NVIDIA®

DX10 – новые возможности и вопросы производительности

Юрий Уральский, NVIDIA

DX10

- Много новой функциональности
  - Не только геометрические шейдеры
- Возможность перестроить ваш графический движок
- Мотивация к созданию новых эффектов

Copyright © NVIDIA Corporation 2005

Outline

- D3D10 API
- Ресурсы и представления
- Инстансинг
- Геометрические шейдеры и stream out
- Массивы текстур и другие возможности

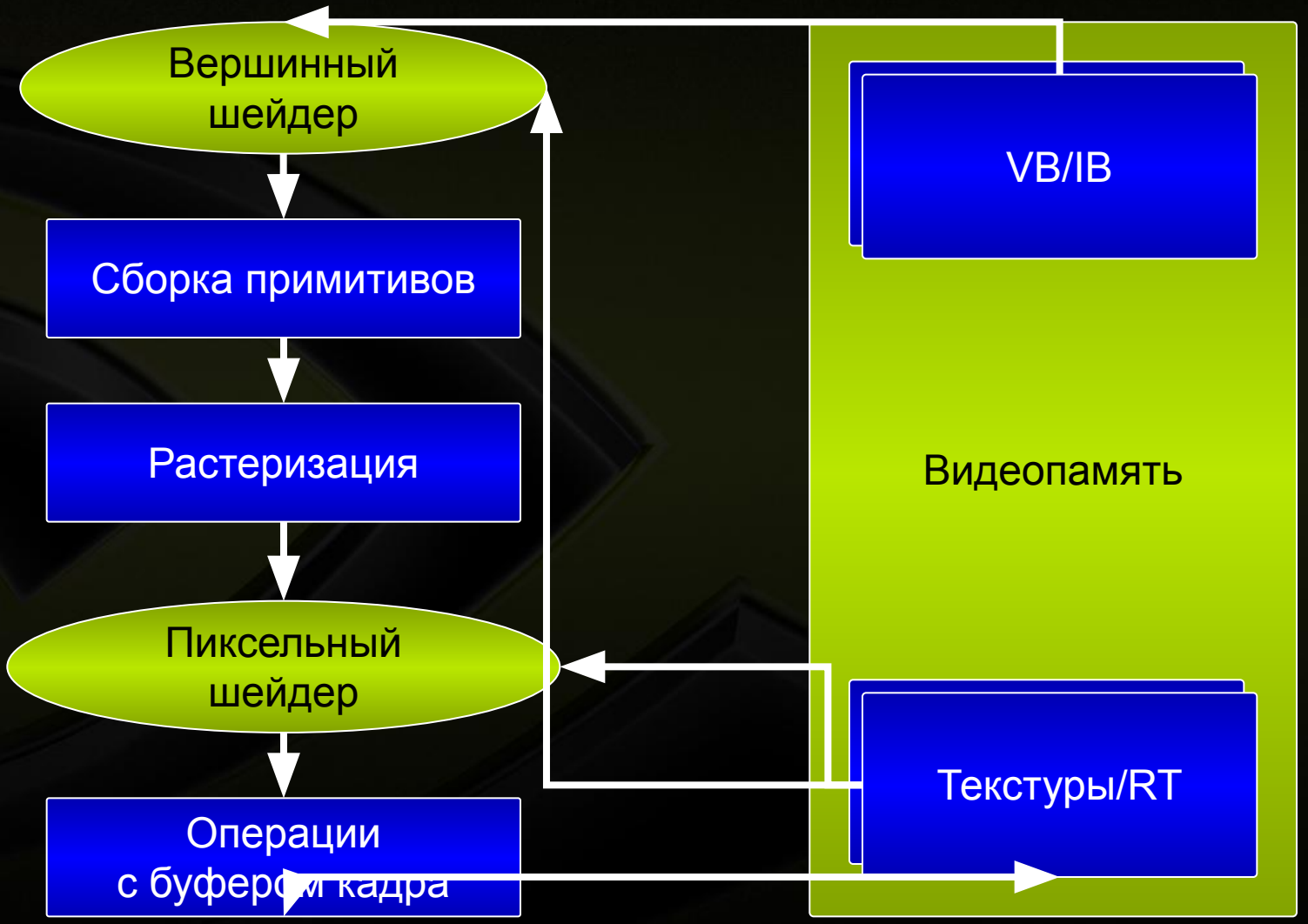
Copyright © NVIDIA Corporation 2005

10 > 9 !

- Новый, компактный API
  - Меньше `HRESULT`, больше `void`
  - Стейт-объекты
- Новая модель драйвера
  - Позволяет значительно сократить CPU overhead
- Виртуализация ресурсов
  - Больше нет lost devices!
- Новый уровень программируемости
  - Унифицированные шейдеры (SM 4.0)

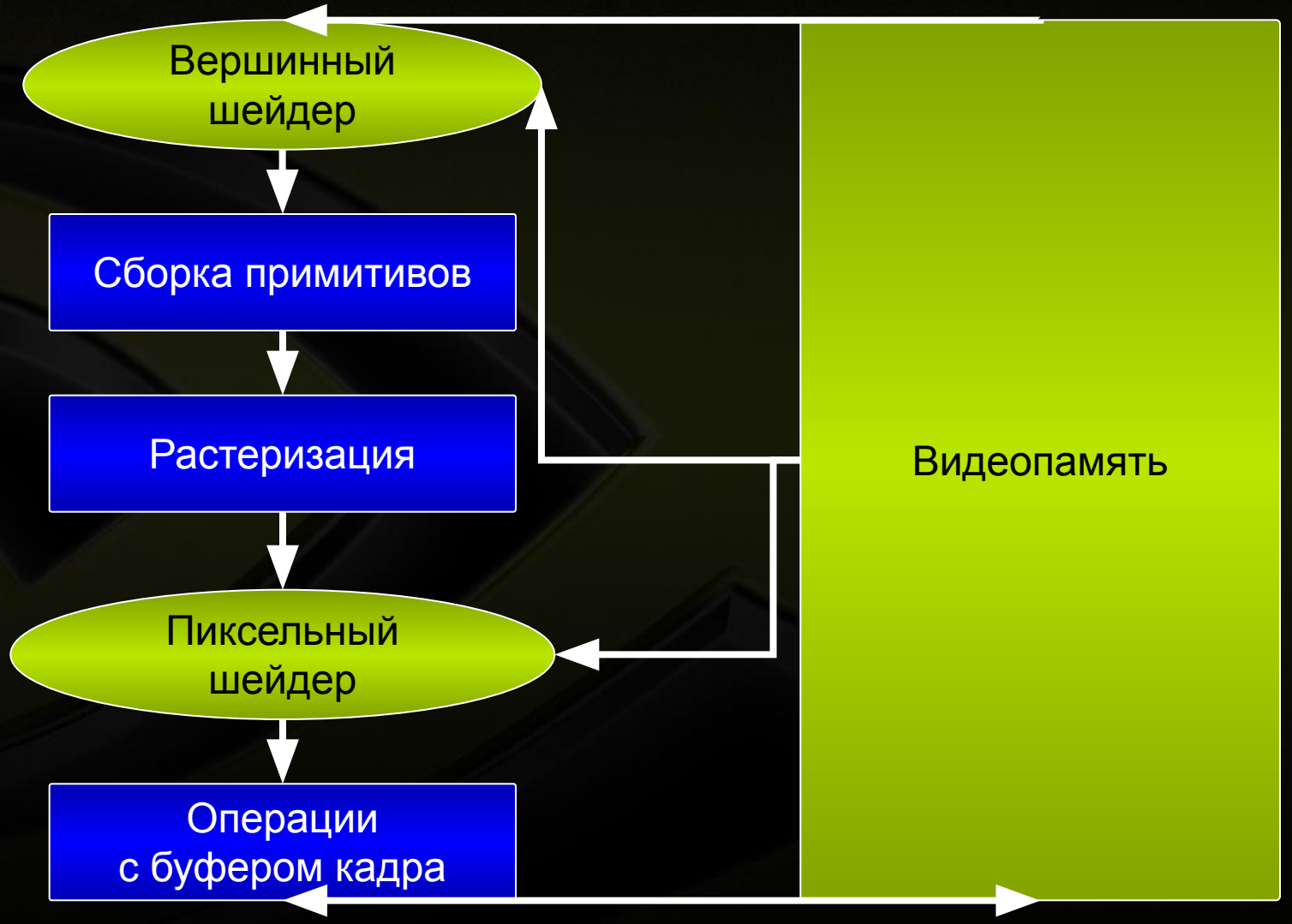
Copyright © NVIDIA Corporation 2005

D3D9



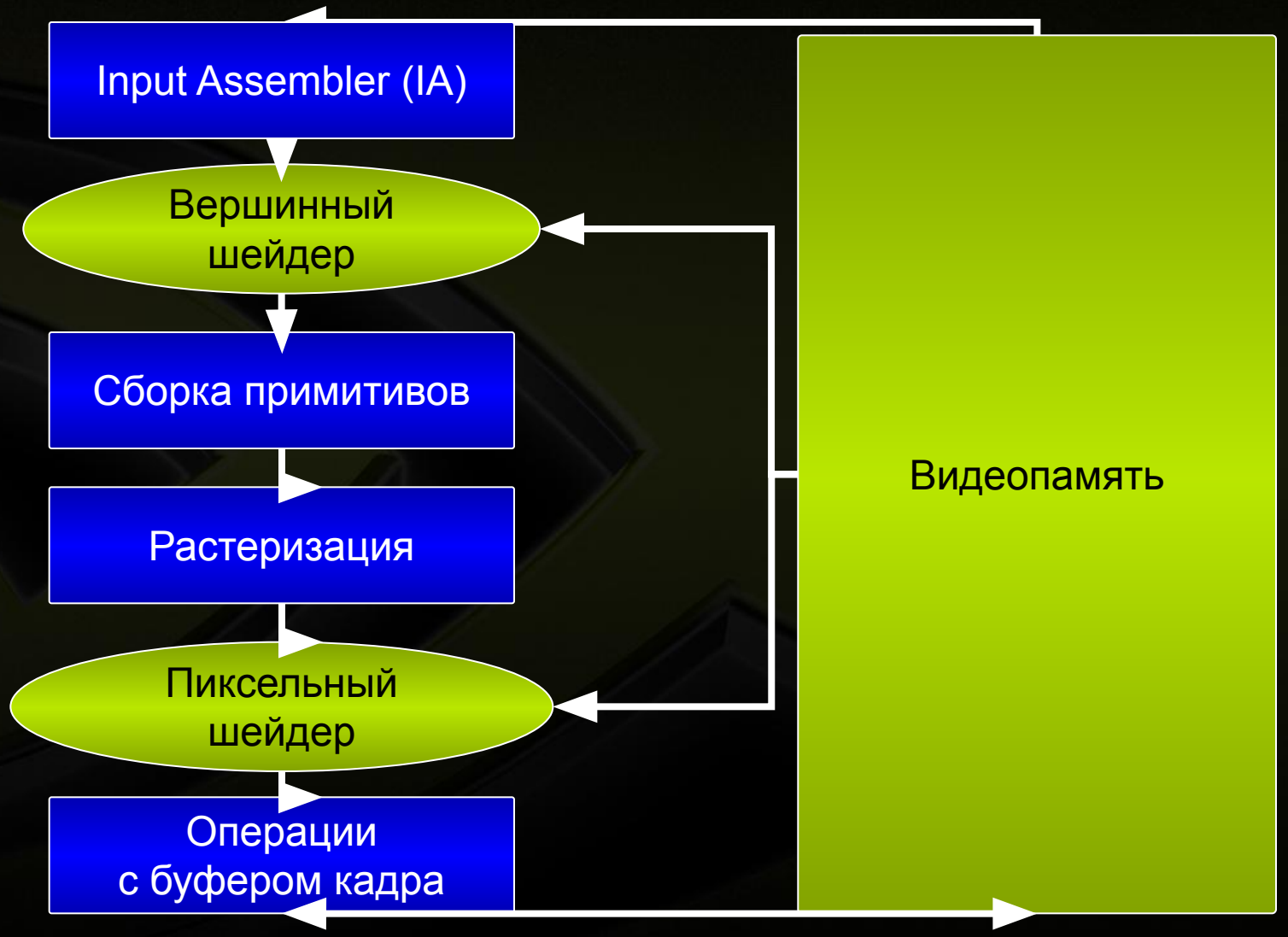
Copyright © NVIDIA Corporation 2005

D3D10



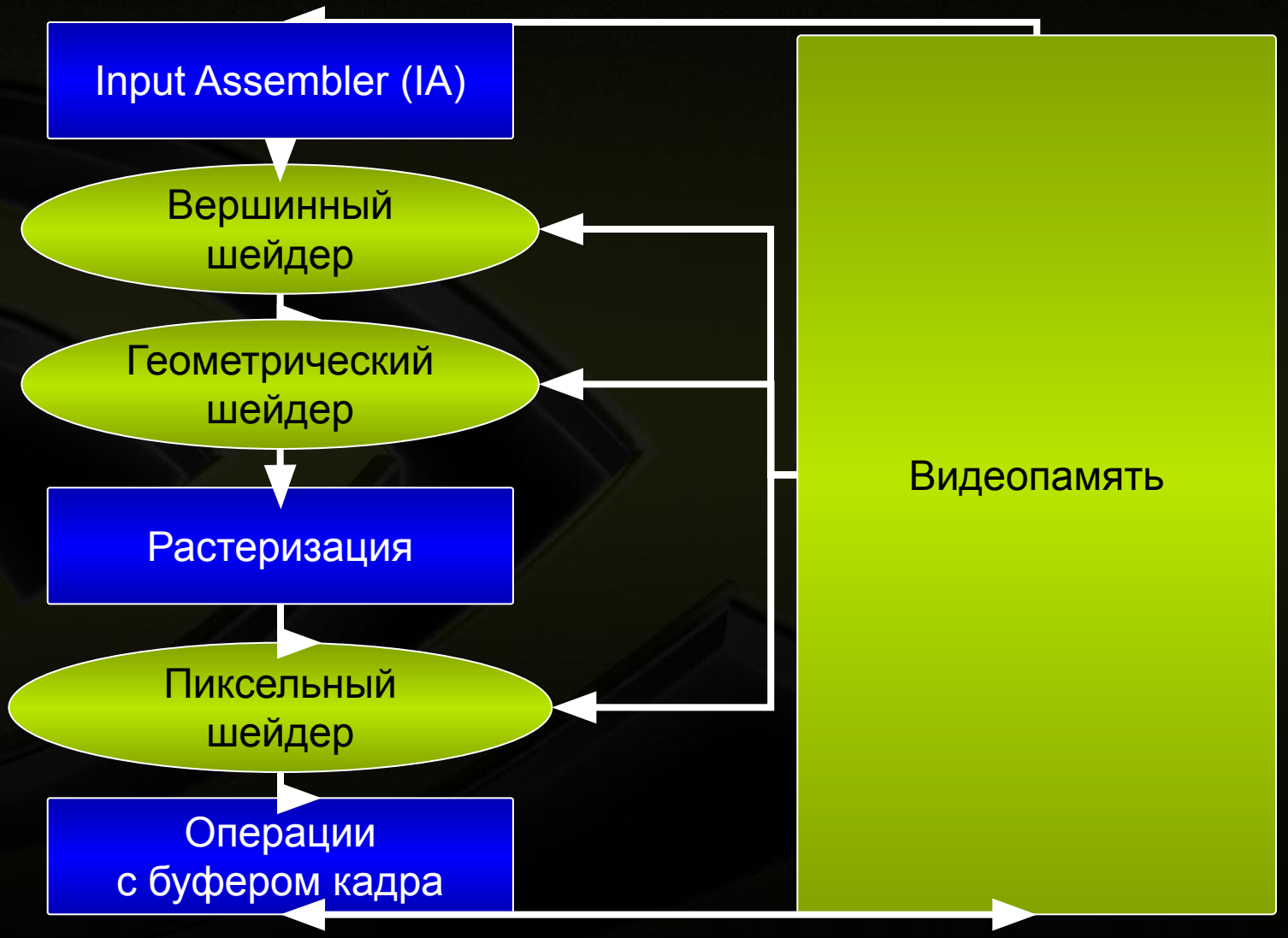
Copyright © NVIDIA Corporation 2005

D3D10



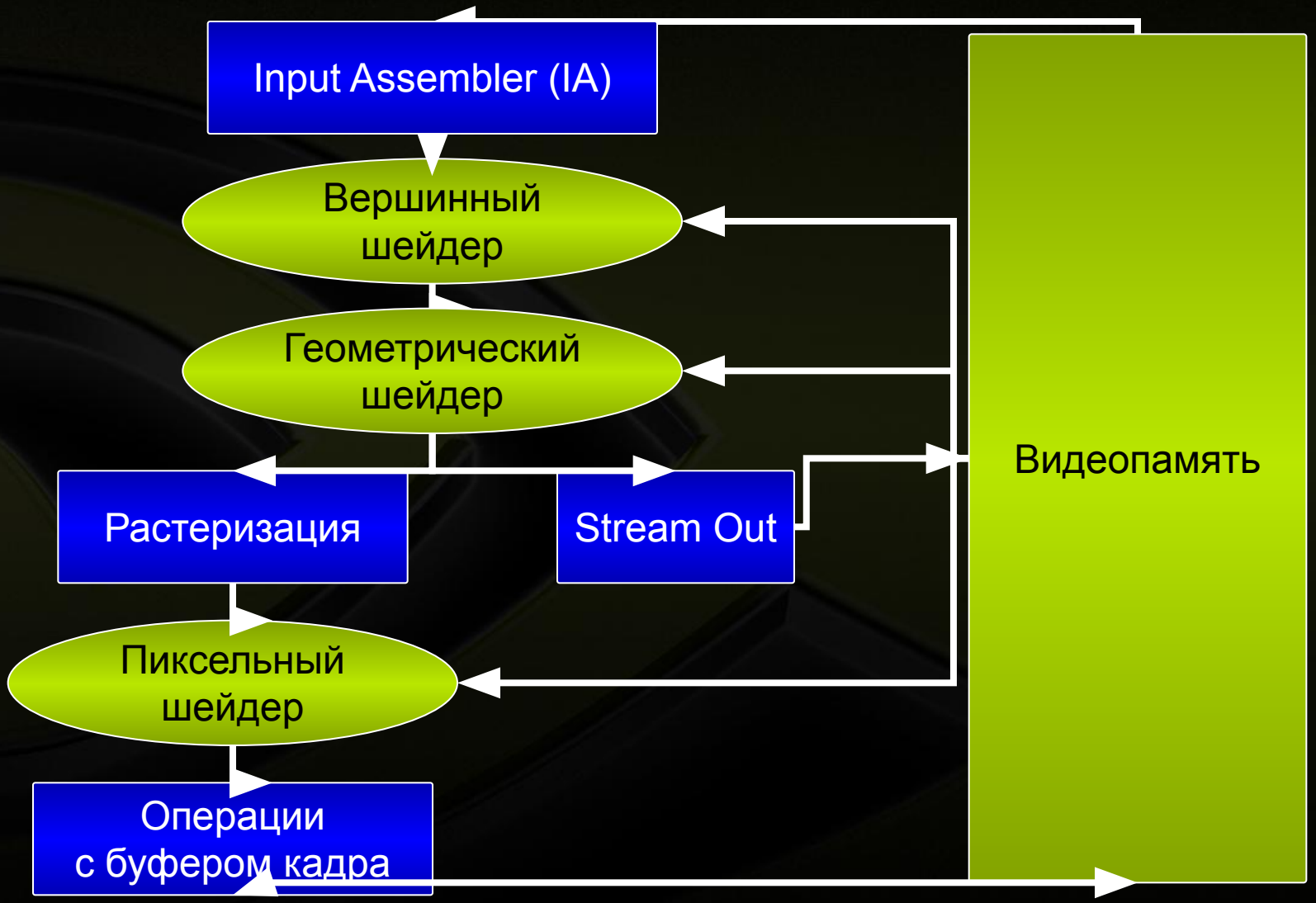
Copyright © NVIDIA Corporation 2005

D3D10



Copyright © NVIDIA Corporation 2005

D3D10



Copyright © NVIDIA Corporation 2005

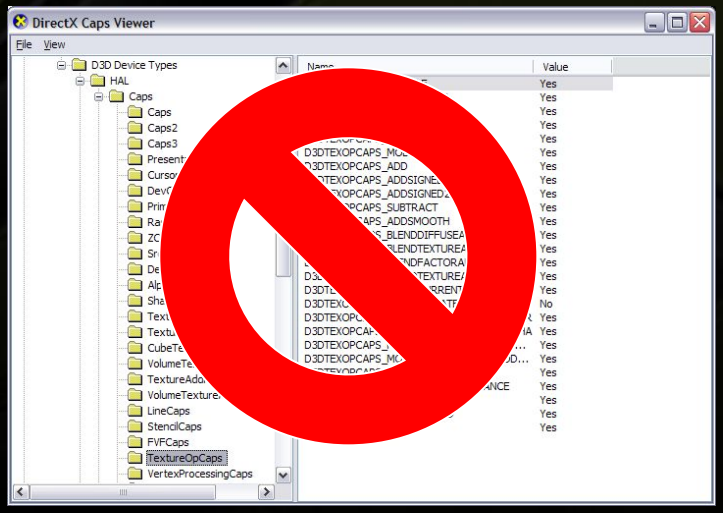
D3D10 API

- `ID3D10Device::IA...` → Input Assembler
- `ID3D10Device::VS...` → Vertex Shader
- `ID3D10Device::GS...` → Geometry Shader
- `ID3D10Device::SO...` → Stream Output
- `ID3D10Device::RS...` → Rasterizer Stage
- `ID3D10Device::PS...` → Pixel Shader
- `ID3D10Device::OM...` → Output Merger

Copyright © NVIDIA Corporation 2005

Забудьте про CapsBits!

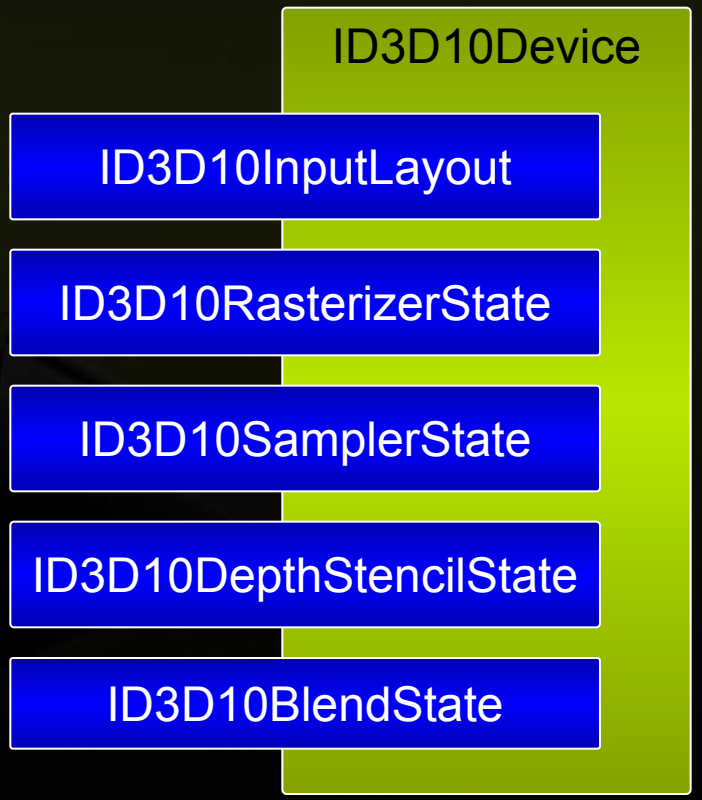
- Вся функциональность *всегда* присутствует
- Большинство форматов *всегда* поддерживается
  - есть исключения блендинг RGB32F
- Поддержка форматов проверяется вызовом `ID3D10Device::CheckFormatSupport`



Copyright © NVIDIA Corporation 2005

Настройка конвейера D3D10

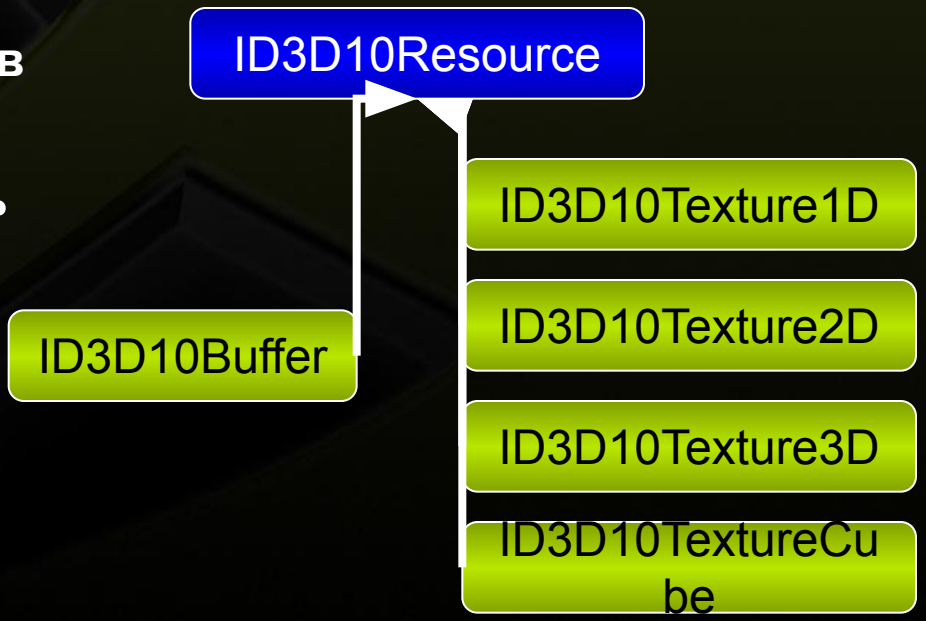
- Настройки конвейера сгруппированы в стейт-объекты
  - Больше нет `SetRenderState(…)`
- Драйвер может эффективно кэшировать состояния конвейера
- 4096 стейт-объектов каждого типа



Copyright © NVIDIA Corporation 2005

Ресурсы D3D10

- Буферы
  - ~Последовательный доступ
  - Элементы могут быть разного типа/размера
- Текстуры
  - Произвольный доступ
  - Состоят из суб-ресурсов
- Тип элементов указывать необязательно
- Виртуализированы OS



Copyright © NVIDIA Corporation 2005

Типы доступа к ресурсам

- `D3D10_USAGE_DEFAULT`
  - Обновляется редко, маппинг невозможен
- `D3D10_USAGE_DYNAMIC`
  - Обновляется часто, CPU имеет прямой доступ
- `D3D10_USAGE_IMMUTABLE`
  - Никогда не обновляется
- `D3D10_USAGE_STAGING`
  - Используется для получения данных из GPU

Copyright © NVIDIA Corporation 2005

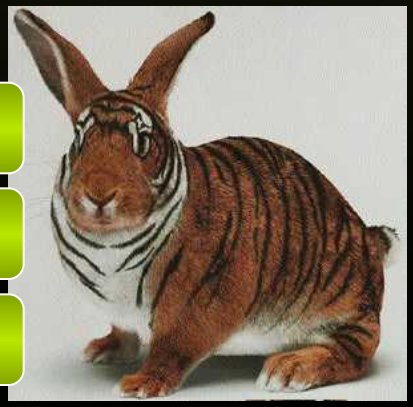
Представления (view)

- Позволяют по-разному интерпретировать данные в ресурсах
  - Рендеринг в VB/IB
  - Рендеринг в константы шейдера
  - Использование stencil-буфера как текстуры
- Позволяют получить доступ к суб-ресурсам
  - Рендеринг в отдельные mip-уровни

ID3D10ShaderResourceView

ID3D10RenderTargetView

ID3D10DepthStencilView



Copyright © NVIDIA Corporation 2005

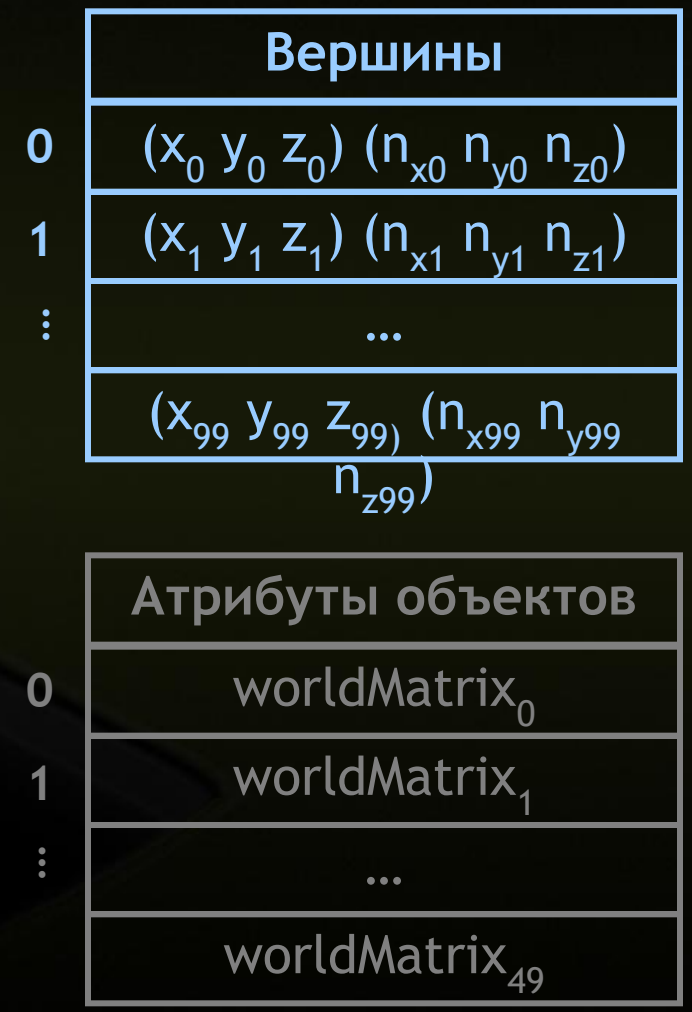
Методы Draw() в D3D10

- `Draw(…)`
- `DrawInstanced(…)`
- `DrawIndexed(…)`
- `DrawIndexedInstanced(…)`
- `DrawAuto(…)`
  - Используется вместе с StreamOut

Copyright © NVIDIA Corporation 2005

Инстансинг в D3D10

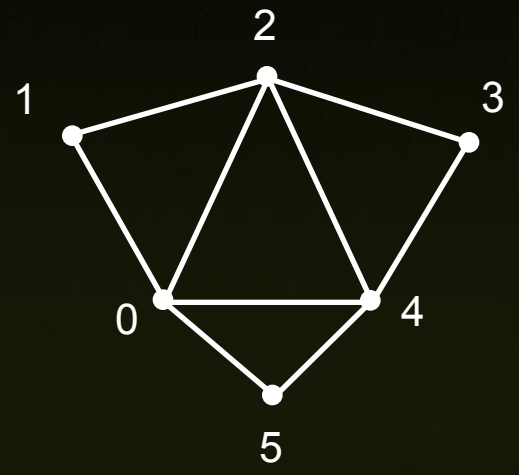
- Полноценная поддержка в API
  - Рендеринг без инстансинга – просто частный случай
- Другие возможности D3D10 делают инстансинг еще более полезным
  - Геометрические шейдеры
  - Массивы текстур
  - Stream Out
  - Instance ID



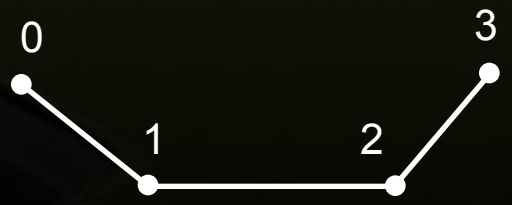
Copyright © NVIDIA Corporation 2005

Геометрический шейдер (GS)

- Программируемость на этапе “сборки примитивов”
- Можно создавать новые примитивы!
- Имеет доступ к соседним примитивам
- Может использовать семантику `SV_PrimitiveID`



Triangles with adjacency



Lines with adjacency

Copyright © NVIDIA Corporation 2005

Геометрический шейдер (GS)

```
[MaxVertexCount(10)]
void GS_Simple(lineadj VSOut In[4], inout
    TriangleStream<GSOut> Stream)
{
    GSOut v;
    …
    Stream.Append(v);
    …
    Stream.Append(v);
    …
    Stream.RestartStrip();
    …
    Stream.Append(v);
}
```

Copyright © NVIDIA Corporation 2005

Примеры использования GS

- Генерирование shadow volume “на лету”
  - Определяем ребра силуэта, генерируем грани
- Motion Blur
  - Считаем локальные скорости, генерируем дополнительную геометрию
- Изо-поверхности на GPU
  - GDC’06, “Practical Metaballs and Isosurfaces”

Copyright © NVIDIA Corporation 2005

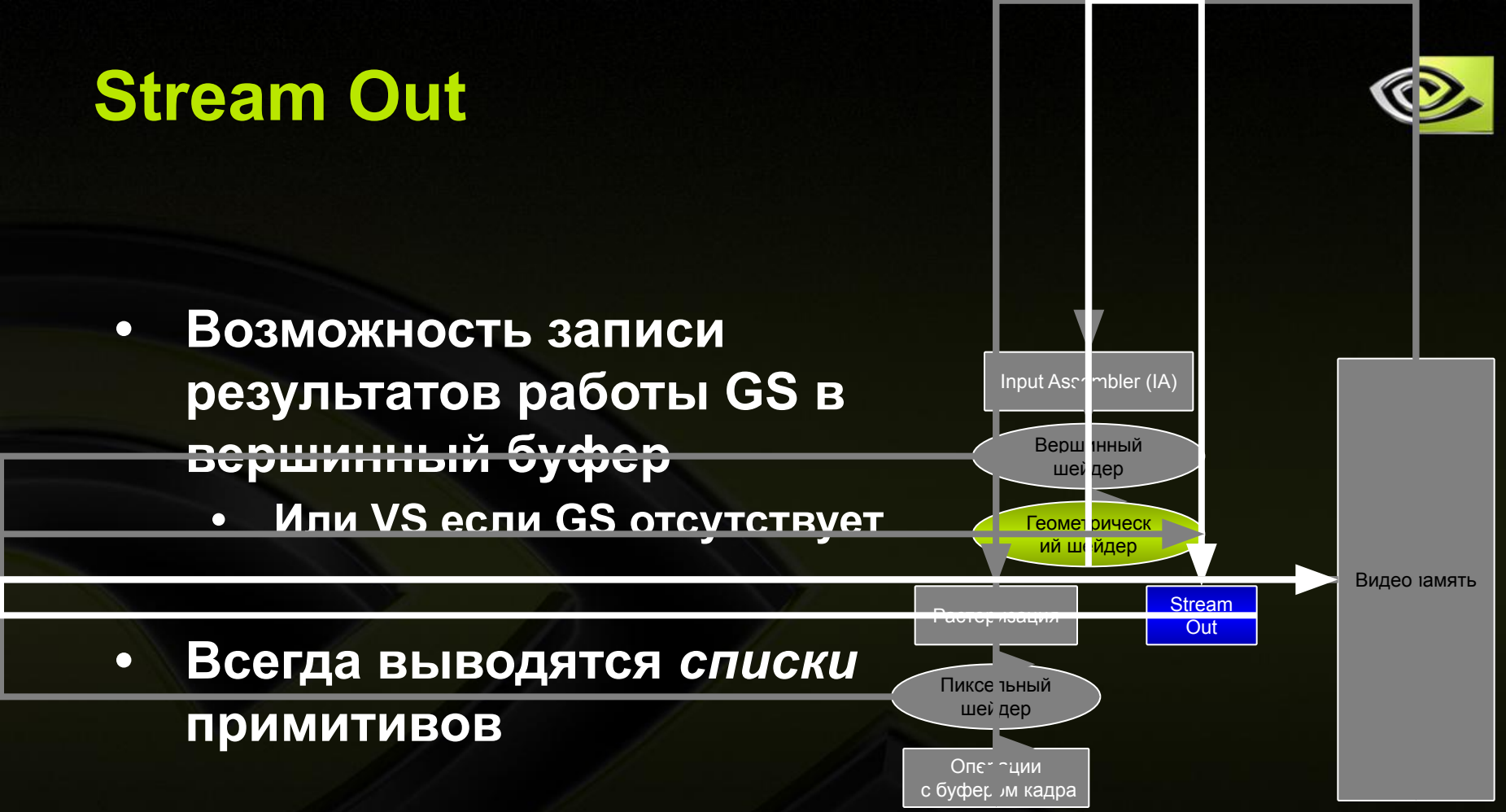
Геометрический шейдер (GS)

- Количество выходных атрибутов ограничено
  - Максимум - 1024 скалярных атрибута в D3D10
- Не используйте для тесселяции!
- Старайтесь не использовать большие значения `MaxVertexCount`

Copyright © NVIDIA Corporation 2005

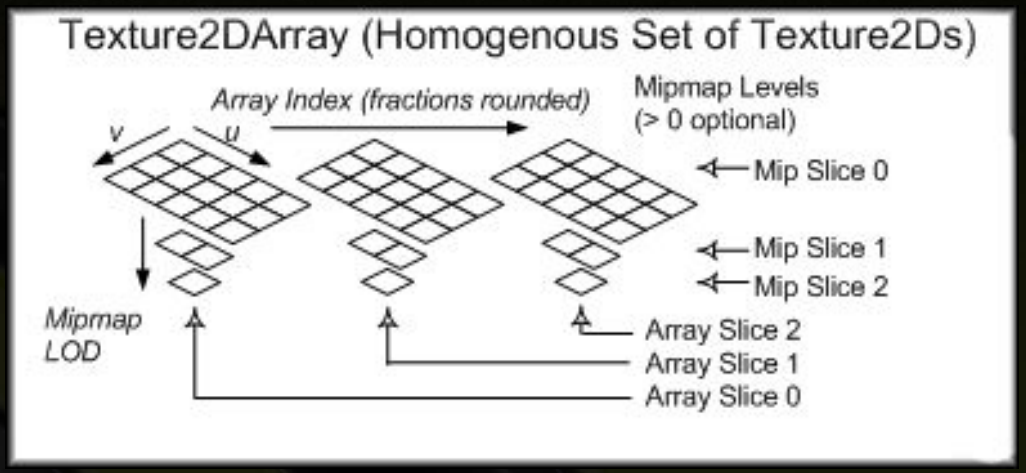
Stream Out

- Возможность записи результатов работы GS в вершинный буфер
  - Или VS если GS отсутствует
- Всегда выводятся *списки* примитивов
- Дает возможность повторно использовать результаты работы GS/VS



Copyright © NVIDIA Corporation 2005

Массивы текстур



- Могут быть динамически индексированы из шейдеров
- Текстуры для инстанцируемых объектов
  - `SV_InstanceID` можно использовать в качестве индекса

Copyright © NVIDIA Corporation 2005

Рендеринг в массив текстур

- Массивы текстур могут использоваться как RT через `ID3D10RenderTargetView`
- В GS переменная с семантикой `SV_RenderTargetArrayIndex` определяет индекс render-текстуры в массиве
  - Можно использовать для рендеринга в cubemap в одном прохоже
- Не то же самое, что MRT!
  - Каждый примитив из GS проходит растеризацию и PS

Copyright © NVIDIA Corporation 2005

Прощай, Fixed Function!

- Нет тумана...
  - ...Эмулируем в VS/PS
- Нет point-спрайтов...
  - ...Используйте GS для создания геометрии
- Нет плоскостей отсечения...
  - ...Используйте семантику `SV_ClipDistance[n]` в VS/GS
- Нет альфа теста!
  - ...Используйте инструкцию `clip()` в пиксельном шейдере
- См. пример FixedFuncEMU в DX10 SDK

Copyright © NVIDIA Corporation 2005

Предикативный рендеринг

- Условное выполнение вызовов Draw()
  - *Без* остановки конвейера
- ... по результату occlusion query
  - Эффективный occlusion culling
- ... при переполнении буфера stream out

Copyright © NVIDIA Corporation 2005

Предикативный рендеринг

```
m_pPredicateQuery->Begin();
//…
// Здесь рисуем простую геометрию
//…
m_pPredicateQuery->End(NULL);

pD3D10Device->SetPredication(m_pPredicateQuery, FALSE);
//…
// Здесь рисуем сложную геометрию
//…

// отключаем предикативный рендеринг
pD3D10Device->SetPredication(NULL, FALSE);
```

Copyright © NVIDIA Corporation 2005

Заключение

- DX10 – большой шаг вперед в функциональности и гибкости

- Начинайте изучать DX10 уже сегодня!
  - Документация/примеры в февральском DX9 SDK

Copyright © NVIDIA Corporation 2005

Наши следующие лекции

16:00 – 16:50 **Инструменты для разработчиков от NVIDIA**
Raul Aguaviva/Филипп Герасимов

17:00 – 18:00 **Расчет физики на GPU**
Simon Green

Copyright © NVIDIA Corporation 2005

Есть идеи?

- Пишите - yuralsky@nvidia.com !

Copyright © NVIDIA Corporation 2005