

고급 ESRAM 12 샘플

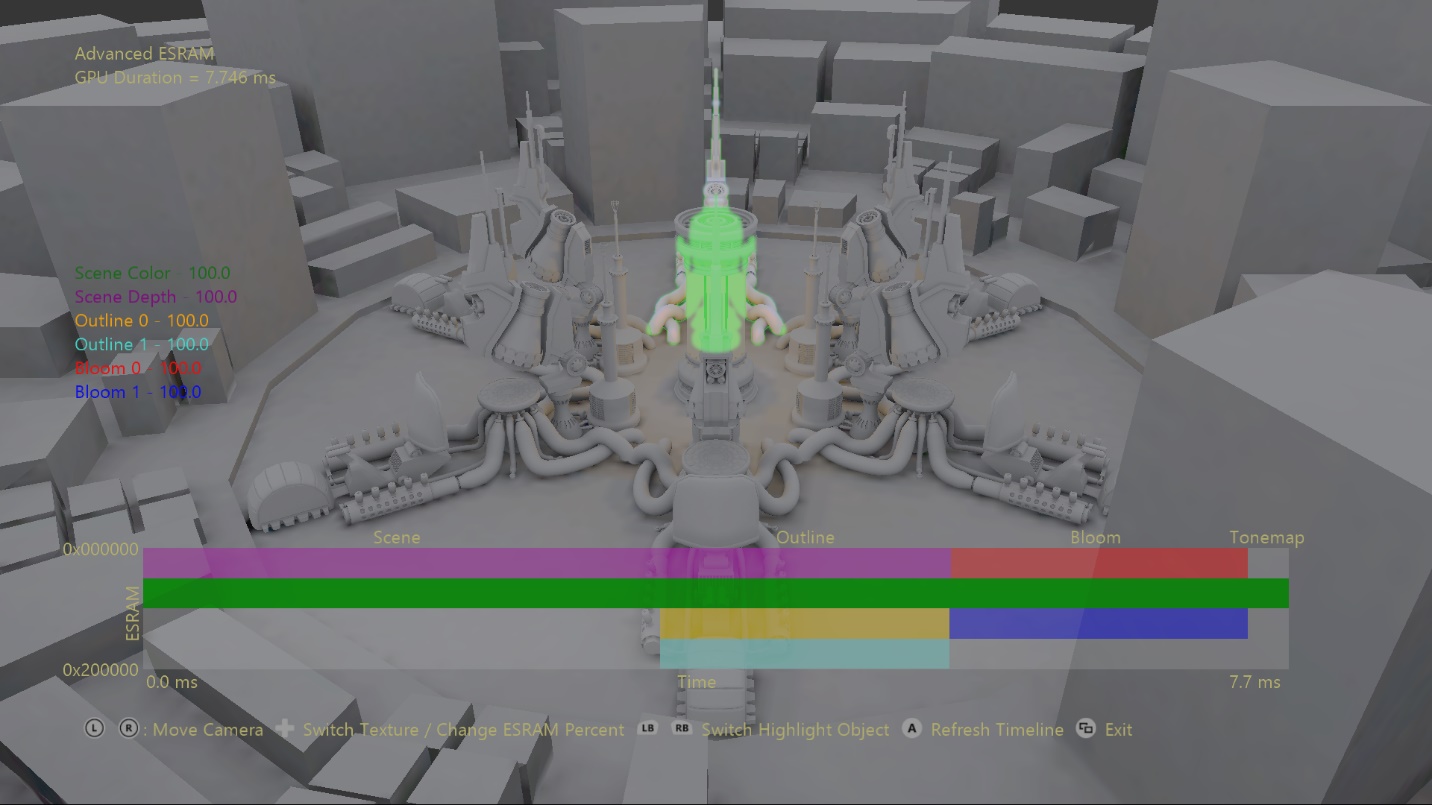
*\*이 샘플은 2016년 10월 Xbox One XDK와 호환됩니다.*

# 설명

이 샘플은 고급 DirectX 12.x 메모리 기능을 사용하여 D3D 리소스용 메모리를 효과적으로 별칭으로 지정합니다. 이 샘플의 핵심에서 API는 ID3D12CommandQueue의 'CopyPageMappingsX'및 'CopyPageMappingsBatchX'입니다. 이 기능들은 CPU 페이지 테이블 엔트리들을 GPU 타임 라인의 GPU에 복사할 수 있게 하여, 완전 가상 D3D 리소스들을 즉시 메모리 페이지에 매핑할 수 있게 합니다.

이 기능을 사용하여 샘플은 ESRAM 및 DRAM을 64KB 페이지 단위로 단계화하는 일시적인 리소스 할당기를 구현합니다. 이것은 ESRAM의 모든 기능을 활용하는 데 사용되는 프레임의 GPU 메모리 사용을 최적으로 유지합니다. 이 인터페이스는 XG 메모리 라이브러리의 XGMemoryLayout 페이지 매핑 함수의 지적 측면을 반영합니다.

알림: Xbox One X에는 더 높은 대역폭의 DRAM을 사용하는 대신 ESRAM이 없습니다. 이 콘솔에서 샘플은 모든 ESRAM 옵션과 시각화를 사용하지 않고 장면을 렌더링합니다.



# 샘플 사용하기

샘플의 주요 기능을 통해 일시적인 텍스처 리소스가 할당되는 위치를 조작할 수 있습니다. 프레임에 사용된 리소스는 장면 색상, 장면 깊이, 윤곽선 2개 및 블룸 2개를 위한 텍스처입니다. 리소스의 ESRAM 구성은 리소스 메모리의 백분율로 맨 왼쪽에 표시됩니다. ESRAM의 레이아웃을 보여주는 시각화는 각 텍스처의 ESRAM 및 DRAM 구성에 대한 변경 사항을 즉시 시각적으로 피드백합니다. 각 리소스의 ESRAM 풋 프린트는 Y축을 따라 표시되며 수명은 X축을 따라 표시됩니다. 시간 축을 따라 사용된 GPU 타이밍은 버튼을 눌러 새로 고칠 수 있습니다.

## 컨트롤

|  |  |
| --- | --- |
| 액션 | 게임 패드 |
| 카메라를 원점을 향해/원위치로 움직입니다. | 왼쪽 엄지 손가락 위/아래 |
| 궤도 카메라 | 오른쪽 엄지 스틱 |
| 카메라 리셋 | 오른쪽 엄지 스틱 (클릭) |
| 사이클 임시 텍스처 | D 패드 왼쪽/오른쪽 |
| ESRAM 백분율 변경 | D- 패드 위/아래 |
| 강조 표시된 개체 순환 | 왼쪽/오른쪽 범퍼 |
| 타임 라인 새로 고침 | A 버튼 |
| 종료 | 보기 버튼 |

# 구현 정보

ESRAM 및 DRAM에서 할당된 64KB 메모리 페이지 (페이지 풀)의 블록을 매핑하기 위해 큰 가상 주소 공간 (VAS)이 생성됩니다. 페이지 풀은 ID3D12Device의 'RegisterPagePoolX'기능을 사용하여 DirectX12에 등록됩니다. 더 이상 사용하지 않을 때는 'UnregisterPagePoolX'로 등록 해제해야 합니다. 이 매핑은 GPU 페이지 테이블에 직접 복사하기 위해 CPU 페이지 테이블 엔트리를 준비하는 역할을 합니다.

ID3D12CommandQueue의 'CopyPageMappingsX'또는 'CopyPageMappingsBatchX'기능을 사용하면 이러한 페이지 풀 내의 페이지 범위를 GPU 타임 라인의 지정된 GPU 가상 주소에 매핑할 수 있습니다. 이를 통해 완전 가상 D3D 리소스를 64KB 물리적 페이지에 유연하게 매핑할 수 있습니다. 이 기능을 사용하면 리소스 간에 메모리 앨리어싱을 간단하게 수행할 수 있습니다.

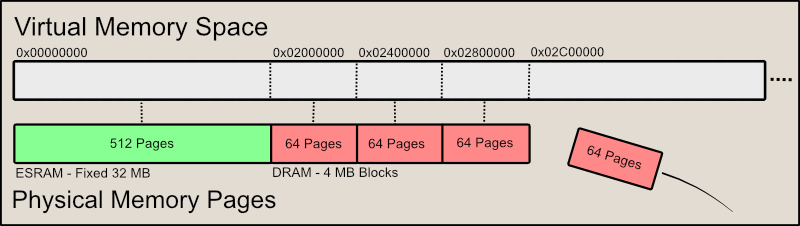


그림 1: 샘플에 사용된 메모리 매핑 패러다임 ESRAM (사용 가능한 경우)은 VAS의 처음 32MB에 매핑되는 반면, 필요에 따라 4MB DRAM 페이지 풀이 추가됩니다.

페이지 블록의 생성 및 관리는 PageAllocator 클래스에 의해 수행됩니다. 가상 주소 범위가 제공되며 할당자는 페이지 풀을 필요에 따라 순차적으로 이 범위에 매핑합니다. 그런 다음 페이지 풀이 'RegisterPagePoolX'를 사용하여 DirectX12에 등록됩니다. 할당자 페이지의 사용량은 완전히 추적됩니다. 처음부터 끝까지 페이지를 할당하고 페이지가 다시 릴리스될 때 페이지를 바꿉니다.

TransientCache는 완전 가상 D3D 리소스를 관리합니다. 이들은 필요에 따라 생성되지만 일반적인 자원을 재작성하는 불필요한 오버 헤드를 피하기 위해 캐시됩니다. 이러한 리소스를 캐시하는 메모리 오버 헤드는 가상 주소 공간만 할당하기 때문에 사실상 0입니다. 각 리소스는 프레임 당 한 번만 할당할 수 있습니다.

이 TransientAllocator 클래스는 페이지 할당자 및 임시 캐시를 사용하여 사용자에게 리소스 요청을 수행합니다. 자원이 요청되면 TransientCache에서 인스턴스를 가져옵니다. 그런 다음 PageAllocators에서 필요한 페이지 수를 할당하고 토큰을 구문 분석하여 ESRAM 또는 DRAM 중 어떤 것을 페이지 단위로 사용할지 결정합니다. 그러면 적절한 구조가 생성되어 나중에 'CopyPageMappingsBatchX', 즉 D3D12XBOX\_PAGE\_MAPPING\_BATCH 및 D3D12XBOX\_PAGE\_MAPPING\_RANGE 구조의 벡터에 제공됩니다.

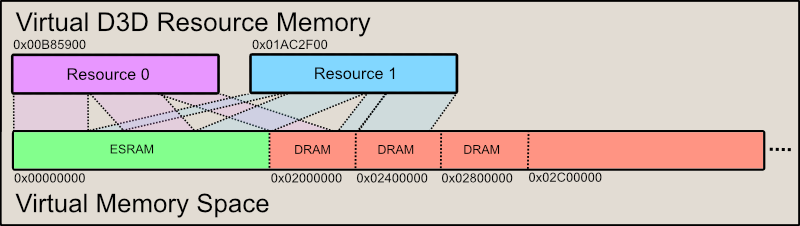


그림 2: 가상 D3D 리소스는 메모리 요구 사항을 충족하기 위해 페이지 풀 내의 페이지 범위에 매핑됩니다. 이러한 매핑은 CopyPageMappingsX 및 CopyPageMappingsBatchX 호출의 결과입니다. 시각적인 단순성을 위해 이 시각화에는 2가지 리소스만 사용되었으며 메모리에는 별칭이 지정되지 않았습니다. 그러나 메모리 앨리어싱은 이 기술의 예상되는 이점입니다.

메모리 별칭 때문에 TransientAllocator는 필요한 경우 셰이더 및 캐시 플러시를 수행합니다. DirectX12에서는 플러시가 리소스 장벽의 일부로 삽입됩니다. 메모리 앨리어싱을 수행하기 위해이 시스템을 우회했기 때문에 수동으로 직접 플러시를 삽입해야 합니다. TransientAllocator는 리소스의 관련 뷰를 검토하여 플러시되어야 하는 셰이더 스테이지와 캐시를 결정합니다.

마지막으로 할당된 임시 리소스를 사용하는 명령 목록이 명령 대기열에 제출되기 전에 TransientAllocator에서 'Finalize'를 호출하여 리소스 매핑을 완료해야 합니다. 이 시점에서 'CopyPageMappingsBatchX' 호출은 명령 대기열에 배치되며, 명령 대기열은 후속 명령 목록에 사용될 자원의 메모리 매핑을 설정합니다.

# 업데이트 기록

2018년 8월 6일 - 샘플 생성.

# 개인정보처리방침

샘플을 컴파일하고 실행할 때 샘플 실행 파일의 이름이 Microsoft로 보내져 샘플 사용을 추적 할 수 있습니다. 이 데이터 수집을 거부하려면 Main.cpp에서 "샘플 사용 텔레메트리"라는 코드 블록을 제거하면 됩니다.

Microsoft의 개인 정보 취급 방침에 대한 일반적인 내용은 [Microsoft 개인 정보 취급 방침](https://privacy.microsoft.com/ko-kr/privacystatement/)을 참조하십시오.