

간단한 DMA 압축 해제 샘플

*\*이 샘플은 2017년 6월 Xbox One XDK와 호환됩니다.*

# 설명



# 이 샘플에서는 몇 가지 시나리오에서 하드웨어 LZ 압축 및 압축 해제 기능을 사용하는 방법을 보여줍니다.

# Xbox One GPU를 사용하면 Jpeg 디코딩 뿐만 아니라 DEFLATE 알고리즘을 구현하는 DMA 엔진에 직접 액세스할 수 있습니다 (<http://www.rfc-base.org/rfc-1951.html>). 이는 CPU 집약적인 작업을 오프로드하는 데 사용할 수 있습니다.

# 샘플 사용하기

## 이 샘플에서는 다음 컨트롤을 사용합니다:

|  |  |
| --- | --- |
| 액션 | 게임 패드 |
| 로드된 텍스처 순환 | Dpad 위/아래 |
| 하드웨어/소프트웨어 압축 토글 | A 버튼 |
| 하드웨어/소프트웨어 압축 해제 토글 | B 버튼 |
| 종료 | 보기 버튼 |

# 구현 정보

## 시나리오 1: 런타임 디코딩/인코딩

샘플이 로드되면 샘플의 Media \ Textures \ folder에 있는 DDS 텍스처 파일을 선택합니다. MediaList.txt 파일을 수정하고 다시 배포하여 파일을 추가할 수 있습니다.

로드가 완료되면 Dma 하드웨어 (1KB 창에서 LZ77)로 압축하거나 참조 압축 라이브러리 Zlib 또는 Zopfli를 사용하여 압축 간에 변경할 수 있습니다. 후자의 소프트웨어 압축 옵션을 사용할 때 압축은 4KB 창을 사용하여 압축 해제 엔진이 지원하는 최적의 옵션으로 제한됩니다. 이러한 옵션을 전환하면 사용 가능한 압축 옵션을 쉽게 비교할 수 있습니다. 또한 큰 파일은 하드웨어와의 호환성을 유지하기 위해 4MB보다 약간 작은 조각으로 나뉘어 있습니다.

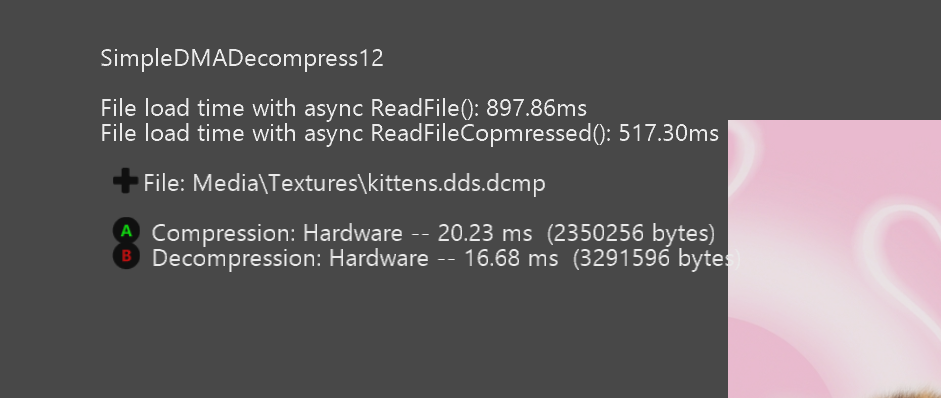
## 시나리오 2: 스트리밍 DMA 압축 풀기

이 시나리오는 대부분의 타이틀에서와 마찬가지로 콘텐츠 제작 파이프라인의 일부로 압축된 디스크에서 스트리밍되는 콘텐츠를 압축 해제합니다. 이 변형에서 모든 압축 해제 작업은 CPU 사이클을 소모하는 대신 전용 하드웨어로 오프로드됩니다.

이를 위해 StreamingDmaCompressionLib 프로젝트는 DMA 하드웨어로 비동기 오프로드하여 효과적으로 I/O 처리량을 높이고 타이틀의 패키지 크기를 줄이는 압축된 블록으로 구성된 간단한 파일 형식을 정의합니다. 사용하기 쉽도록 XboxDmaCompression 도구 (PC 전용)를 사용하면 하드웨어가 지원하는 최적의 방식으로 전체 파일을 압축할 수 있습니다. 또는, 압축된 내용을 더 큰 복합 파일에 포함시키 위해 스트림 레벨에서 압축하도록 라이브러리가 직접 사용될 수 있습니다.

결과 파일(또는 내장된 압축 스트림)은 기본적으로 Win32 ReadFile () API와 API 호환되는 ReadFileCompressed ()를 사용하여 읽습니다. 이는 오버랩된 비동기식 I/O를 위한 것으로, 버퍼링 및 버퍼링되지 않은 읽기를 지원하며 후자의 성능이 향상됩니다.

이 시스템의 처리량 특성을 보여주기 위해 기본 MediaList.txt는 텍스처를 포함하여 20 개의 압축본과 20개의 압축되지 않은 본을 복제하고 initialization \ load 경로에서 한 번에 하나의 배치를 로드합니다. 각 배치의 총 압축되지 않은 바이트는 59.6MB이며 압축되지 않은 로드 (66MB/s)와 압축된 로드 (115MB/s)의 실제 처리량은 크게 다릅니다. DmaCompressionTool과 함께 사용하면 DMA 압축 해제 하드웨어가 타이틀에 적합한지 판단할 수 있습니다. 라이브러리는 완전히 비동기식이므로 기본 Windows와 겹치는 \ async I/O와 유사하게 확장되어 실행 중 요청 수에 따라 증가합니다.



ReadFileCompressed () API를 사용할 때 가장 큰 차이점은 API가 호출될 때 압축 해제된 버퍼의 크기를 알 수 없다는 점입니다. 이 정보를 미리 알기 위해 타이틀을 요구하는 대신, 다른 차단 읽기가 필요하기 때문에 API는 사용자 지정 할당자를 전달하거나 기본 할당자를 사용할 수 있도록 지원하며 파일 헤더가 들어있는 블록이 로드되면 할당됩니다.

ReadFileCompressed와 관련하여 중복 된 I/O의 다른 부분들은 ReadFile과 동일하지만 몇 가지 예외가 있습니다. ReadFileCompressed는 버퍼되지 않은 파일에 대한 비 섹터 정렬 읽기를 처리할 수 있습니다. 내부적으로 pre \ post 버퍼링을 처리합니다. 그러나 ReadFileCompressed는 CPU에 의해 복사되는 메모리가 없어도 dma가 저장소 컨트롤러에서 이동하고 dma가 decompressor를 통해 이동할 수 있도록 4바이트로 읽기를 정렬해야 합니다. StreamingDmaCompressionLib를 사용하여 더 큰 복합 파일 내에서 스트림을 압축하는 경우 압축된 스트림의 첫 번째 바이트가 파일 내에서 4바이트로 정렬되어 있는지 확인해야 합니다.

# 알려진 문제점

* 이 샘플에서는 일시 중지/다시 시작을 제대로 구현하지 않습니다.

# 업데이트 기록

이 샘플의 최초 버전은 2012년에 작성되었습니다. 2018년 9월에 재작성되었습니다.

# 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.