De Re PlayStation®Vita

- Brief Hardware Overview
- Key Points of Differentiation
  - Hardware as well as Software
- Development Hardware
- Highlight Simplified Software Libraries & T&Ms
  - Implementation Tech Demo videos
  - Example of use in game
- Graphics on PS®Vita (Neil Brown)

PlayStation®Vita

“Designed for the Ultimate Portable Gaming Experience”

Comparison to PSP-3000 screen: x4 higher resolution

x600 increased contrast

5 inch OLED screen: engineered for extremely low power consumption

PS VITA
PlayStation.Vita

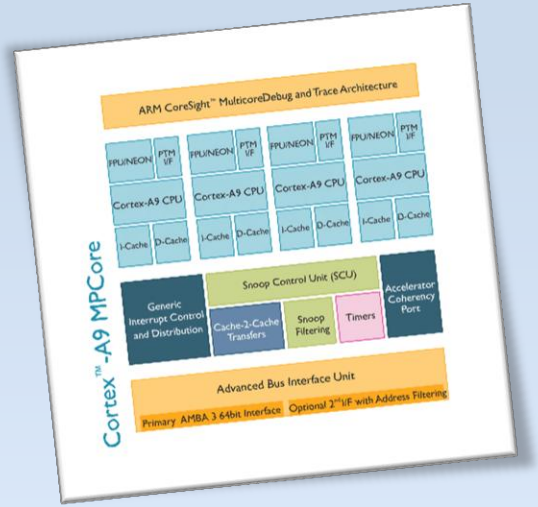
SCEE Research & Development

Hardware Overview

- 4x ARM® Cortex™A9 Processor
  - High performance 32-bit processors
    (3 available to applications)

- POWERVR SGX543MP4+ GPU
  - Power-efficient multi-core GPU

- Dedicated HW for Media playback



Hardware

- Standard Iconic PlayStation Buttons
- Dual Analog Sticks
- Front Touch Screen
- Rear Touch Pad
- Front & Rear Camera
- Motion Sensors

Software

- Social Networking
  - Designed from ground up as a social networking device
- Location Based Gaming
  - Location Services
  - near
- Covered in a dedicated talk:

  Building Community with PlayStation®Network

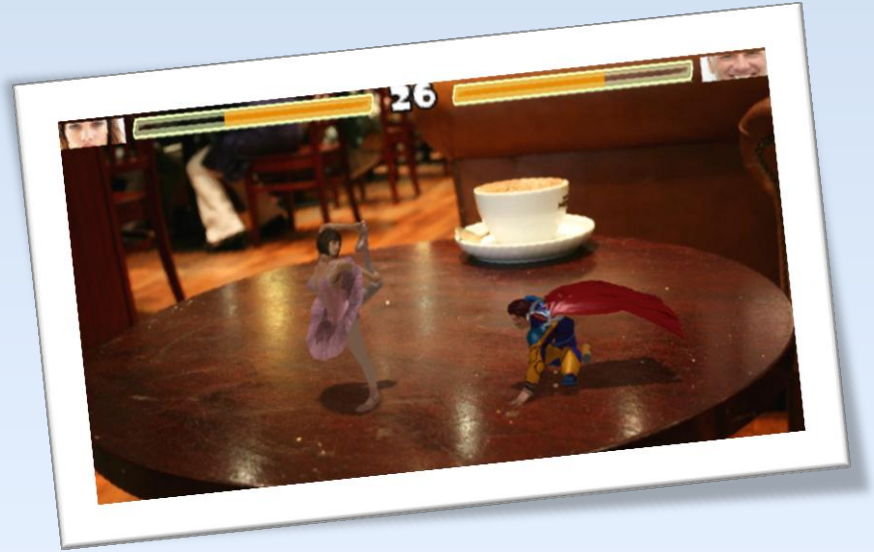
  Downloadable here:
  http://research.scee.net/files/presentations/develop2011/BuildingCommunitywithPlayStationNetwork.pdf

PSVITA PlayStation.Vita

SCEE Research & Development

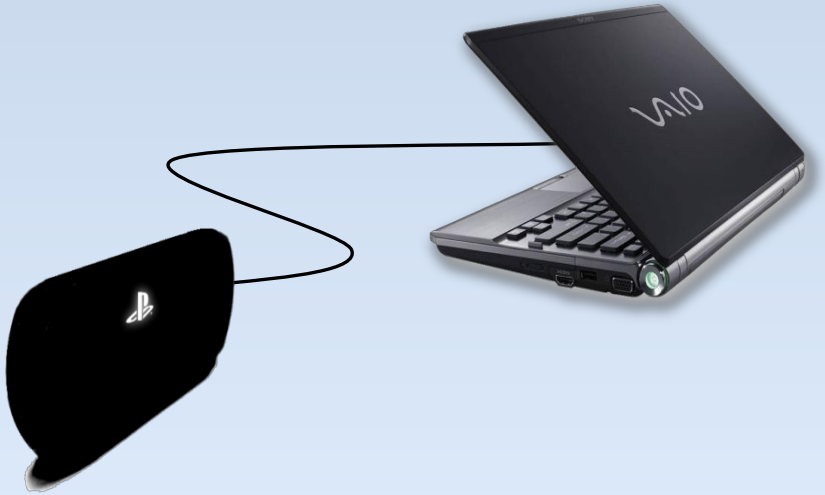
Software

- Augmented Reality (AR)
  - Facial Recognition
  - Natural Marker
  - SLAM



PS®Vita Development Kit

- DevKits
  - Connect via USB to PC
- TestKits
  - Both DevKits & TestKit are consumer console form factor



Simplified Libraries

- We've created some really simplified libraries to help you if you're a small team, or you don't need fine grained control
- Simplified video player
  - Start, stop, rewind
- Simplified audio & MIDI music playback libraries
- Simplified network interface
- The low level libraries are still available if you want to manage everything yourself
- The simplified libraries are optimised so are not slower than the low level API

Other Libraries to Simplify Development

- Vector Maths and Geometry
- Rigid Body Physics Simulation
- Animation
- Facial Recognition

- Even optimised game engine:
  PhyreEngine™

PhyreEngine™

- PlayStation optimised game engine
  - PSP, PS3, PS Vita, PC
- Provided as source
  - Portable, reusable
- Free to use, either full or in part
- Partner Programme
  - Integrating with middleware to ensure developers can choose the best solutions

Tools & Middleware Partners*

AiLive
Allegorithmic
Audiokinetic
Autodesk (inc. Scaleform)
Blitz Tech
CRI
Epic Games
Firelight Technologies (FMOD)
GameSpy Technology
Havok
NaturalMotion
Nvidia
Premium Agency
RAD Game Tools
RakNet
SpeedTree
Terminal Reality
Trinigy
Umbra
Vicious Cycle
Web Technology

*List only show T&M partners who are ready to provide evaluation version right now, others to follow – valid July 2011

Facial Recognition Tech Demo

- Video Chat tech demo By SCE Japan Studio
- Uses facial recognition libs:
  - Face Detection
  - Feature Detection
    - Eyes, Nose, Mouth, etc
  - Head tracking
  - Face attributes
    - Smile, Male/Female, Age (Adult/Child), Glasses
    - Eyes open/close (early alpha code)

Augmented Reality (AR)

- Most games until now use a special tracking marker
- On PS Vita R&D in Natural Marker technology allows us to use almost any image as a tracking marker
- Real time **S**imultaneous **L**ocalisation **A**nd **M**apping (SLAM)
  - Use any patterned surface as a marker

AR for gaming

- Order of priority when implementing AR:
  - Robustness (reliable)
  - Speed & Frame Rate
  - Accuracy (dependable)

- Just in case you are interested in medical AR research they come up with the following order of priority:
  - Accuracy (it needs to be dependable otherwise could have fatal consequences)
  - Speed & Frame Rate (easy to achieve as cost of hardware is not a major factor)
  - Robustness (lower as location, lighting, etc are usually predictable)

PlayStation®Vita Inception

- Unlike our previous hardware, PS Vita was not created in isolation by our hardware engineers in Japan
- Huge amounts of input from our 1st party as well as 3rd party studios (inc T&M partners) from a very early stage
- Not just the inputs mechanisms and gameplay features you'd expect, but also
  - Hardware – SOC: CPU and GPU
  - Software – Libraries and API design and implementation
  - Tools – design and implementation
- All aspects of PS Vita has been developed from the ground up with developers in mind
- In a very real sense it has been created by developers for developers

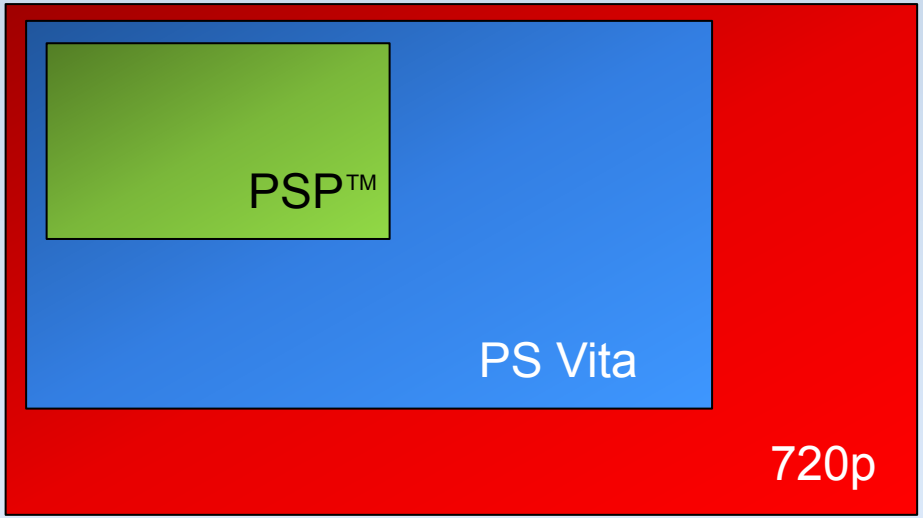
Useful Links

- **tpr_registration@scee.net** - 1st port of call in becoming a registered developer
- **https://www.tpr.scee.net/** - Registered developers on existing PS platforms
- **http://www.worldwidestudios.net/xdev** - For registered developers who wish to propose titles to SCEE XDEV
- **SCEA PubFund** - Please contact your SCEE account manager who will reach out to SCEA Developer Relations who oversee PubFund (open to all developers globally)
- **http://research.scee.net/** - SCEE R&D public site where we place all public presentations.

Graphics on PlayStation®Vita

- Introduction
- Hardware
  - Tile Based Deferred Rendering
- Software
  - Programming Model

Platform Comparison: Screen



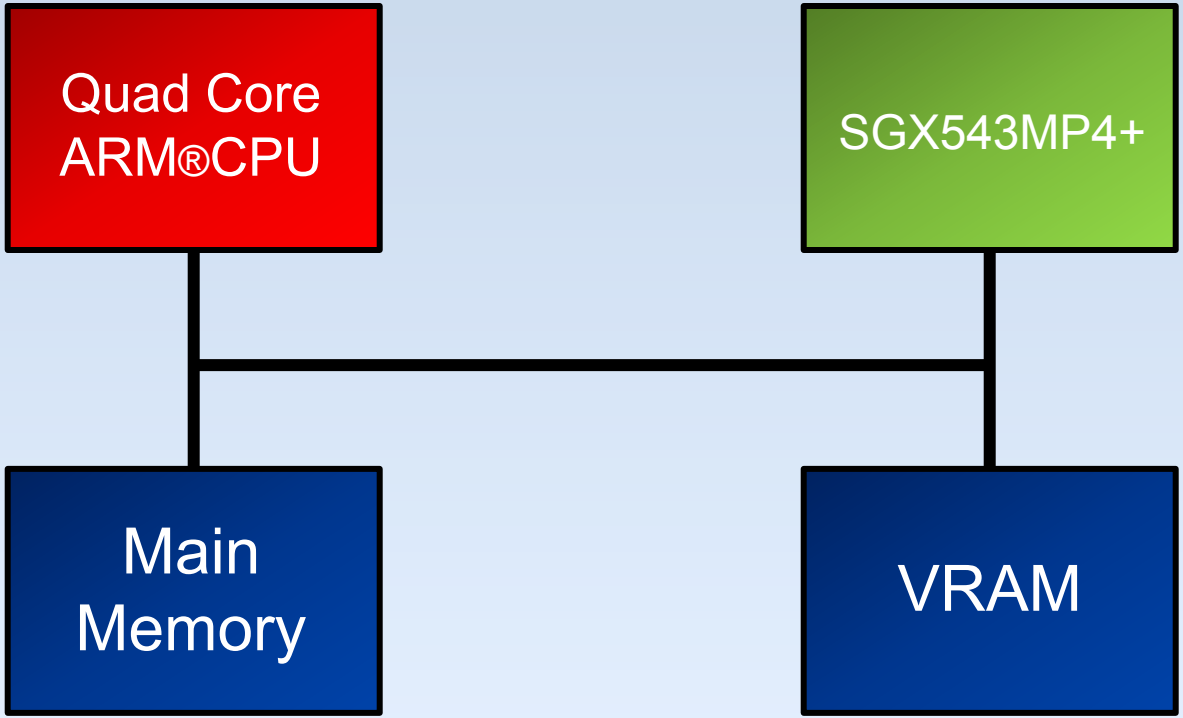
PSP™ = 480x272

PS Vita = 960x544

720p = 1280x720

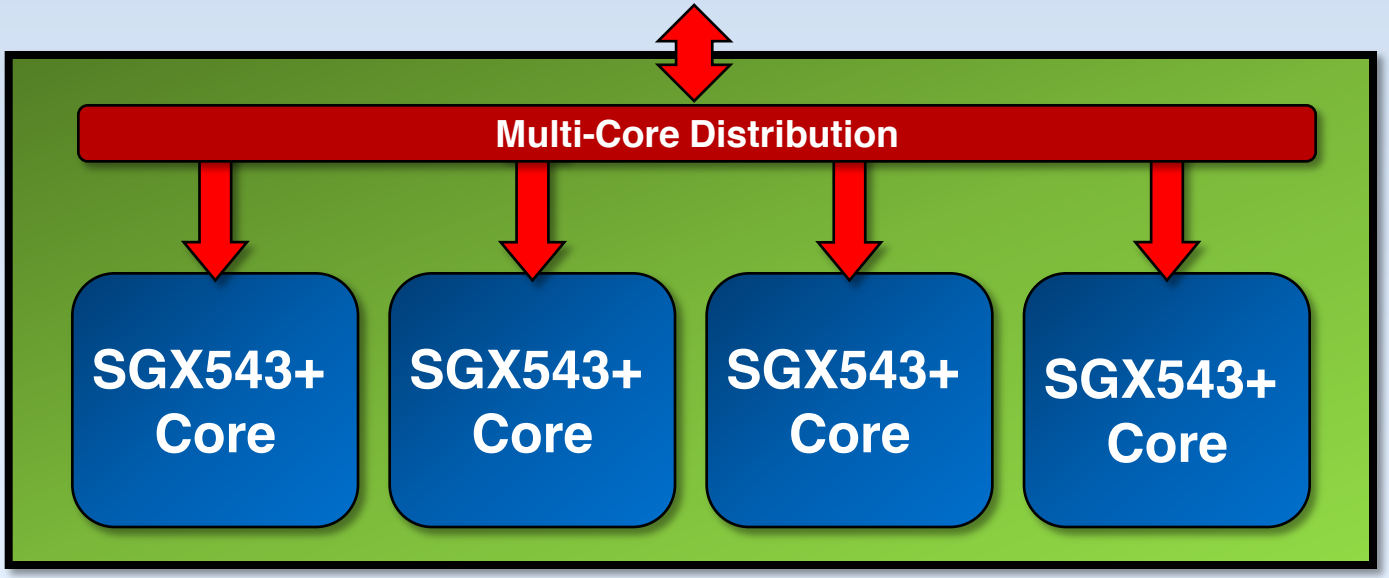
- High Resolution
- High Pixel Density
- Efficient MSAA

PlayStation®Vita System



POWERVR SGX543MP4+

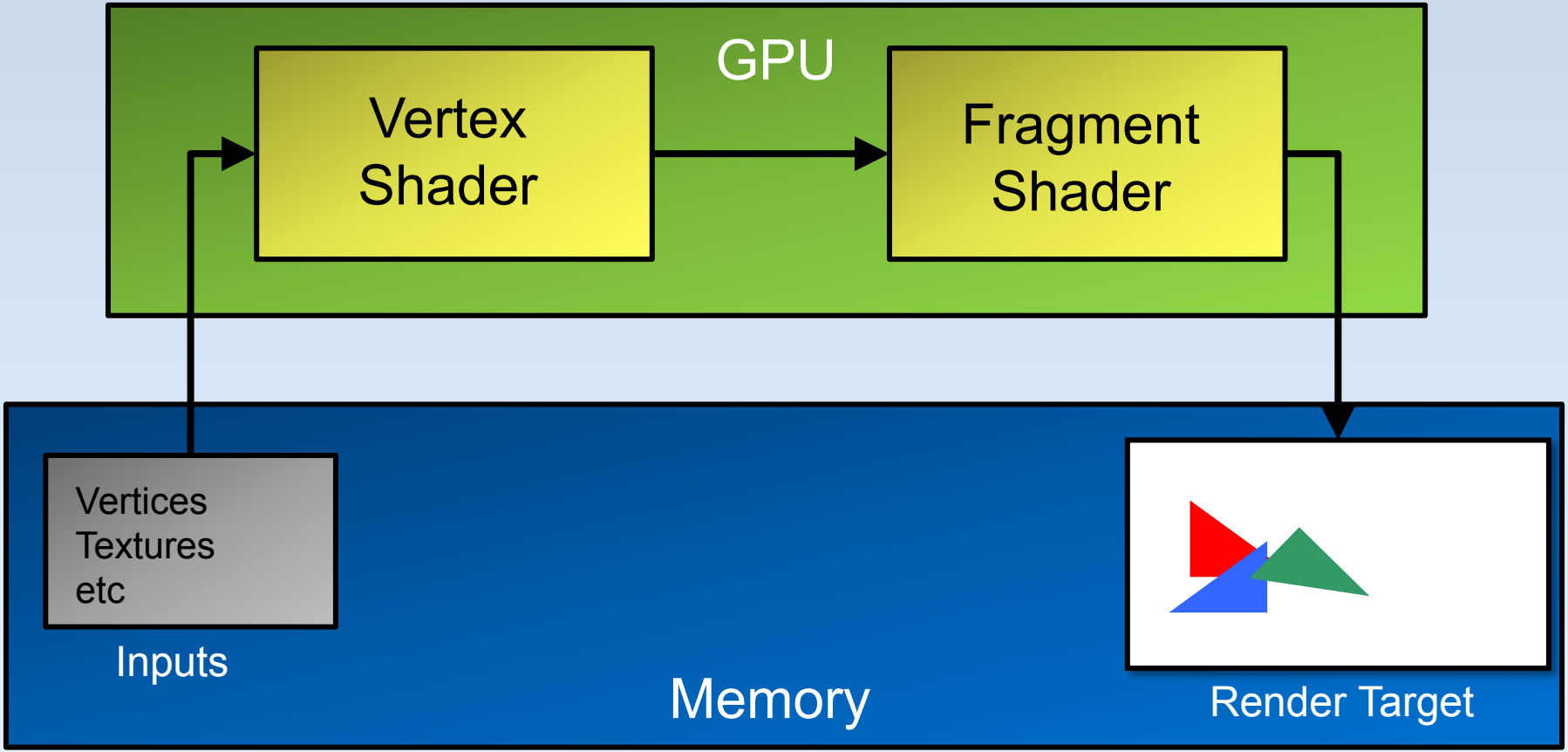
- Power-efficient multi-core GPU
  - Automatic dynamic load balancing



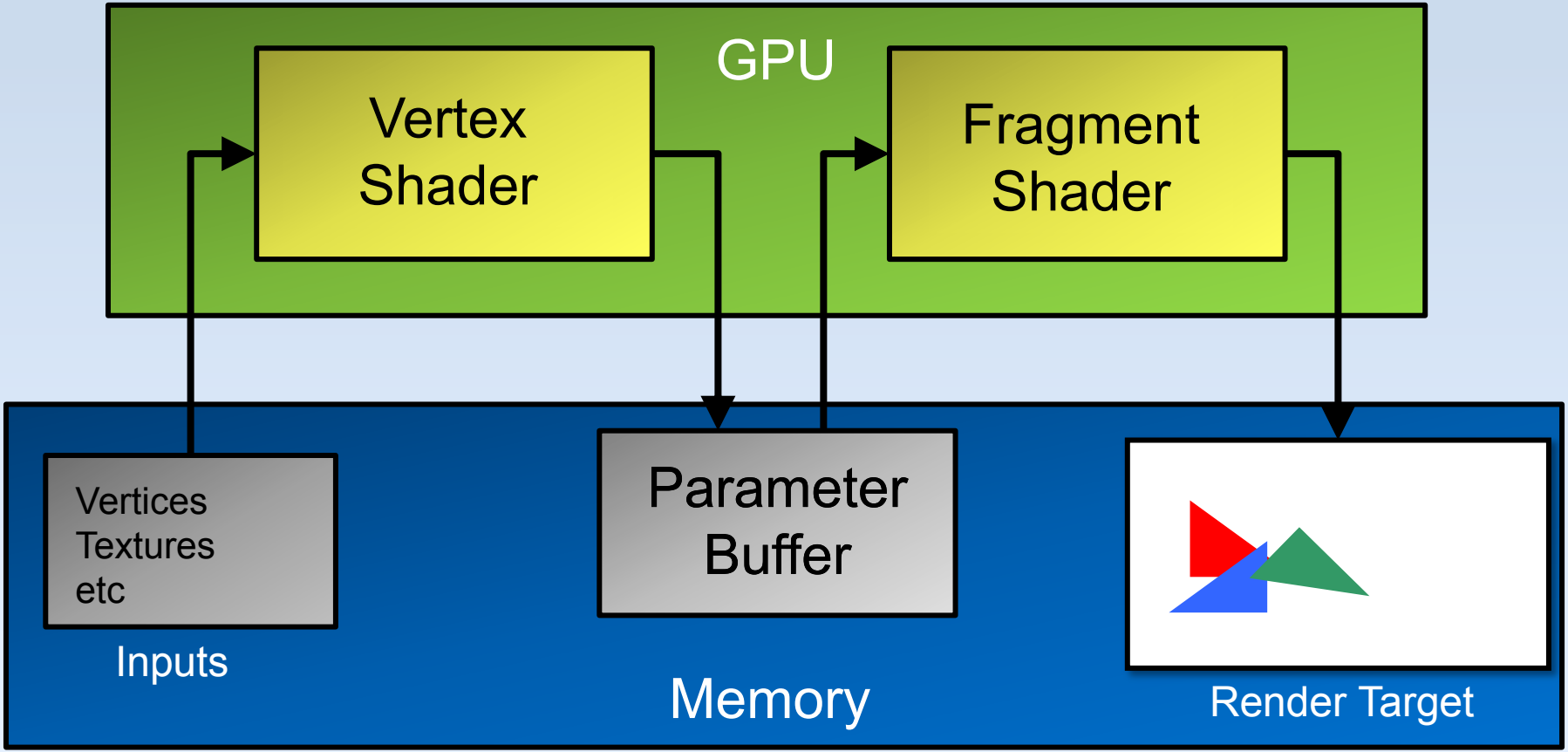
Hiding Latencies

- Multiple threads in parallel on each core
  - Switch to another thread while stalled
  - Zero cycle overhead
  - Vertex and fragment threads at the same time
- Textures can be fetched ahead of time
  - Before the shader runs

Immediate Mode Renderer

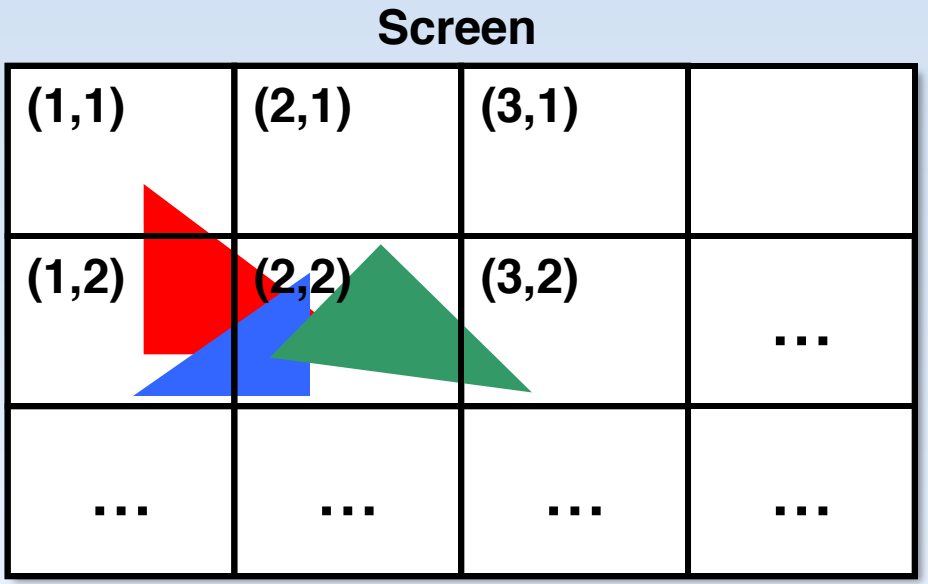


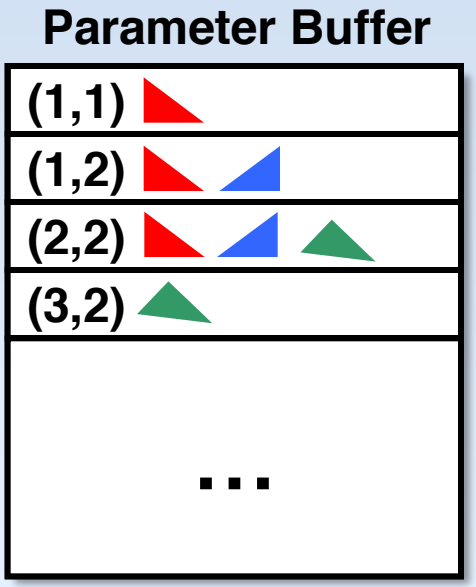
Tile Based Deferred Renderer



Tile-Based Deferred Rendering

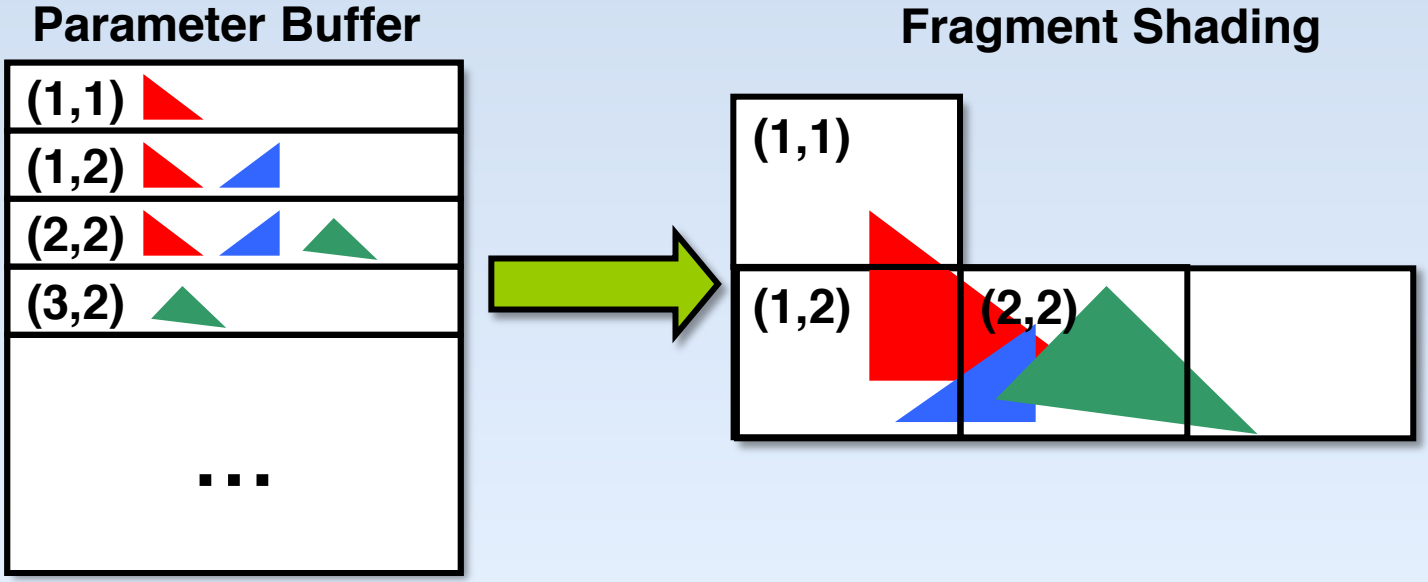
- Screen is split into tiles
- Each tile references relevant primitives in the parameter buffer



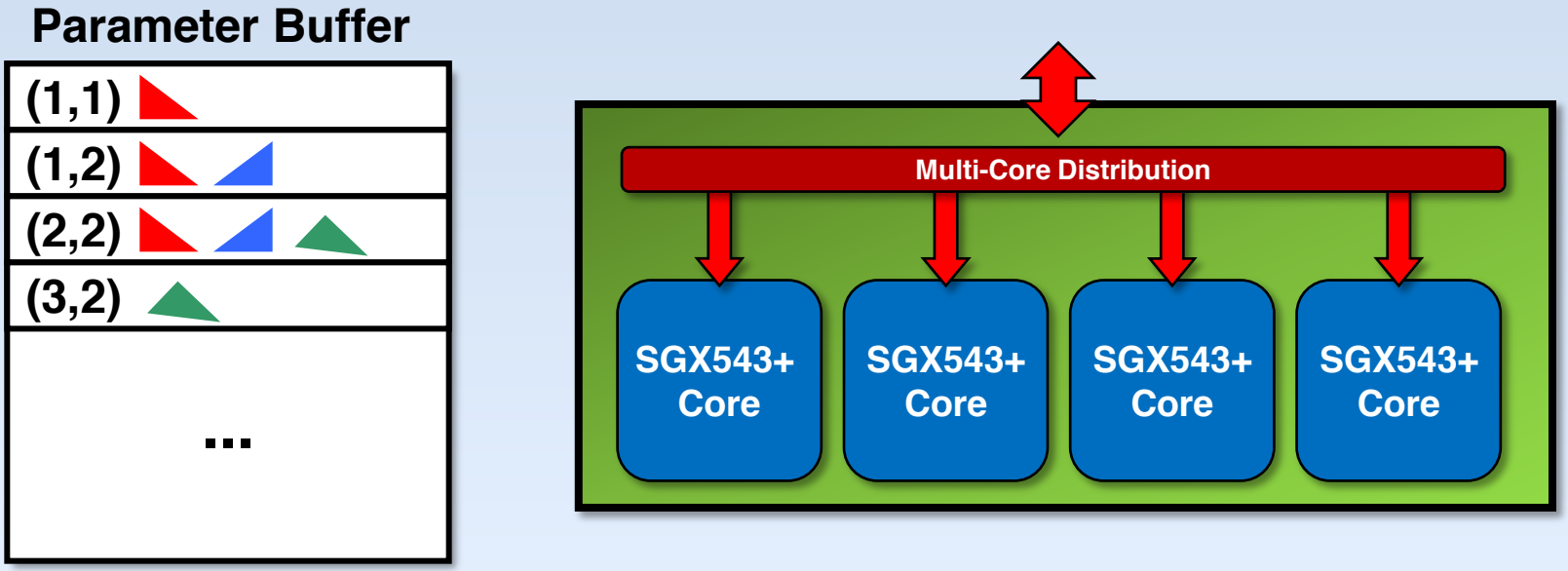


Tile-Based Deferred Rendering

- Fragments in each tile processed for shading

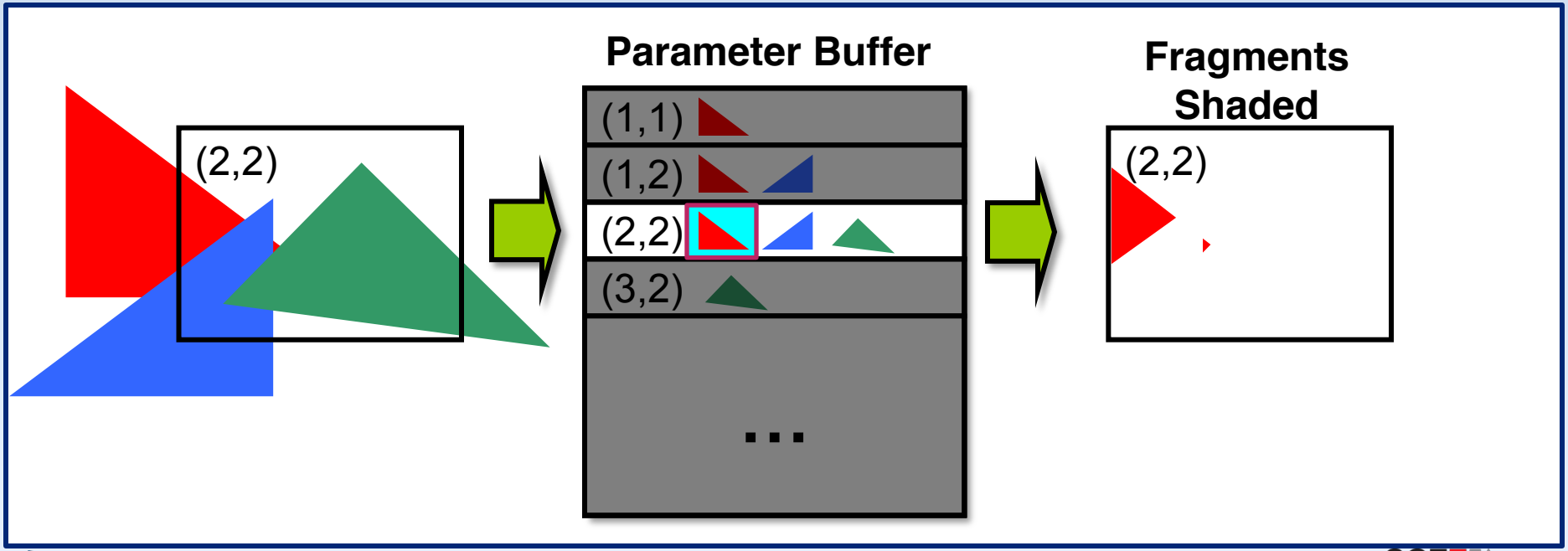


Tile Distribution



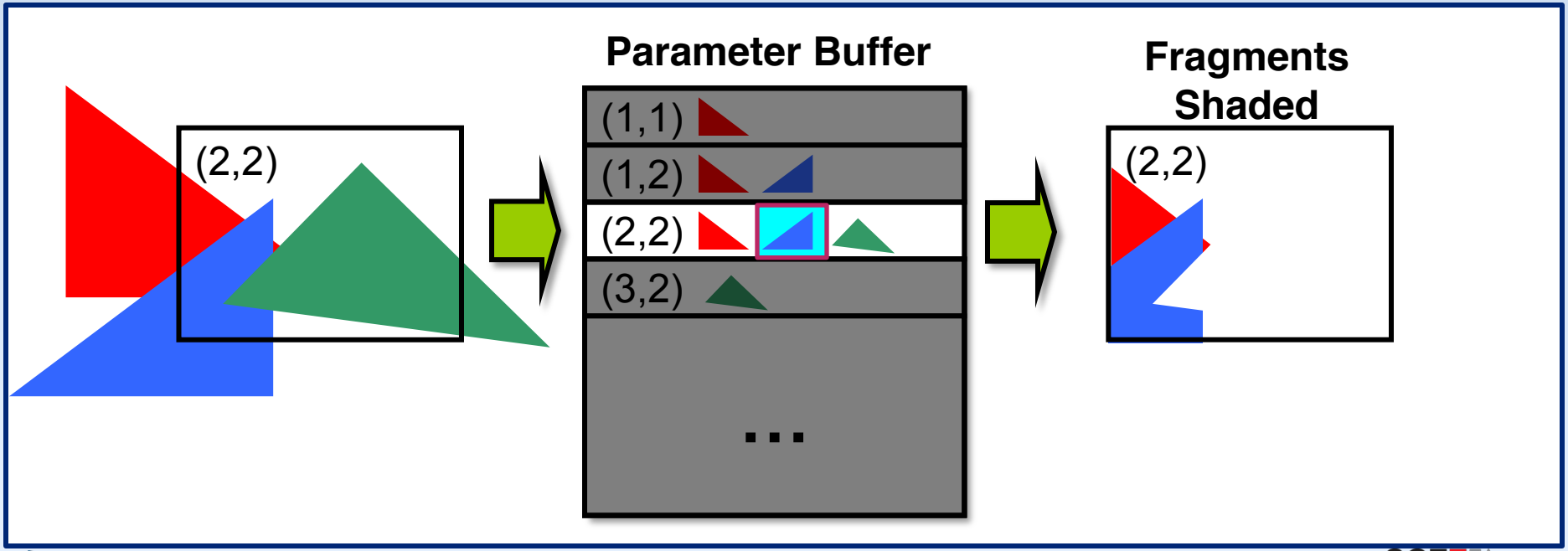
Opaque Geometry

- Only visible fragments in a tile get shaded



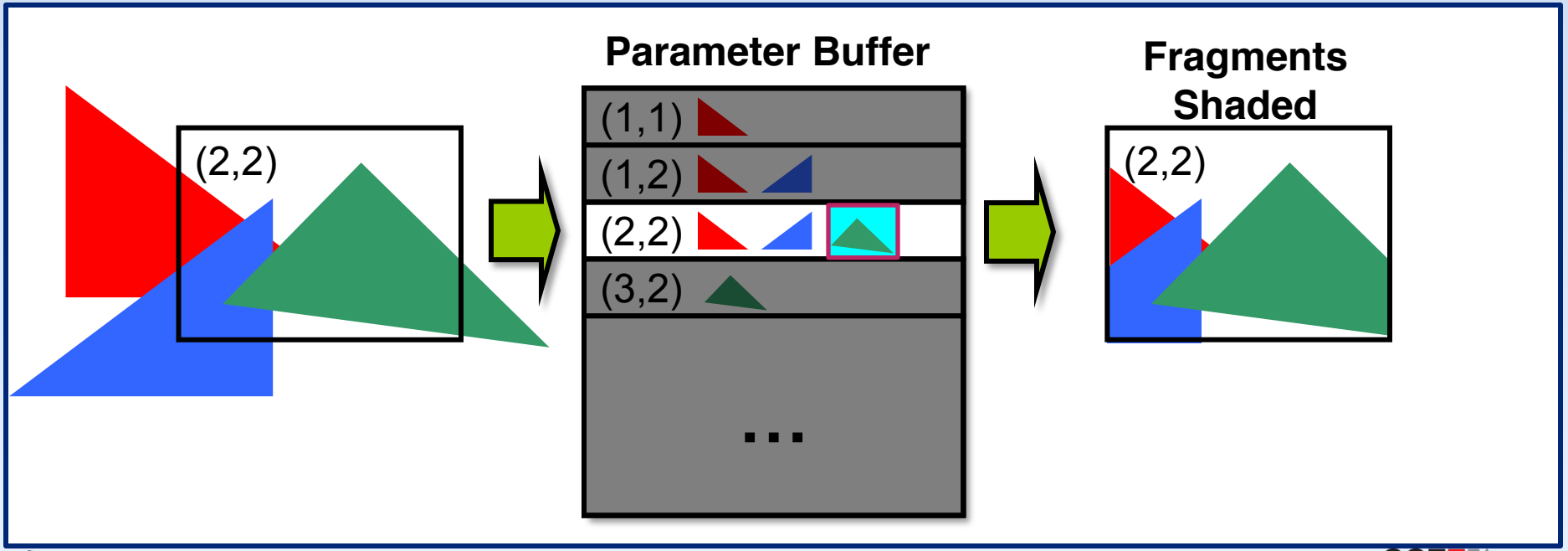
Opaque Geometry

- Only visible fragments in a tile get shaded



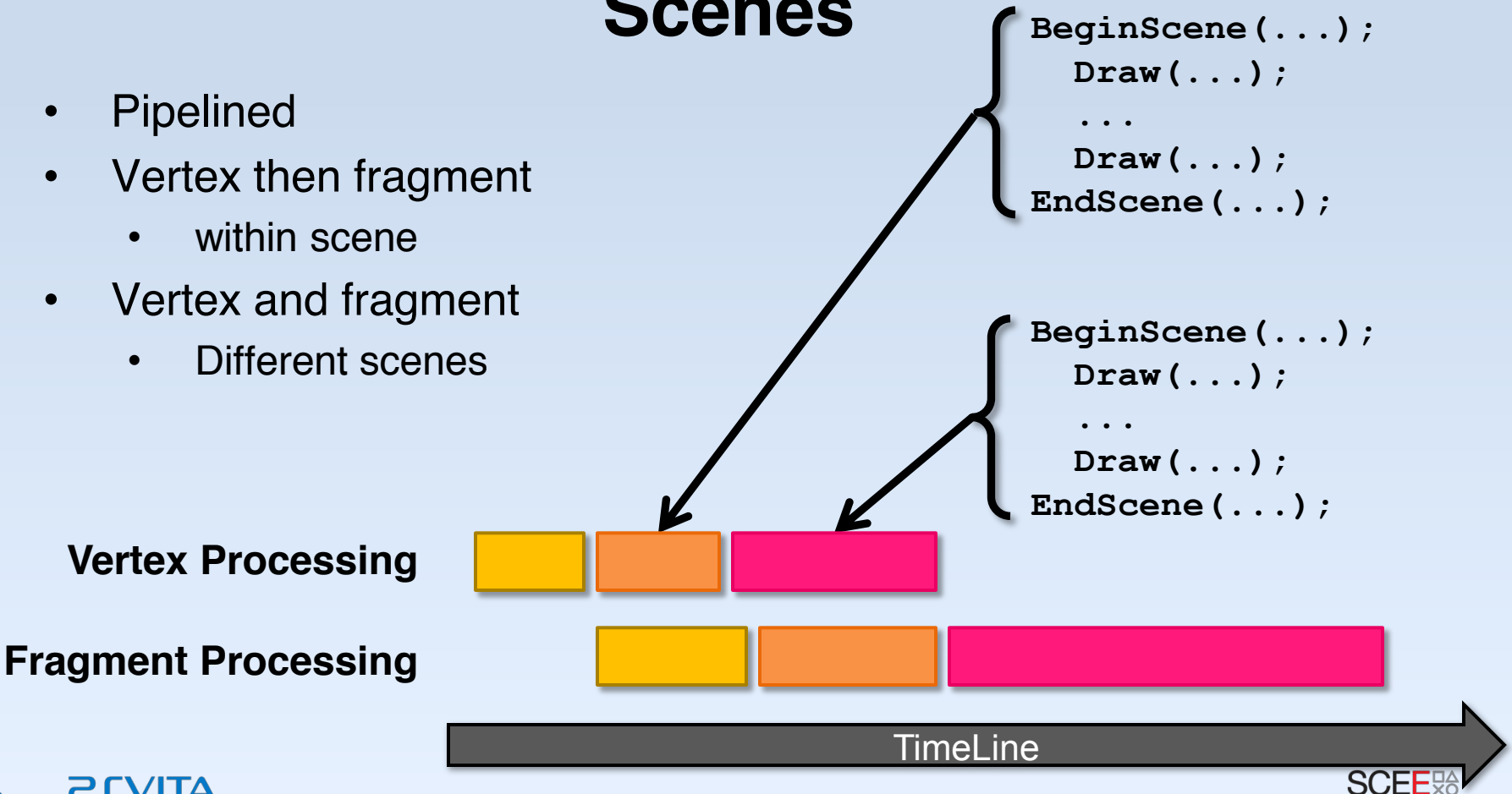
Opaque Geometry

- Only visible fragments in a tile get shaded



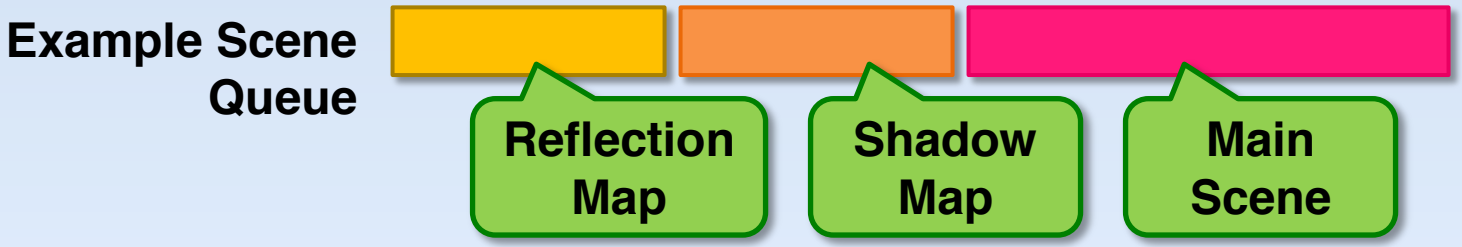
Scenes

- Pipelined
- Vertex then fragment
  - within scene
- Vertex and fragment
  - Different scenes



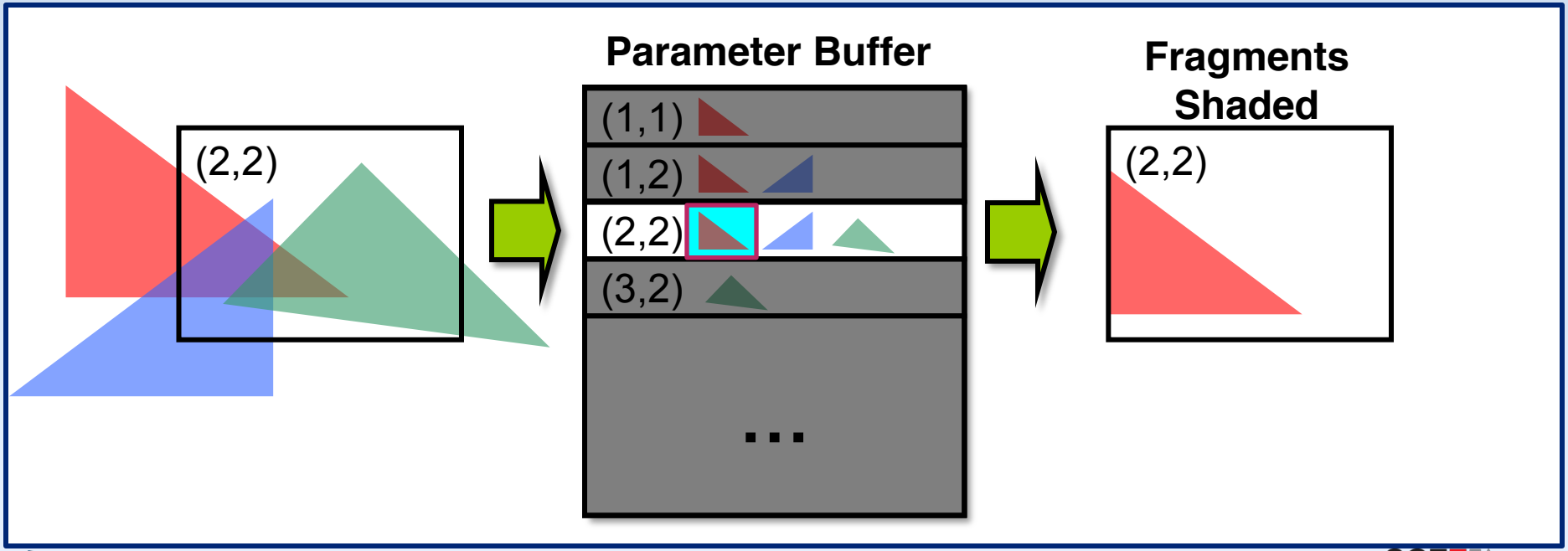
Submission Process

- GPU consumes work in relatively large jobs



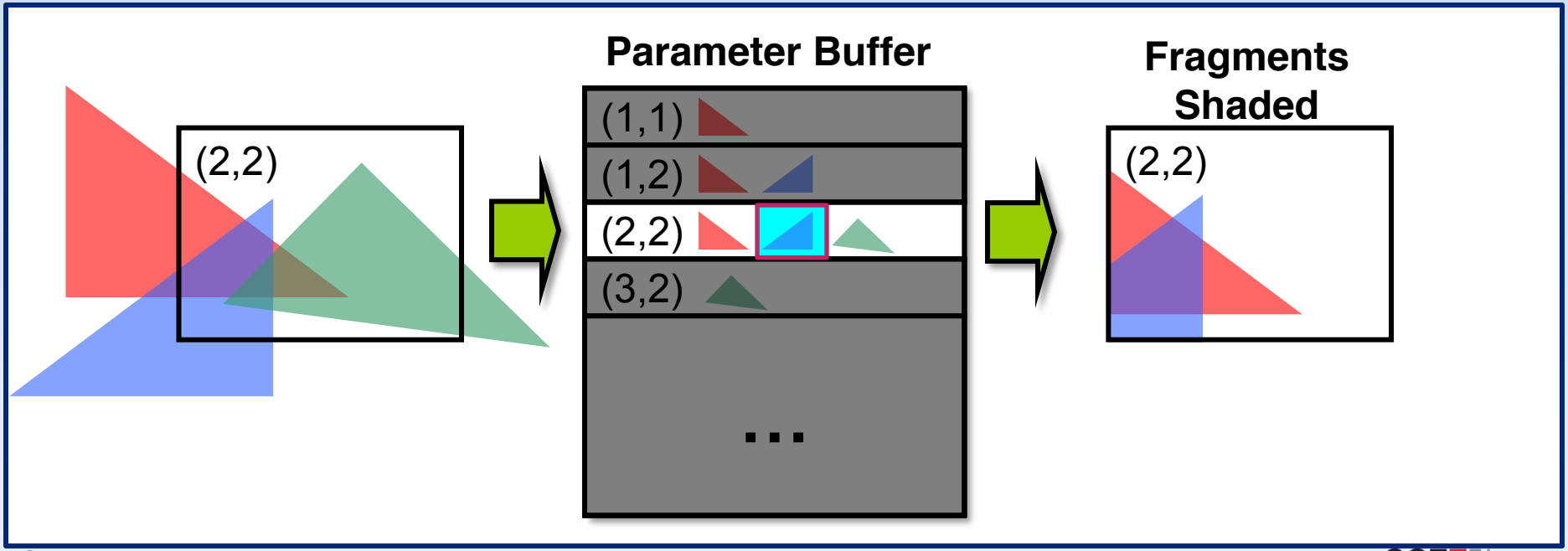
Translucent Geometry

- Each blended fragment gets shaded multiple times
- Processing occurs in submission order



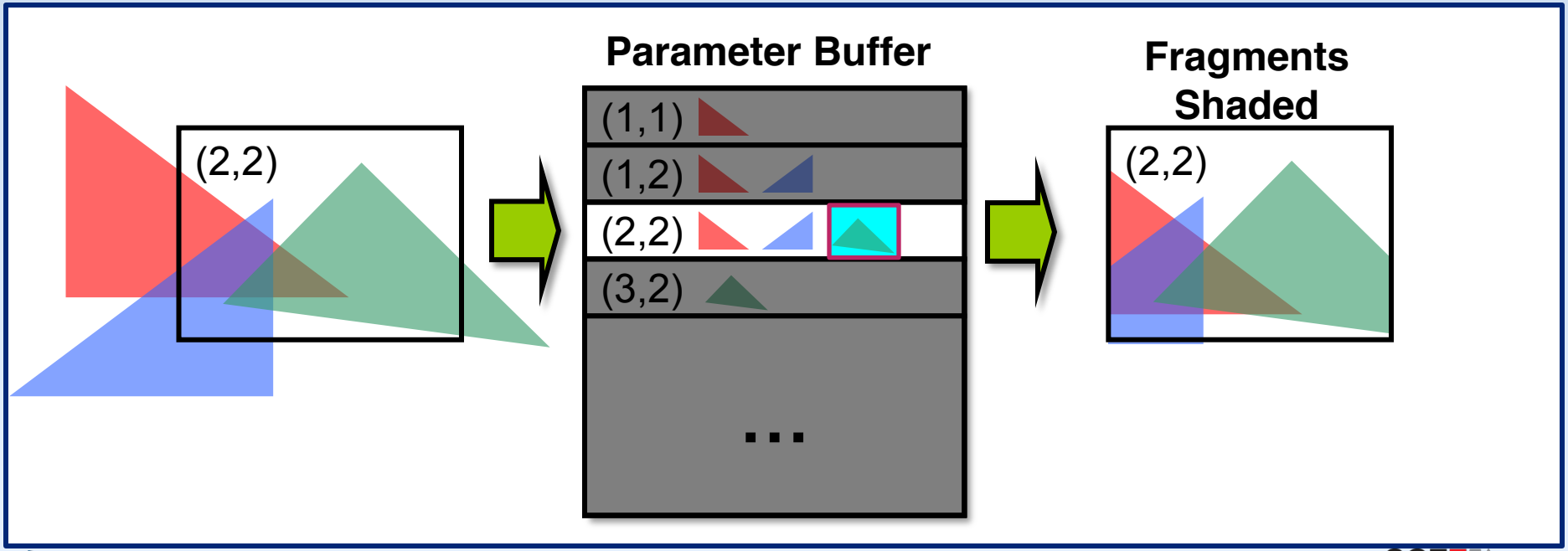
Translucent Geometry

- Each blended fragment gets shaded multiple times
- Processing occurs in submission order

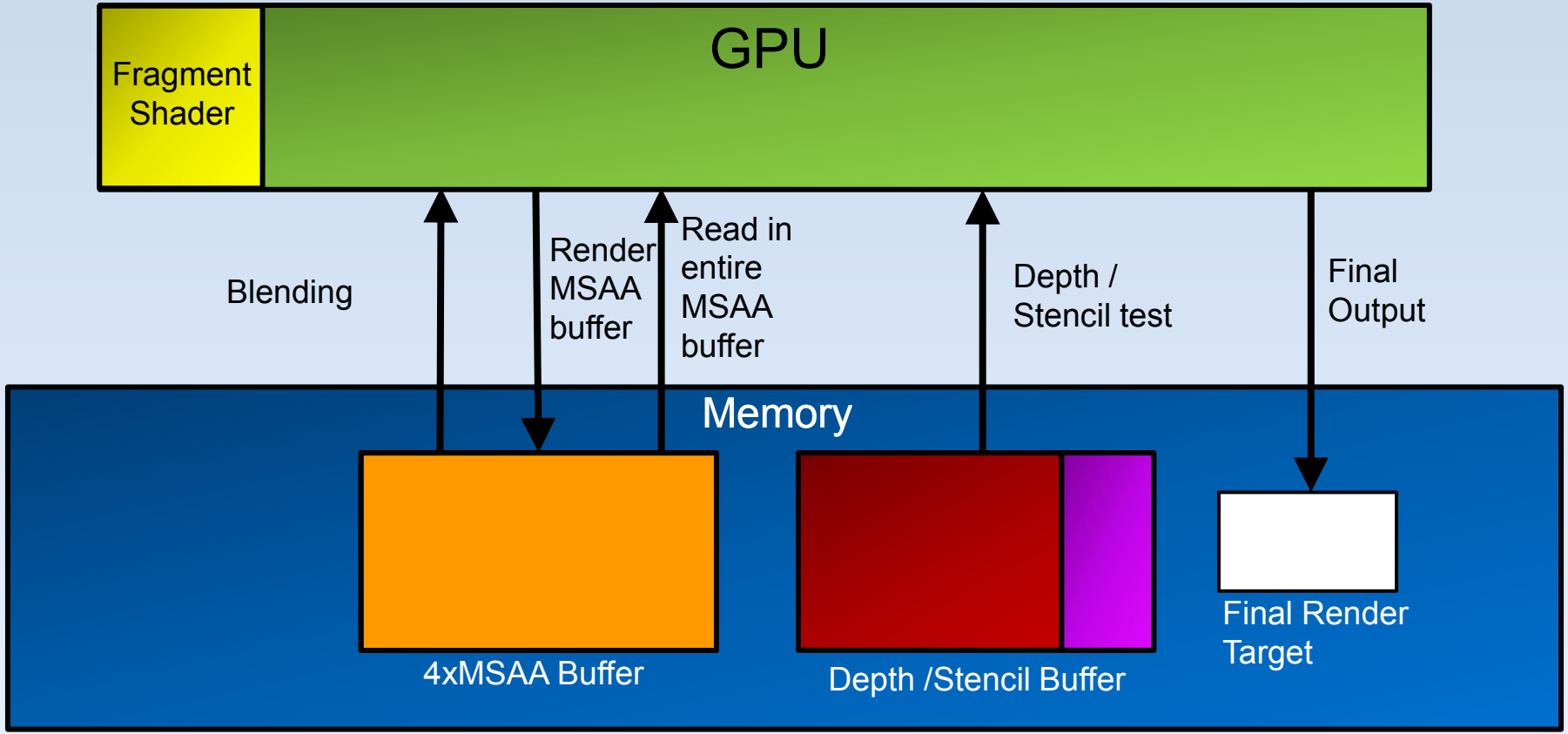


Translucent Geometry

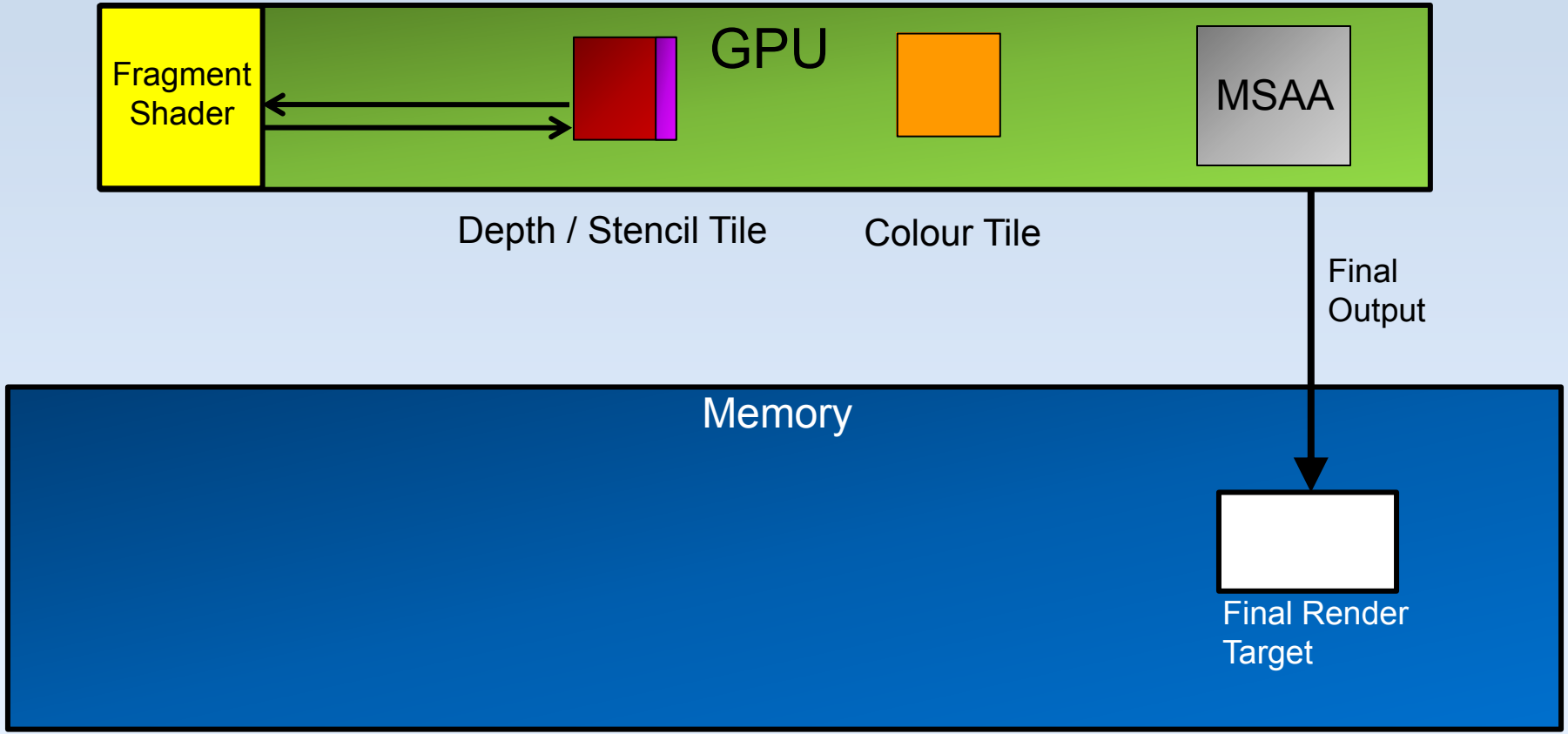
- Each blended fragment gets shaded multiple times
- Processing occurs in submission order



Immediate Mode Renderer



Tile Based Deferred Renderer



Programming Model

- No unnecessary complications
- Tile-based processing handled behind the scenes
- Leaving you a familiar API interface

Shader Model 3.x+

- Familiar format of
  - Vertex / Fragment Shaders
  - Geometry
  - Textures
  - Render Targets
- Reduce asset size, but keep same visual fidelity
- Can be up and running in a few weeks

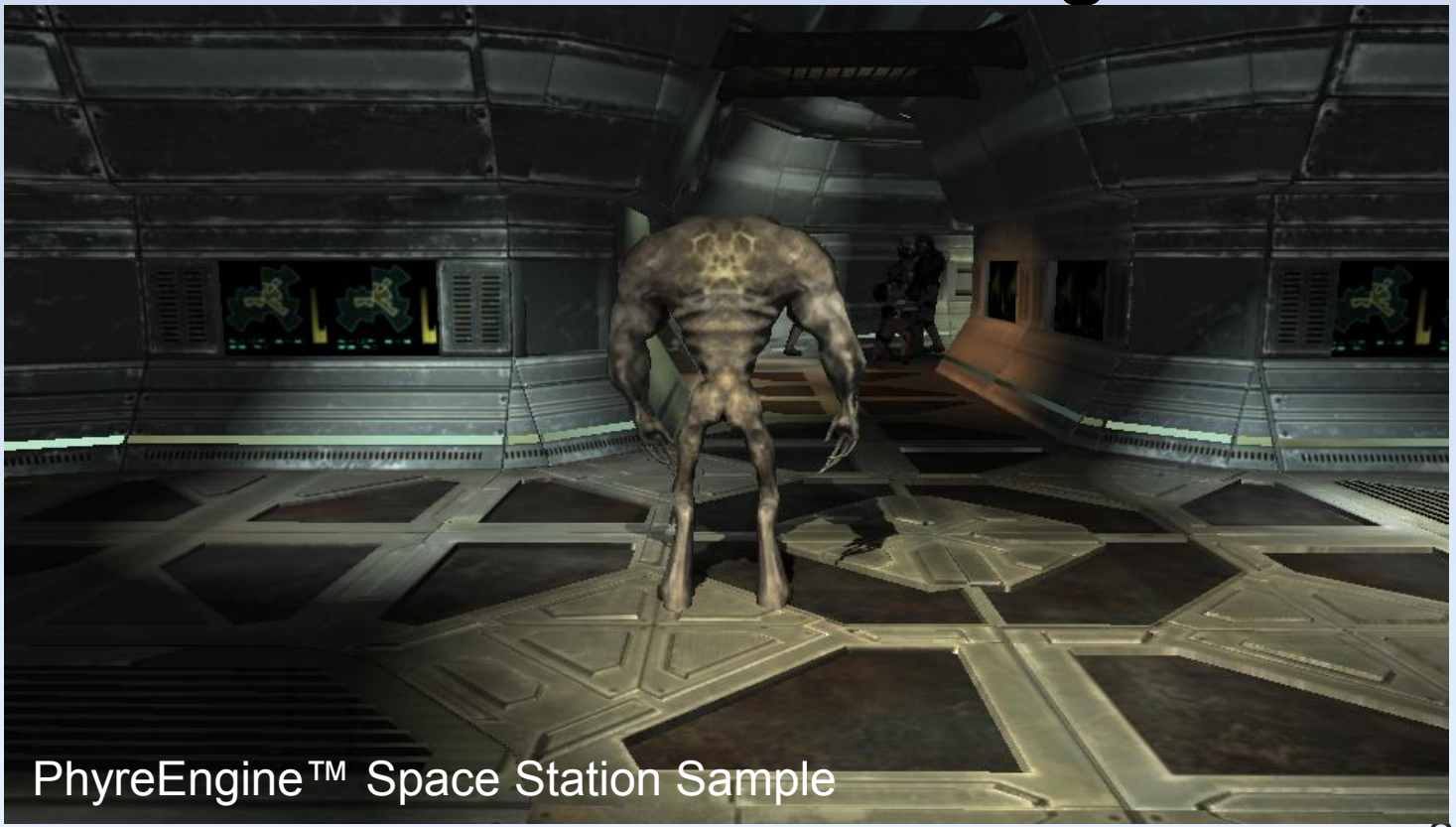
Dynamic Lighting

Rich Environment Detail

Water Effects

Post Processing

Deferred Rendering



Razor

- Next Gen version of SN Tuner and GPAD rolled into one
- Analyse CPU and GPU at the same time
  - Host PC
  - Target on screen HUD
- Fully integrated into Visual Studio
- Easy to track down you bottlenecks
  - Comprehensive CPU and GPU counters
  - Covers all cores
  - Intuitive Interface
- Lots of really cool features I'm not allowed to talk about

Summary

- Created from the ground up for developers
- Rich set of interfaces
- Hardware does the hard work
- Shader Model 3.x+
- Powerful and intuitive tools
- Developer Friendly

Thanks To

- SCEI
- SCEE R&D
- SCE ATG
- SN Systems
- WWS 1st and 2nd party developers