lite (fixed-pipeline).
- Arytmetyka stałoprzecinkowa (brak FPU w profilu).
- Integracja z SoC: CSR + dostęp do pamięci przez magistrale.
- Weryfikacja funkcjonalna w symulacji i na płytce.

Środowisko i narzędzia

- Symulacja i testy funkcjonalne na poziomie modułów.
- Integracja z Qsys/Platform Designer (Avalon-MM).
- Aplikacje demonstracyjne w userspace Linux.

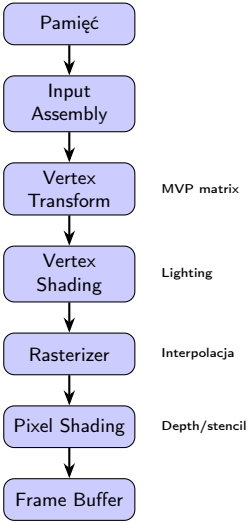
Architektura systemu (wysoki poziom)

- HPS (ARM) konfiguruje GPU przez rejestry CSR.
- GPU jako akcelerator w logice FPGA.
- Magistrale: Avalon-MM (CSR i dostęp do pamięci), mostek do Wishbone.
- Bufory: wierzchołki, indeksy, ramka, głębokość, szablon.

Architektura potoku graficznego

1. Wejście i składanie prymitywów (Input Assembly).
2. Transformacje wierzchołków (Vertex Transform).
3. Cieniowanie wierzchołków (Vertex Shading).
4. Rasteryzacja (barycentryczne, perspektywa).
5. Operacje fragmentów: depth/stencil/blend.

Diagram potoku graficznego



Zgodność z OpenGL ES 1.1 Common-Lite

- Zaimplementowane: transformacje, rasteryzacja trójkątów, oświetlenie ambient/diffuse, depth/stencil, blending.
- Braki: teksturowanie, specular, linie/punkty, MSAA.
- Skupienie na kluczowych elementach fixed-pipeline i stabilności działania.

Arytmetyka stałoprzecinkowa

- Brak FPU w profilu Common-Lite — konieczność fixed-point.
- Stabilność czasowa i przewidywalne zasoby w FPGA.
- Kluczowe miejsca: macierze, interpolacja perspektywiczna, testy głębokości.

Interfejs CSR i konfiguracja

- Rejestry sterujące: adresy buforów, formaty danych, topologia.
- Parametry macierzy, oświetlenia, viewport/scissor.
- Tryby testów depth/stencil i ustawienia blendingu.
- Dostęp z Linuxa przez mapowanie `/dev/mem`.

Wkład własny

- Projekt architektury fixed-pipeline i podział na moduły sprzętowe.
- Implementacja modułów Amaranth: transformacje, rasteryzacja, testy, blending.
- Zaprojektowanie mapy CSR i integracja z SoC.
- Środowisko testowe i aplikacje demonstracyjne.

Testowanie i weryfikacja

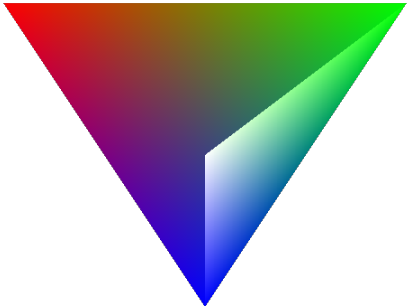
- Testy jednostkowe modułów potoku (symulacja).
- Porównanie wyników renderingu z referencją (obrazy PPM).
- Testy integracyjne na płycie DE1-SoC.

Wyniki — zasoby FPGA (Cyclone V)

- ALM: 31 372 / 32 070 (98%).
- DSP: 75 / 87 (86%).
- Pamięć: 634 097 / 4 065 280 bitów (16%).
- PLL: 3 / 6 (50%), DLL: 1 / 4 (25%).

Wyniki i demonstracja

- Poprawne renderowanie trójkątów z interpolacją i testami.
- Sceny demonstracyjne uruchamiane na płycie DE1-SoC.



Ograniczenia

- Brak teksturowania i specular.
- Brak rasteryzacji linii i punktów.
- Wysokie zużycie ALM (98%) ogranicza dalszą rozbudowę bez optymalizacji.

Wnioski

- Fixed-pipeline GPU na FPGA jest możliwe i edukacyjnie wartościowe.
- Amaranth HDL ułatwia iterację i testowanie architektury.
- Architektura modułowa ułatwia rozwój i izolację błędów.

Plany rozwoju

- Rozszerzenie funkcjonalności o teksturowanie, MSAA, shadery.
- Ulepszenia wydajności: większa równoległość, TBR.
- Lepsze narzędzia debug/telemetria w CSR.
- Zbliżenie do pełnej zgodności z OpenGL ES 1.1.

Źródła

- OpenGL ES 1.1 Specification (Khronos Group).
- Dokumentacja Amaranth HDL.
- Dokumentacja DE1-SoC i Cyclone V.

Demonstracja

Krótka demonstracja działania akceleratora na płycie DE1-SoC

Pytania

Dziękuję za uwagę