Direct Access to Video Encoding and Decoding

Session 513
David Eldred
Tech Lead, Frameworks with “Video” in their names
Introduction

Hardware encode and decode can improve user experience
Introduction

Hardware encode and decode can improve user experience

- Better performance
Hardware encode and decode can improve user experience

- Better performance
- Increased efficiency
Introduction

Hardware encode and decode can improve user experience

- Better performance
- Increased efficiency
- Extend battery life
What You Will Learn
What You Will Learn

Case Studies
What You Will Learn

Case Studies
1. Displaying an H.264 stream in a layer in your application
What You Will Learn

Case Studies

1. Displaying an H.264 stream in a layer in your application
2. Decoding an H.264 stream and accessing the decoded buffers
What You Will Learn

Case Studies
1. Displaying an H.264 stream in a layer in your application
2. Decoding an H.264 stream and accessing the decoded buffers
3. Compressing a sequence of images into a movie file
What You Will Learn

Case Studies
1. Displaying an H.264 stream in a layer in your application
2. Decoding an H.264 stream and accessing the decoded buffers
3. Compressing a sequence of images into a movie file
4. Compressing a sequence of images into an H.264 stream for the network
What You Will Learn

Case Studies
1. Displaying an H.264 stream in a layer in your application
2. Decoding an H.264 stream and accessing the decoded buffers
3. Compressing a sequence of images into a movie file
4. Compressing a sequence of images into an H.264 stream for the network

Using multi-pass encoding in AVFoundation and Video Toolbox
Media Interfaces Overview
With a focus on video
Media Interfaces Overview
With a focus on video
Media Interfaces Overview
With a focus on video

AVKit
AVFoundation
Media Interfaces Overview
With a focus on video

- AVKit
- AVFoundation
- Video Toolbox
Media Interfaces Overview
With a focus on video

- AVKit
- AVFoundation
- Video Toolbox
- Core Media
- Core Video
Media Interface Focus

AVKit

AVFoundation

Video Toolbox

Core Media

Core Video
Media Interface Focus

AVFoundation
- Decompress direct to display
- Compress directly to file
Media Interface Focus

AVFoundation
- Decompress direct to display
- Compress directly to file

Video Toolbox
- Decompress to CVPixelBuffer
- Compress to CMSampleBuffer
Media Interface Focus

- AVKit
- AVFoundation
- Video Toolbox
Hardware Codec Usage

AVKit

AVFoundation

Video Toolbox
## Hardware Codec Usage

<table>
<thead>
<tr>
<th></th>
<th>iOS</th>
<th>OS X</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVKit</td>
<td>Always</td>
<td>When Available</td>
</tr>
<tr>
<td>AVFoundation</td>
<td>Always</td>
<td>When Available</td>
</tr>
<tr>
<td>Video Toolbox</td>
<td>Always</td>
<td>When Available and Requested</td>
</tr>
</tbody>
</table>
The Cast of Characters

CVPixelBuffer
CVPixelBufferPool
pixelBufferAttributes
CMTime
CMVideoFormatDescription
CMBlockBuffer
CMSampleBuffer
CMClock
CMTimebase
The Cast of Characters

CVPixelBuffer
CVPixelBufferPool
pixelBufferAttributes
CMTime
CMVideoFormatDescription
CMBlockBuffer
CMSampleBuffer
CMClock
CMT imebase
The Cast of Characters

CVPixelBuffer
CVPixelBufferPool
pixelBufferAttributes
CMTime
CMVideoFormatDescription
CMBlockBuffer
CMSampleBuffer
CMClock
CMTimeTypebase
The Cast of Characters

CVPixelBuffer
CVPixelBufferPool
pixelBufferAttributes
CMTime
CMVideoFormatDescription
CMBlockBuffer
CMSampleBuffer
CMClock
CMTimebase

Uncompressed Raster Image Buffer
The Cast of Characters

CVPixelBuffer
CVPixelBufferPool
pixelBufferAttributes
CMTime
CMVideoFormatDescription
CMBlockBuffer
CMSampleBuffer
CMClock
CMT imebase
The Cast of Characters

CVPixelBuffer
CVPixelBufferPool
pixelBufferAttributes
CMTime
CMVideoFormatDescription
CMBlockBuffer
CMSampleBuffer
CMClock
CMTimebase

CFDictionary of requirements;
May Include:
• Width/height
• Pixel format type (e.g., 32BGRA, YCbCr420)
• Compatibility (e.g., OpenGL ES, Core Animation)
The Cast of Characters

CVPixelBuffer
CVPixelBufferPool
pixelBufferAttributes
CMT ime
CMVideoFormatDescription
CMBlockBuffer
CMSampleBuffer
CMClock
CMT imebase

64-bit Time Value (Numerator)
32-bit Time Scale (Denominator)
The Cast of Characters

CVPixelBuffer
CVPixelBufferPool
pixelBufferAttributes
CMTime
CMVideoFormatDescription
CMBlockBuffer
CMSampleBuffer
CMClock
CMTimebase

Width/Height
Format Type—(kCMPixelFormat_32BGRA, kCMVideoCodecType_H264, …)
Extensions—(Pixel Aspect Ratio, Color Space, …)
The Cast of Characters

CVPixelBuffer
CVPixelBufferPool
pixelBufferAttributes
CMTime
CMVideoFormatDescription
CMBlockBuffer
CMSampleBuffer
CMClock
CMTimebase
The Cast of Characters

CVPixelBuffer
CVPixelBufferPool
pixelBufferAttributes
CMTime
CMVideoFormatDescription
CMBlockBuffer
CMSampleBuffer
CMClock
CMTimestamp
The Cast of Characters

CVPixelBuffer
CVPixelBufferPool
pixelBufferAttributes
CMTime
CMVideoFormatDescription
CMBlockBuffer
CMSampleBuffer
CMClock
CMT imebase

CMSampleBuffer

or

CMSampleBuffer

Compressed Video Frame

Uncompressed Raster Image
The Cast of Characters

CVPixelBuffer
CVPixelBufferPool
pixelBufferAttributes
CMTime
CMVideoFormatDescription
CMBlockBuffer
CMSampleBuffer
CMClock
CMTimebase
The Cast of Characters

- CVPixelBuffer
- CVPixelBufferPool
- pixelBufferAttributes
- CMTime
- CMVideoFormatDescription
- CMBlockBuffer
- CMSampleBuffer
- CMClock
- CMTimebase
The Cast of Characters

CVPixelBuffer
CVPixelBufferPool
pixelBufferAttributes
CMTime
CMVideoFormatDescription
CMBlockBuffer
CMSampleBuffer
CMClock
CMTimebase
The Cast of Characters

- CVPixelBuffer
- CVPixelBufferPool
- pixelBufferAttributes
- CMT ime
- CMVideoFormatDescription
- CMBlockBuffer
- CMSampleBuffer
- CMClock
- CMT imebase

Compressed Video Frame

Uncompressed Raster Image
The Cast of Characters

CVPixelBuffer
CVPixelBufferPool
pixelBufferAttributes
CMTime
CMVideoFormatDescription
CMBlockBuffer
CMSampleBuffer
CMClock
CMTimebase
The Cast of Characters

Wraps a source of time
A clock’s time always increases

CMClockGetHostTimeClock() wraps mach_absolute_time()
The Cast of Characters

- CVPixelBuffer
- CVPixelBufferPool
- pixelBufferAttributes
- CMTime
- CMVideoFormatDescription
- CMBlockBuffer
- CMSampleBuffer
- CMClock
- CMT imebase
The Cast of Characters

A controlled view onto a CMClock
The Cast of Characters

A controlled view onto a CMClock

`CMTtimebaseSetTime(timebase, kCMTtimeZero);`
The Cast of Characters

A controlled view onto a CMClock

CMTimebaseSetTime(timebase, kCMTimeZero);
CMTimebaseSetRate(timebase, 1.0);
Case One
Displaying video from a network stream
Case One Overview

Network
Case One Overview

Network → Compressed Video Samples

H.264 | H.264 | H.264

Network

Compressed Video Samples
Case One Overview

Network

→ H.264 → H.264 → H.264 → AVSampleBufferDisplayLayer → Phone

Compressed Video Samples
AVSampleBufferDisplayLayer
A closer look
AVSampleBufferDisplayLayer

A closer look
AVSampleBufferDisplayLayer

A closer look
AVSampleBufferDisplayLayer
A closer look

CVPixelBuffers

H.264

CMSampleBuffers

AVSampleBufferDisplayLayer

Video Decoder

CVPixelBuffers
AVSampleBufferDisplayLayer
A closer look

CMSampleBuffers
H.264  H.264  H.264

AVSampleBufferDisplayLayer
Video Decoder

CVPixelBuffers

iPhone
AVSampleBufferDisplayLayer Input

Elementary Stream

H.264  H.264  H.264
AVSampleBufferDisplayLayer Input

Elementary Stream

H.264  H.264  H.264

CMSampleBuffers

H.264  H.264  H.264
AVSampleBufferDisplayLayer Input

Elementary Stream

processing

CMSampleBuffers

H.264 H.264 H.264

H.264 H.264 H.264
H.264 Syntax

Elementary Stream  MPEG-4
H.264 Syntax

Elementary Stream

MPEG-4
H.264 Syntax
Network Abstraction Layer (NAL)
H.264 Syntax

Network Abstraction Layer (NAL)

H.264 stream consists of a sequence of NAL Units (NALUs)
H.264 Syntax

Network Abstraction Layer (NAL)

H.264 stream consists of a sequence of NAL Units (NALUs)

NALUs may contain
H.264 Syntax

Network Abstraction Layer (NAL)

H.264 stream consists of a sequence of NAL Units (NALUs)

NAL Units may contain

• Video frame (or slice of video frame)
H.264 Syntax

Network Abstraction Layer (NAL)

H.264 stream consists of a sequence of NAL Units (NALUs)

NAL Units may contain

- Video frame (or slice of video frame)
H.264 Syntax

Network Abstraction Layer (NAL)

H.264 stream consists of a sequence of NAL Units (NALUs)

NAL Units may contain
- Video frame (or slice of video frame)
H.264 Syntax

Network Abstraction Layer (NAL)

H.264 stream consists of a sequence of NAL Units (NALUs)

NAL Units may contain

• Video frame (or slice of video frame)
• H.264 parameter sets
  - Sequence Parameter Set (SPS) and Picture Parameter Set (PPS)
H.264 Syntax
Parameter sets: SPS and PPS

Elementary Stream
MPEG-4
H.264 Syntax

Parameter sets: SPS and PPS

Elementary Stream

- SPS
- PPS
- I frame
- P frame
- B frame
- B frame...

Parameter Sets in Stream
H.264 Syntax
Parameter sets: SPS and PPS

Elementary Stream
- SPS
- PPS
- I frame
- P frame
- B frame
- B frame
- ...

Parameter Sets in Stream

MPEG-4
- I frame
- P frame
- B frame
- B frame
- ...

Parameter Sets in Format Description

CMVideoFormatDescription
| Elementary Stream | MPEG-4 |
H.264 Syntax
Conversion

Elementary Stream

- SPS
- PPS

MPEG-4
H.264 Syntax Conversion

Elementary Stream

- SPS
- PPS

MPEG-4

- SPS
- PPS

CMVideoFormatDescription
H.264 Syntax Conversion

Elementary Stream

SPS
PPS

MPEG-4

SPS
PPS

CMVideoFormatDescription

CMVideoFormatDescriptionCreateFromH264ParameterSets
H.264 Syntax

NAL Unit headers

Elementary Stream     MPEG-4
H.264 Syntax

NAL Unit headers

Elementary Stream

00 00 01

NAL Unit

3- or 4-Byte Header:
Start Code
H.264 Syntax

NAL Unit headers

Elementary Stream

00 00 01

NAL Unit

3- or 4-Byte Header:
Start Code

MPEG-4

00 00 80 00

NAL Unit
(32768 bytes)

4-Byte Header:
Length
Building a CMSampleBuffer
NAL Unit conversion
Building a CMSampleBuffer

NAL Unit conversion

00 00 80 00

NAL Unit
(32768 bytes)
Building a CMSampleBuffer

NAL Unit conversion

CMBlockBuffer

00 00 80 00

NAL Unit
(32768 bytes)
Building a CMSampleBuffer

NAL Unit conversion

CMBlockBuffer

00 00 80 00

NAL Unit
(32768 bytes)

SPS

PPS

CMVideoFormatDesc
Building a CMSampleBuffer

NAL Unit conversion
Building a CMSampleBuffer

NAL Unit conversion

CMBlockBuffer
00 00 80 00

NAL Unit (32768 bytes)

+ CMVideoFormatDesc

SPS

PPS

+ CMT ime

= CMSampleBuffer

CMT ime

CMVideoFormatDesc

CMBlockBuffer
Building a CMSampleBuffer

NAL Unit conversion

CMSampleBufferCreate

CMBlockBuffer
00 00 80 00
NAL Unit (32768 bytes)

CMVideoFormatDesc
SPS
PPS

CMTime

CMSampleBufferCreate

CMSampleBuffer
CMTime
CMVideoFormatDesc
CMBlockBuffer
AVSampleBufferDisplayLayer and Time

AVSampleBufferDisplayLayer

Video Decoder

Video Decoder
CMSampleBuffers
H.264 5.50s
H.264 5.40s
H.264 5.30s

AVSampleBufferDisplayLayer

Video Decoder

Video Decoder

AVSampleBufferDisplayLayer

Video
Decoder
AVSampleBufferDisplayLayer and Time

CMSampleBuffers

H.264 5.50s
H.264 5.40s
H.264 5.30s

Video Decoder

AVSampleBufferDisplayLayer

CVPixelBuffers

5.20s
5.10s
5.00s
AVSampleBufferDisplayLayer and Time

CMSampleBuffers
H.264 5.50s
H.264 5.40s
H.264 5.30s

AVSampleBufferDisplayLayer

Video Decoder

CVPixelBuffers
5.20s 5.10s 5.00s

hostTime
AVSampleBufferDisplayLayer and Time

```c
sbDisplayLayer.controlTimebase = CMTimebaseCreateWithMasterClock(CMClockGetHostTimeClock());
CMTimebaseSetTime(sbDisplayLayer.controlTimebase, CMTimeMake(5, 1));
CMTimebaseSetRate(sbDisplayLayer.controlTimebase, 1.0);
```
Feeding AVSampleBufferDisplayLayer

Two scenarios
Feeding AVSampleBufferDisplayLayer

Two scenarios

Periodic Source
Feeding AVSampleBufferDisplayLayer

Two scenarios

Periodic Source

Unconstrained Source
Feeding AVSampleBufferDisplayLayer

Periodic source
Frame arrival corresponds to display frequency

Enqueue with:

```
[sbDisplayLayer enqueueSampleBuffer:sbuf];
```
Feeding AVSampleBufferDisplayLayer

Unconstrained source
Feeding AVSampleBufferDisplayLayer

Unconstrained source

AVSampleBufferDisplayLayer throttles input:

```
[sbDisplayLayer requestMediaDataWhenReadyOnQueue:dispatchQueue usingBlock:^{
  while ([sbDisplayLayer isReadyForMoreMediaData]) {
    CMSampleBuffer sbuf = copyNextSBuf();
    [sbDisplayLayer enqueueSampleBuffer:sbuf];
    CFRelease(sbuf);
  }
}];
```
Feeding AVSampleBufferDisplayLayer

AVSampleBufferDisplayLayer throttles input:

```objective-c
[sbDisplayLayer requestMediaDataWhenReadyOnQueue:dispatchQueue usingBlock:^{
    while ([sbDisplayLayer isReadyForMoreMediaData]) {
        CMSampleBuffer sbuf = copyNextSBuf();
        [sbDisplayLayer enqueueSampleBuffer:sbuf];
        CFRelease(sbuf);
    }
}];
```
Feeding AVSampleBufferDisplayLayer

Unconstrained source

AVSampleBufferDisplayLayer throttles input:

```
[sbDisplayLayer requestMediaDataWhenReadyOnQueue:dispatchQueue usingBlock:^{
    while ([sbDisplayLayer isReadyForMoreMediaData]) {
        CMSampleBuffer sbuf = copyNextSBuf();
        [sbDisplayLayer enqueueSampleBuffer:sbuf];
        CFRelease(sbuf);
    }
}}];
```
Feeding AVSampleBufferDisplayLayer
Unconstrained source

AVSampleBufferDisplayLayer throttles input:

```
[sbDisplayLayer requestMediaDataWhenReadyOnQueue:dispatchQueue usingBlock:^{
    while ([sbDisplayLayer isReadyForMoreMediaData]) {
        CMSampleBuffer sbuf = copyNextSBuf();
        [sbDisplayLayer enqueueSampleBuffer:sbuf];
        CFRelease(sbuf);
    }
}];
```
AVSampleBufferDisplayLayer Summary
AVSampleBufferDisplayLayer Summary

Creation of AVSampleBufferDisplayLayer
AVSampleBufferDisplayLayer Summary

Creation of AVSampleBufferDisplayLayer
Converting an H.264 elementary stream into CMSampleBuffers
Creation of AVSampleBufferDisplayLayer
Converting an H.264 elementary stream into CMSampleBuffers
Providing CMSampleBuffers to AVSampleBufferDisplayLayer
AVSampleBufferDisplayLayer Summary

Creation of AVSampleBufferDisplayLayer
Converting an H.264 elementary stream into CMSampleBuffers
Providing CMSampleBuffers to AVSampleBufferDisplayLayer
Using a custom CMTimebase with AVSampleBufferDisplayLayer
Case Two
Getting CVPixelBuffers from a compressed stream
AVSampleBufferDisplayLayer
AVSampleBufferDisplayLayer

CMSampleBuffers
H.264  H.264  H.264

Video Decoder

CVPixelBuffers

AVSampleBufferDisplayLayer

→

→

iPhone
VTDecompressionSession

Getting access to the decoder
VTDecompressionSession

Getting access to the decoder
VTDecompressionSession
Getting access to the decoder

CMSampleBuffers
H.264 -> H.264 -> H.264

VTDecompressionSession
Video Decoder
VTDecompressionSession

Getting access to the decoder

CMSampleBuffers — H.264 — H.264 — H.264 — VTDecompressionSession — Video Decoder — CVPixelBuffers
VTDecompressionSession

Getting access to the decoder

CMSampleBuffers

H.264 → H.264 → H.264

VTDecompression Session

Video Decoder

OutputCallback

CVPixelBuffers
Creating a VTDecompressionSession

What you need
Creating a VT DecompressionSession

What you need

• Description of source data—CMVideoFormatDescription
Creating a VTDecompressionSession

What you need

• Description of source data—CMVideoFormatDescription
• Requirements for output buffers—pixelBufferAttributes
Creating a VTDecompressionSession

What you need

• Description of source data—CMVideoFormatDescription
• Requirements for output buffers—pixelBufferAttributes
• A VTDecompressionOutputCallback
Requirements for Output CVPixelBuffers
Creating a pixelBufferAttributes dictionary
Requirements for Output CVPixelBuffers
Creating a pixelBufferAttributes dictionary

Example: OpenGL ES render pipeline
Requirements for Output CVPixelBuffers
Creating a pixelBufferAttributes dictionary

Example: OpenGL ES render pipeline

To require OpenGL ES compatibility—

```swift
NSDictionary *destinationImageBufferAttributes = 
    [NSDictionary dictionaryWithObjectsAndKeys:
        [NSNumber numberWithBool:YES], (id)kCVPixelBufferOpenGLESCompatibilityKey,
        nil];
```
Requirements for Output CVPixelBuffers

Creating a pixelBufferAttributes dictionary

Example: OpenGL ES render pipeline

To require OpenGL ES compatibility—

```swift
NSDictionary *destinationImageBufferAttributes =
    [NSDictionary dictionaryWithObjectsAndKeys:
        [NSNumber numberWithBool:YES], (id)kCVPixelBufferOpenGLESCompatibilityKey,
        nil];
```
Optimizing Output

Do not over specify
Optimizing Output

Do not over specify

kCVPixelBufferOpenGLESCompatibilityKey requested
Optimizing Output

Do not over specify

kCVPixelBufferOpenGLESCompatibilityKey requested
Optimizing Output

Do not over specify

kCVPixelBufferOpenGLESCompatibilityKey requested

kCVPixelBufferOpenGLESCompatibilityKey and ‘BGRA’ pixel format required
Optimizing Output
Do not over specify

kCVPixelBufferOpenGLESCompatibilityKey requested

kCVPixelBufferOpenGLESCompatibilityKey and ‘BGRA’ pixel format required
Optimizing Output
Do not over specify

kCVPixelBufferOpenGLESCompatibilityKey requested

VTDecompressionSession
Decoder
YUV

OpenGL ES

kCVPixelBufferOpenGLESCompatibilityKey and ‘BGRA’ pixel format required

VTDecompressionSession
Decoder
YUV
BGRA

OpenGL ES

BGRA
Optimizing Output
Do not over specify

kCVPixelBufferOpenGLESCompatibilityKey requested

kCVPixelBufferOpenGLESCompatibilityKey and ‘BGRA’ pixel format required
VTDecompressionOutputCallback
VTDecompressionOutputCallback

VTDecompressionSession → OutputCallback
VTDecompressionOutputCallback receives
VTDecompressionOutputCallback receives

- Output CVPixelBuffer
VTDecompressionOutputCallback receives

- Output CVPixelBuffer
- Presentation time stamp
VTDecompressionOutputCallback receives

- Output CVPixelBuffer
- Presentation time stamp
- Decompression error codes
VTDecompressionOutputCallback receives

- Output CVPixelBuffer
- Presentation time stamp
- Decompression error codes
- Dropped frames
Feeding VTDecompressionSession

CMSampleBuffers

H.264  H.264  H.264  \rightarrow  VTDecompressionSession
Feeding VTDecompressionSession

```c
err = VTDecompressionSessionDecodeFrame( session, sbuf, inFlags, refCon, &outFlags );
```
Feeding VTDecompressionSession

err = VTDecompressionSessionDecodeFrame( session, sbuf, inFlags, refCon, &outFlags );

Wants CMSampleBuffers
Feeding VTDecompressionSession

err = VTDecompressionSessionDecodeFrame( session, sbuf, inFlags, refCon, &outFlags );

Wants CMSampleBuffers
Decode order
Feeding VTDecompressionSession

```
err = VTDecompressionSessionDecodeFrame( session, sbuf, inFlags, refCon, &outFlags );
```

Wants CMSampleBuffers
Decode order
Synchronous by default
Feeding VTDecompressionSession

err = VTDecompressionSessionDecodeFrame( session, sbuf, inFlags, refCon, &outFlags );

Wants CMSampleBuffers
Decode order
Synchronous by default
Set kVTDecodeFrame_EnableAsynchronousDecompression for async
Async Decompression
Async Decompression

Decoder blocks when full—Decoder back pressure
Async Decompression

Decoder blocks when full—Decoder back pressure

Finish async frames with VTDecompressionSessionWaitForAsynchronousFrames
Changing CMVideoFormatDescription
Changing CMVideoFormatDescription

- I frame
- P frame
- B frame
- B frame
- B frame
- ...
Changing CMVideoFormatDescription

```
err = VTDecompressionSessionCreate( kCFAllocatorDefault, formatDesc1, ... ,
&session );
```
err = VTDecompressionSessionCreate( kCFAllocatorDefault, formatDesc1, ... , &session );
Changing CMVideoFormatDescription

err = VTDecompressionSessionCreate( kCFAllocatorDefault, formatDesc1, ... , &session );

Boolean needNewSession = ( VTDecompressionSessionCanAcceptFormatDescription( session, formatDesc2 ) == false);
VTDecompressionSession Summary
VTDecompressionSession Summary

Creation of VTDecompressionSession
VTDecompressionSession Summary

Creation of VTDecompressionSession
Make optimal decisions about output requirements
VTDecompressionSession Summary

Creation of VTDecompressionSession
Make optimal decisions about output requirements
Run your VTDecompressionSession synchronously and asynchronously
VTDecompressionSession Summary

Creation of VTDecompressionSession
Make optimal decisions about output requirements
Run your VTDecompressionSession synchronously and asynchronously
Handling changes in CMVideoFormatDescription
Case Three

Compressing CVPixelBuffers into a file
Compressing Video into a File
Compressing Video into a File

CVPixelBuffers
Compressing Video into a File

CVPixelBuffers → Movie File
Compressing Video into a File

CVPixelBuffers -> AVAssetWriter -> Movie File
Compressing Video into a File

Input: Camera

- CVPixelBuffers

Processing:
- Video Encoder

Output:
- AVAssetWriter

Result: Movie File
Compressing Video into a File

Video Encoder

AVAssetWriter

CMSampleBuffers

H.264

Movie File

CVPixelBuffers
Compressing Video into a File

- CVPixelBuffers
- Video Encoder
- CMSampleBuffers
- AVAssetWriter
- H.264
- File Writer
- Movie File
For More Details on AVAssetWriter

WWDC 2013—Moving to AVKit and AVFoundation
WWDC 2011—Working with Media in AVFoundation
Case Four
Compressing CVPixelBuffers for the network
Back Inside AVAssetWriter

1. CVPixelBuffers
2. AVAssetWriter
3. Video Encoder
4. CMSampleBuffers
5. H.264
6. File Writer
7. Movie File
VTCompressionSession

Getting access to the encoder
VTCompressionSession

Getting access to the encoder
VTCompressionSession

Getting access to the encoder
VTCompressionSession

Getting access to the encoder
Getting access to the encoder
VTCompressionSession
Getting access to the encoder

VTCompressionSession
Video Encoder

CVPixelBuffers

H.264

H.264

CMSampleBuffers

H.264

Wireless symbol
Creating a VTCompressionSession

What you need:
Creating a VTCompressionSession

What you need:

• Dimensions for compressed output
Creating a VTCompressionSession

What you need:

• Dimensions for compressed output
• Format for compression (e.g., kCMVideoCodecType_H264)
Creating a VTCompressionSession

What you need:

• Dimensions for compressed output
• Format for compression (e.g., kCMVideoCodecType_H264)
• PixelBufferAttributes describing source buffer requirements (optional)
Creating a VTCompressionSession

What you need:

- Dimensions for compressed output
- Format for compression (e.g., kCMVideoCodecType_H264)
- PixelBufferAttributes describing source buffer requirements (optional)
- A VTCompressionOutputCallback
Configuring VTCompressionSession
Configuring VTCompressionSession

Configure compression using VTSessionSetProperty() calls
Configuring VTCompressionSession

Configure compression using VTSessionSetProperty() calls
kVTCompressionPropertyKey_AllowFrameReordering
Configuring VTCompressionSession

Configure compression using VTSessionSetProperty() calls

- kVTCompressionPropertyKey_AllowFrameReordering
- kVTCompressionPropertyKey_AverageBitRate
Configuring VTCompressionSession

Configure compression using VTSessionSetProperty() calls

- kVTCompressionPropertyKey_AllowFrameReordering
- kVTCompressionPropertyKey_AverageBitRate
- kVTCompressionPropertyKey_H264EntropyMode
Configuring VTCompressionSession

Configure compression using VTSessionSetProperty() calls

kVTCompressionPropertyKey_AllowFrameReordering
kVTCompressionPropertyKey_AverageBitRate
kVTCompressionPropertyKey_H264EntropyMode
  kVT264EntropyMode_CAVLC/kVT264EntropyMode_CABAC
Configuring VTCompressionSession

Configure compression using VTSessionSetProperty() calls

kVTCompressionPropertyKey_AllowFrameReordering
kVTCompressionPropertyKey_AverageBitRate
kVTCompressionPropertyKey_H264EntropyMode
  kVTH264EntropyMode_CAVLC/kVTH264EntropyMode_CABAC
kVTCompressionPropertyKey_RealTime
Configuring VTCompressionSession

Configure compression using VTSessionSetProperty() calls

- kVTCompressionPropertyKey_AllowFrameReordering
- kVTCompressionPropertyKey_AverageBitRate
- kVTCompressionPropertyKey_H264EntropyMode
  - kVT_H264EntropyMode_CAVLC/kVT_H264EntropyMode_CABAC
- kVTCompressionPropertyKey_RealTime
- kVTCompressionPropertyKey_ProfileLevel
Configuring VTCompressionSession

Configure compression using VTSessionSetProperty() calls

- kVTCompressionPropertyKey_AllowFrameReordering
- kVTCompressionPropertyKey_AverageBitRate
- kVTCompressionPropertyKey_H264EntropyMode
  - kVT_H264EntropyMode_CAVLC/kVT_H264EntropyMode_CABAC
- kVTCompressionPropertyKey_RealTime
- kVTCompressionPropertyKey_ProfileLevel
  - for example: kVTProfileLevel_H264_Main_AutoLevel
Configuring VTCompressionSession

Configure compression using VTSessionSetProperty() calls

- kVTCompressionPropertyKey_AllowFrameReordering
- kVTCompressionPropertyKey_AverageBitRate
- kVTCompressionPropertyKey_H264EntropyMode
  - kVTH264EntropyMode_CAVLC/kVTH264EntropyMode_CABAC
- kVTCompressionPropertyKey_RealTime
- kVTCompressionPropertyKey_ProfileLevel
  - for example: kVTProfileLevel_H264_Main_AutoLevel

...and many more
Feeding VTCompressionSession
err = VTCompressionSessionEncodeFrame( session, pixelBuffer, presentationTime, ... );
Feeding VTCompressionSession

err = VTCompressionSessionEncodeFrame( session, pixelBuffer, presentationTime, ... );

• Source frames as CVPixelBuffers with presentation time
Feeding VTCompressionSession

err = VTCompressionSessionEncodeFrame( session, pixelBuffer, presentationTime, ... );

- Source frames as CV PixelBuffers with presentation time
- Presentation order
Feeding VTCompressionSession

```
err = VTCompressionSessionEncodeFrame( session, pixelBuffer, presentationTime, ... );
```

- Source frames as CVPixelBuffers with presentation time
- Presentation order
- Output may be delayed
Feeding VTCompressionSession

err = VTCompressionSessionEncodeFrame( session, pixelBuffer, presentationTime, ... );

• Source frames as CVPixelBuffers with presentation time
• Presentation order
• Output may be delayed
• Use VTCompressionSessionCompleteFrames() to finish pending frames
VTCompressionOutputCallback

VTCompressionSession
VTCompressionOutputCallback:

VTCompressionSession → H.264 → OutputCallback
VTCompressionOutputCallback:

- Output CMSampleBuffer
VTCompressionOutputCallback:

- Output CMSampleBuffer
- Compression error codes
VTCompressionOutputCallback:
- Output CMSampleBuffer
- Compression error codes
- Dropped frames
VTCompressionOutputCallback:

- Output CMSampleBuffer
- Compression error codes
- Dropped frames
- Frames emitted in decode order
CMSampleBuffers and Elementary Streams
CMSampleBuffers and Elementary Streams

MPEG-4

H.264 → H.264 → H.264 → CMSampleBuffers
CMSampleBuffers and Elementary Streams

MPEG-4

H.264
H.264
H.264

CMSampleBuffers

Elementary Stream
CMSampleBuffers and Elementary Streams

MPEG-4

H.264 → H.264 → H.264

Processing

Elementary Stream
H.264 Syntax

Conversion of parameter sets

MPEG-4

Elementary Stream
H.264 Syntax
Conversion of parameter sets

MPEG-4

SPS

PPS

CMVideoFormatDescription

Elementary Stream
H.264 Syntax
Conversion of parameter sets

MPEG-4

CMVideoFormatDescription

SPS
PPS

Elementary Stream

SPS
PPS
H.264 Syntax
Conversion of parameter sets

MPEG-4
CMVideoFormatDescription

Elementary Stream
CMVideoFormatDescriptionGetH264ParameterSetAtIndex

SPS
PPS
H.264 Syntax
NAL Unit headers

MPEG-4

Elementary Stream
H.264 Syntax

NAL Unit headers

MPEG-4

00 00 80 00

I Frame
(32768 bytes)

4-Byte Header:
Length

Elementary Stream
H.264 Syntax

NAL Unit headers

MPEG-4

00 00 80 00

I Frame
(32768 bytes)

4-Byte Header:
Length

Elementary Stream

00 00 01

I Frame

3- or 4-Byte Header:
Start Code
VTCompressionSession Summary
VTCompressionSession Summary

Creation of VTCompressionSession
VTCompressionSession Summary

Creation of VTCompressionSession
Configuring the compressor
VTCompressionSession Summary

Creation of VTCompressionSession
Configuring the compressor
Providing CVPixelBuffers to VTCompressionSession
VTCompressionSession Summary

Creation of VTCompressionSession
Configuring the compressor
Providing CVPixelBuffers to VTCompressionSession
Converting CMSampleBuffers into H.264 elementary stream packaging
Multi-Pass Encoding

Erik Turnquist
Core Media Engineer
Quality vs. Bit Rate
<table>
<thead>
<tr>
<th>Quality</th>
<th>Bit Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>High</td>
<td></td>
</tr>
<tr>
<td>Medium</td>
<td></td>
</tr>
<tr>
<td>Low</td>
<td></td>
</tr>
</tbody>
</table>
High
Medium
Low

Quality
Bit Rate
Both Quality and Bit Rate can be set to Low, Medium, or High levels.
What Is Multi-Pass Encoding?
Single-Pass Encoding
Single-Pass Encoding

CVPixelBuffers → VTCompressionSession (Video Encoder) → CMSampleBuffers
  H.264 → H.264 → Movie File
Single-Pass Encoding

CVPixelBuffers → VTCompressionSession → Video Encoder → CMSampleBuffers

H.264 → H.264 → Movie File

Finished
Single-Pass Encoding

CVPixelBuffers → VTCompressionSession → Video Encoder → CMSampleBuffers → Movie File

Finished
Multi-Pass Encoding
Multi-Pass Encoding

CVPixelBuffers → VTCompressionSession → Video Encoder → CMSampleBuffers

H.264 → H.264
Multi-Pass Encoding

VTCompressionSession

Video Encoder

CMSampleBuffers

Frame Database

H.264

H.264

CVPixelBuffers

Encoder Database
Multi-Pass Encoding

VTCompressionSession

Video Encoder

CMSampleBuffers

H.264  H.264

Frame Database

CVPixelBuffers

Encoder Database
Multi-Pass Encoding

VTCompressionSession

Finished

CVPixelBuffers

Video Encoder

CMSampleBuffers

Frame Database

H.264

Encoder Database
Multi-Pass Encoding

VTCompressionSession

Video Encoder

CVPixelBuffers

CMSampleBuffers

Frame Database

Finished

H.264

CMSampleBuffers

H.264

Encoder Database
Multi-Pass Encoding

VTCompressionSession

Video Encoder

CVPixelBuffers

Encoder Database

CMSampleBuffers

Frame Database

H.264

Movie File
Multi-Pass Encoding

- CVPixelBuffers
- VTCompressionSession
- Video Encoder
- CMSampleBuffers: H.264, H.264
- Frame Database
- Encoder Database
- Finished

Movie File
# Multi-Pass Encoder Features

<table>
<thead>
<tr>
<th></th>
<th>Single-Pass</th>
<th>Multi-Pass</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hardware Accelerated</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Knowledge of Future</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Change Decisions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Optimal Quality per Bit</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Multi-Pass Encoder Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Single-Pass</th>
<th>Multi-Pass</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hardware Accelerated</td>
<td>✔️</td>
<td>✔️</td>
</tr>
</tbody>
</table>
## Multi-Pass Encoder Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Single-Pass</th>
<th>Multi-Pass</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hardware Accelerated</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Knowledge of Future</td>
<td>✗</td>
<td>✔</td>
</tr>
</tbody>
</table>
## Multi-Pass Encoder Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Single-Pass</th>
<th>Multi-Pass</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hardware Accelerated</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Knowledge of Future</td>
<td>✗</td>
<td>✔</td>
</tr>
<tr>
<td>Change Decisions</td>
<td>✗</td>
<td>✔</td>
</tr>
</tbody>
</table>
# Multi-Pass Encoder Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Single-Pass</th>
<th>Multi-Pass</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hardware Accelerated</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Knowledge of Future</td>
<td>✗</td>
<td>✔</td>
</tr>
<tr>
<td>Change Decisions</td>
<td>✗</td>
<td>✔</td>
</tr>
<tr>
<td>Optimal Quality per Bit</td>
<td>✗</td>
<td>✔</td>
</tr>
</tbody>
</table>
New APIs
AVKit
AVFoundation
Video Toolbox
Core Media
Core Video
AVFoundation

New AVAssetExportSession property
Pass descriptions for AVAssetWriterInput
Reuse of AVAssetReaderOutput
AVAssetExportSession
Overview
AVAssetExportSession

Overview

Movie File \(\rightarrow\) CMSampleBuffers \(\rightarrow\) Video Decoder \(\rightarrow\) CVPixelBuffers \(\rightarrow\) Video Encoder \(\rightarrow\) CMSampleBuffers \(\rightarrow\) Movie File

H.264

Movie File
AVAssetExportSession

Overview

AVAssetExportSession

Movie File → CMSampleBuffers → H.264 → Video Decoder → CVPixelBuffers → Video Encoder → CMSampleBuffers → H.264 → Movie File
AVAssetExportSession
New APIs
AVAssetExportSession

New APIs

Multiple passes are taken care of automatically
AVAssetExportSession

New APIs

Multiple passes are taken care of automatically
• Falls back to single-pass if not supported
AVAssetExportSession

New APIs

Multiple passes are taken care of automatically
• Falls back to single-pass if not supported

@property (nonatomic) BOOL canPerformMultiplePassesOverSourceMediaData;
AVAssetWriter

Overview
AVAssetWriter
Overview

1. CVPixelBuffers
2. Video Encoder
3. CMSampleBuffers
4. H.264
5. Movie File
AVAssetWriter

Overview

OpenGL

CVPixelBuffers → Video Encoder → CMSampleBuffers → Movie File
AVAssetWriter Overview

OpenGL ES

Overview

AVAssetWriterInput

CVPixelBuffers

CMSampleBuffers

Video Encoder

H.264

Movie File
AVAssetWriter

Overview
AVAssetWriter

Overview

1. CVPixelBuffers
2. Video Encoder
3. CMSampleBuffers
4. H.264
5. Movie File
AVAssetWriter

Overview

Movie File → CMSampleBuffers → Video Decoder → CVPixelBuffers → Video Encoder → CMSampleBuffers → Movie File
AVAssetWriter
Overview

AVAssetReaderOutput

Movie File → CMSampleBuffers → H.264 → Video Decoder

AVAssetWriterInput

CVPixelBuffers → Video Encoder → CMSampleBuffers → H.264 → Movie File
AVAssetWriterInput
New APIs
AVAssetWriterInput

New APIs

Enable multi-pass encoding if supported

@property (nonatomic) BOOL performsMultiPassEncodingIfSupported;
**AVAssetWriterInput**

**New APIs**

Enable multi-pass encoding if supported

```c
@property (nonatomic) BOOL performsMultiPassEncodingIfSupported;
```

End current pass after appending samples

```c
- (void)markCurrentPassAsFinished;
```
AVAssetWriterInput

New APIs

Enable multi-pass encoding if supported

@property (nonatomic) BOOL performsMultiPassEncodingIfSupported;

End current pass after appending samples

- (void)markCurrentPassAsFinished;

Triggers encoder analysis
AVAssetWriterInput

New APIs

Enable multi-pass encoding if supported

@property (nonatomic) BOOL performsMultiPassEncodingIfSupported;

End current pass after appending samples

-(void)markCurrentPassAsFinished;

Triggers encoder analysis

Encoder decides if it wants more passes and what time ranges
AVAssetWriterInputPassDescription
AVAssetWriterInputPassDescription

Encoder’s request for samples in next pass
- May contain subsets of entire sequence
AVAssetWriterInputPassDescription

Encoder’s request for samples in next pass
• May contain subsets of entire sequence

Query pass description time ranges

@property (nonatomic, readonly) NSArray *sourceTimeRanges;
AVAssetWriterInputPassDescription

Encoder’s request for samples in next pass

• May contain subsets of entire sequence

Query pass description time ranges

@property (nonatomic, readonly) NSArray *sourceTimeRanges;

Array of CMTimeRanges as NSValues
AVAssetWriterInput
Pass descriptions
Pass descriptions

Block is called when encoder makes decision about next pass

- (void)respondToEachPassDescriptionOnQueue:(dispatch_queue_t)queue usingBlock:(dispatch_block_t)block;
AVAssetWriterInput

Pass descriptions

Block is called when encoder makes decision about next pass

- (void)respondToEachPassDescriptionOnQueue:(dispatch_queue_t)queue
  usingBlock:(dispatch_block_t)block;

Get the new description

@property (readonly) AVAssetWriterInputPassDescription *currentPassDescription;
AVAssetWriterInput
Sample
AVAssetWriterInput

Sample

[assetWriterInput respondToEachPassDescriptionOnQueue:queue usingBlock:^{
    AVAssetWriterInputPassDescription *pass = [assetWriterInput currentPassDescription];
    if (pass != nil) {
        // Reconfigure source to deliver samples
        [source deliverSamplesFromPassDescription:pass];

        // Ready to start next pass
        [assetWriterInput requestMediaDataWhenReadyOnQueue:queue usingBlock:block];
    } else {
        // Finished
        [assetWriterInput markAsFinished];
    }
}];
AVAssetWriterInput
Sample

[assetWriterInput respondToEachPassDescriptionOnQueue:queue usingBlock:^{
    AVAssetWriterInputPassDescription *pass = [assetWriterInput currentPassDescription];
    if (pass != nil) {
        // Reconfigure source to deliver samples
        [source deliverSamplesFromPassDescription:pass];

        // Ready to start next pass
        [assetWriterInput requestMediaDataWhenReadyOnQueue:queue usingBlock:block];
    } else {
        // Finished
        [assetWriterInput markAsFinished];
    }
}];
AVAssetWriterInput
Sample

[assetWriterInput respondToEachPassDescriptionOnQueue:queue usingBlock:^{
    AVAssetWriterInputPassDescription *pass = [assetWriterInput currentPassDescription];
    if (pass != nil) {
        // Reconfigure source to deliver samples
        [source deliverSamplesFromPassDescription:pass];

        // Ready to start next pass
        [assetWriterInput requestMediaDataWhenReadyOnQueue:queue usingBlock:block];
    } else {
        // Finished
        [assetWriterInput markAsFinished];
    }
}];
AVAssetWriterInput
Sample

[assetWriterInput respondToEachPassDescriptionOnQueue:queue usingBlock:^{
    AVAssetWriterInputPassDescription *pass = [assetWriterInput currentPassDescription];
    if (pass != nil) {
        // Reconfigure source to deliver samples
        [source deliverSamplesFromPassDescription:pass];
        // Ready to start next pass
        [assetWriterInput requestMediaDataWhenReadyOnQueue:queue usingBlock:block];
    } else {
        // Finished
        [assetWriterInput markAsFinished];
    }
}];
```objective-c
[assetWriterInput respondToEachPassDescriptionOnQueue:queue usingBlock:^(AVAssetWriterInputPassDescription *pass) {
    if (pass != nil) {
        // Reconfigure source to deliver samples
        [source deliverSamplesFromPassDescription:pass];

        // Ready to start next pass
        [assetWriterInput requestMediaDataWhenReadyOnQueue:queue usingBlock:block];
    } else {
        // Finished
        [assetWriterInput markAsFinished];
    }
}];
```
AVAssetReaderOutput

New APIs
AVAssetReaderOutput

New APIs

Prepare source for multi-pass

@property (nonatomic) BOOL supportsRandomAccess;
AVAssetReaderOutput

New APIs

Prepare source for multi-pass

@property (nonatomic) BOOL supportsRandomAccess;

Reconfigure source to deliver samples in time ranges

-(void)resetForReadingTimeRanges:(NSArray *)timeRanges;
AVAssetReaderOutput
New APIs

Prepare source for multi-pass

@property (nonatomic) BOOL supportsRandomAccess;

Reconfigure source to deliver samples in time ranges

- (void)resetForReadingTimeRanges:(NSArray *)timeRanges;

All passes have completed

- (void)markConfigurationAsFinal;
AVAssetReader and AVAssetWriter
Enable AVAssetReaderOutput if AVAssetWriterInput support multi-pass

readerOutput.supportsRandomAccess = writerInput.canPerformMultiplePasses;
Enable AVAssetReaderOutput if AVAssetWriterInput support multi-pass

```swift
readerOutput.supportsRandomAccess = writerInput.canPerformMultiplePasses;
```

Reconfigure source to deliver samples for an AVAssetWriterInput

```swift
[readerOutput resetForReadingTimeRanges:passDescription.sourceTimeRanges];
```
AVAssetReaderOutput

Sample
AVAssetReaderOutput

Sample

```swift
[assetWriterInput respondToEachPassDescriptionOnQueue:queue usingBlock:^{
    AVAssetWriterInputPassDescription *pass = [assetWriterInput currentPassDescription];
    if (currentPass != nil) {
        // Reconfigure source to deliver samples
        [readerOutput resetForReadingTimeRanges:pass.sourceTimeRanges];

        // Ready to start next pass
        [assetWriterInput requestMediaDataWhenReadyOnQueue:queue usingBlock:block];
    } else {
        // Finished
        [assetWriterInput markAsFinished];
    }
};
```
Video Toolbox

Encoder frame analysis database
• VTMultiPassStorage

Additions to VTCompressionSession

Compressed frame database
• VTFrameSilo
Video Toolbox Multi-Pass Architecture
Video Toolbox Multi-Pass Architecture

CVPixelBuffers -> VTCompressionSession -> Video Encoder -> CMSampleBuffers

H.264 -> H.264
Video Toolbox Multi-Pass
Architecture

- CVPixelBuffers
- VTCompressionSession
- Video Encoder
- CMSampleBuffers
- H.264
- H.264
- VTMultiPassStorage
- VTFrameSilo
Video Toolbox Multi-Pass
Architecture

VTCompressionSession
Video Encoder
VTMultiPassStorage

CVPixelBuffers

Resend Frames

H.264
H.264

CMSampleBuffers

VTFramesilo
Video Toolbox Multi-Pass Architecture

- CVPixelBuffers
- VTCompressionSession
- Video Encoder
- VTMultiPassStorage
- CMSampleBuffers
  - H.264
  - H.264
- VTFrameSilo
Video Toolbox Multi-Pass

Architecture

CVPixelBuffers → VTCompressionSession → Video Encoder

VTMultiPassStorage

CMSampleBuffers → H.264 → H.264 → VTFrameSilo

Finished
Video Toolbox Multi-Pass

Architecture

VTCompressionSession

Video Encoder

CMSampleBuffers

H.264
H.264

VTFrameSilo

CVPixelBuffers

VTMultiPassStorage

Finished
Video Toolbox Multi-Pass Architecture

VTCompressionSession

Video Encoder

VTMultiPassStorage

CMSampleBuffers

H.264

H.264

VTMultiPassStorage

VTFrameSilo

Movie File

Finished
Video Toolbox Multi-Pass

Architecture

- CVPixelBuffers
- VTCompressionSession
  - Video Encoder
  - VTMultiPassStorage
- CMSampleBuffers
  - H.264
- VTFramesilo
  - Movie File
VTMultiPassStorage

Encoder Analysis

VTMultiPassStorage
Create the encoder analysis storage

```c
error = VTMultiPassStorageCreate( allocator, fileURL, timeRange, options, &storage );
```
VTMultiPassStorage

Create the encoder analysis storage

```c
error = VTMultiPassStorageCreate( allocator, fileURL, timeRange, options, &storage );
```

Close the file

```c
error = VTMultiPassStorageClose( storage );
```
VTCompressionSession

Enable multi-pass

```c
error = VTSessionSetProperty( session,
    kVTCompressionPropertyKey_MultiPassStorage, multiPassStorage );
```
VTCompressionSession
VTCompressionSession

Begin a pass

```c
error = VTCompressionSessionBeginPass( session, 0, NULL );
```
Begin a pass

```
error = VTCompressionSessionBeginPass( session, 0, NULL );
```

End a pass

• Asks the encoder if another pass can be performed

```
error = VTCompressionSessionEndPass( session,
    &furtherPassesRequested, NULL );
```
Begin a pass

error = VTCompressionSessionBeginPass( session, 0, NULL );

End a pass

- Asks the encoder if another pass can be performed
  error = VTCompressionSessionEndPass( session,
                                      &furtherPassesRequested, NULL );

Retrieve time ranges for next pass

error = VTCompressionSessionGetTimeRangesForNextPass( session,
                                                       &timeRangesCount,
                                                       &timeRangeArray );
VTFrameSilo

CMSampleBuffers

H.264  H.264

VTFrameSilo
VTFrameSilo

Create compressed frame storage

```
error = VTFrameSiloCreate( allocator, fileURL, timeRange, options, &silo );
```
VTFrameSilo

Create compressed frame storage

```c
error = VTFrameSiloCreate( allocator, fileURL, timeRange, options, &silo );
```

Add a sample in VTCompressionOutputCallback

```c
error = VTFrameSiloAddSampleBuffer( silo, sampleBuffer );
```
VTFrameSilo

Create compressed frame storage

```c
error = VTFrameSiloCreate( allocator, fileURL, timeRange, options, &silo );
```

Add a sample in VTCompressionOutputCallback

```c
error = VTFrameSiloAddSampleBuffer( silo, sampleBuffer );
```

Prepare for next pass

```c
error = VTFrameSiloSetTimeRangesForNextPass( silo,
                                          timeRangeCount, timeRangeArray );
```
VTFrameSilo
File output

VTFrameSilo → Movie File
Retrieve samples for time range

```c
error = VTFrameSiloCallBlockForEachSampleBuffer(silo,
    timeRange, ^(CMSampleBuffer sampleBuffer) {
        // append sampleBuffer to movie file
    });
```
Considerations
Use Case Considerations

Best Choice
Use Case Considerations

<table>
<thead>
<tr>
<th>Best Choice</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real Time</td>
</tr>
<tr>
<td>Your App Experiment</td>
</tr>
</tbody>
</table>
## Use Case Considerations

<table>
<thead>
<tr>
<th></th>
<th>Best Choice</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real Time</td>
<td>Single-Pass</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Use Case Considerations

<table>
<thead>
<tr>
<th></th>
<th>Best Choice</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><strong>Real Time</strong></td>
</tr>
<tr>
<td>Minimum Power Use</td>
<td></td>
</tr>
<tr>
<td>Use Case Considerations</td>
<td></td>
</tr>
<tr>
<td>-------------------------</td>
<td></td>
</tr>
<tr>
<td><strong>Best Choice</strong></td>
<td></td>
</tr>
<tr>
<td>Real Time</td>
<td>Single-Pass</td>
</tr>
<tr>
<td>Minimum Power Use</td>
<td>Single-Pass</td>
</tr>
</tbody>
</table>
## Use Case Considerations

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Best Choice</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real Time</td>
<td>Single-Pass</td>
</tr>
<tr>
<td>Minimum Power Use</td>
<td>Single-Pass</td>
</tr>
<tr>
<td>Minimum Temporary Storage</td>
<td></td>
</tr>
</tbody>
</table>
Use Case Considerations

<table>
<thead>
<tr>
<th></th>
<th>Best Choice</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real Time</td>
<td>Single-Pass</td>
</tr>
<tr>
<td>Minimum Power Use</td>
<td>Single-Pass</td>
</tr>
<tr>
<td>Minimum Temporary Storage</td>
<td>Single-Pass</td>
</tr>
<tr>
<td>Your App Experiment</td>
<td></td>
</tr>
</tbody>
</table>
# Use Case Considerations

<table>
<thead>
<tr>
<th></th>
<th>Best Choice</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real Time</td>
<td>Single-Pass</td>
</tr>
<tr>
<td>Minimum Power Use</td>
<td>Single-Pass</td>
</tr>
<tr>
<td>Minimum Temporary Storage</td>
<td>Single-Pass</td>
</tr>
<tr>
<td>Best Quality</td>
<td></td>
</tr>
</tbody>
</table>
## Use Case Considerations

<table>
<thead>
<tr>
<th></th>
<th>Best Choice</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real Time</td>
<td>Single-Pass</td>
</tr>
<tr>
<td>Minimum Power Use</td>
<td>Single-Pass</td>
</tr>
<tr>
<td>Minimum Temporary Storage</td>
<td>Single-Pass</td>
</tr>
<tr>
<td>Best Quality</td>
<td>Multi-Pass</td>
</tr>
</tbody>
</table>
# Use Case Considerations

<table>
<thead>
<tr>
<th>Feature</th>
<th>Best Choice</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real Time</td>
<td>Single-Pass</td>
</tr>
<tr>
<td>Minimum Power Use</td>
<td>Single-Pass</td>
</tr>
<tr>
<td>Minimum Temporary Storage</td>
<td>Single-Pass</td>
</tr>
<tr>
<td>Best Quality</td>
<td>Multi-Pass</td>
</tr>
<tr>
<td>Closer to Target Bit Rate</td>
<td></td>
</tr>
</tbody>
</table>
## Use Case Considerations

<table>
<thead>
<tr>
<th>Feature</th>
<th>Best Choice</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real Time</td>
<td>Single-Pass</td>
</tr>
<tr>
<td>Minimum Power Use</td>
<td>Single-Pass</td>
</tr>
<tr>
<td>Minimum Temporary Storage</td>
<td>Single-Pass</td>
</tr>
<tr>
<td>Best Quality</td>
<td>Multi-Pass</td>
</tr>
<tr>
<td>Closer to Target Bit Rate</td>
<td>Multi-Pass</td>
</tr>
</tbody>
</table>
# Use Case Considerations

<table>
<thead>
<tr>
<th>Feature</th>
<th>Best Choice</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real Time</td>
<td>Single-Pass</td>
</tr>
<tr>
<td>Minimum Power Use</td>
<td>Single-Pass</td>
</tr>
<tr>
<td>Minimum Temporary Storage</td>
<td>Single-Pass</td>
</tr>
<tr>
<td>Best Quality</td>
<td>Multi-Pass</td>
</tr>
<tr>
<td>Closer to Target Bit Rate</td>
<td>Multi-Pass</td>
</tr>
<tr>
<td>Okay to Take Longer</td>
<td></td>
</tr>
</tbody>
</table>
# Use Case Considerations

<table>
<thead>
<tr>
<th>Use Case</th>
<th>Best Choice</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real Time</td>
<td>Single-Pass</td>
</tr>
<tr>
<td>Minimum Power Use</td>
<td>Single-Pass</td>
</tr>
<tr>
<td>Minimum Temporary Storage</td>
<td>Single-Pass</td>
</tr>
<tr>
<td>Best Quality</td>
<td>Multi-Pass</td>
</tr>
<tr>
<td>Closer to Target Bit Rate</td>
<td>Multi-Pass</td>
</tr>
<tr>
<td>Okay to Take Longer</td>
<td>Multi-Pass</td>
</tr>
</tbody>
</table>
## Use Case Considerations

<table>
<thead>
<tr>
<th>Feature</th>
<th>Best Choice</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real Time</td>
<td>Single-Pass</td>
</tr>
<tr>
<td>Minimum Power Use</td>
<td>Single-Pass</td>
</tr>
<tr>
<td>Minimum Temporary Storage</td>
<td>Single-Pass</td>
</tr>
<tr>
<td>Best Quality</td>
<td>Multi-Pass</td>
</tr>
<tr>
<td>Closer to Target Bit Rate</td>
<td>Multi-Pass</td>
</tr>
<tr>
<td>Okay to Take Longer</td>
<td>Multi-Pass</td>
</tr>
<tr>
<td>Your App</td>
<td></td>
</tr>
</tbody>
</table>
## Use Case Considerations

<table>
<thead>
<tr>
<th>Feature</th>
<th>Best Choice</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real Time</td>
<td>Single-Pass</td>
</tr>
<tr>
<td>Minimum Power Use</td>
<td>Single-Pass</td>
</tr>
<tr>
<td>Minimum Temporary Storage</td>
<td>Single-Pass</td>
</tr>
<tr>
<td>Best Quality</td>
<td>Multi-Pass</td>
</tr>
<tr>
<td>Closer to Target Bit Rate</td>
<td>Multi-Pass</td>
</tr>
<tr>
<td>Okay to Take Longer</td>
<td>Multi-Pass</td>
</tr>
<tr>
<td>Your App</td>
<td>Experiment</td>
</tr>
</tbody>
</table>
Content Considerations

Best Choice
### Content Considerations

<table>
<thead>
<tr>
<th>Complexity</th>
<th>Choice</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low Complexity</td>
<td>Best Choice</td>
</tr>
<tr>
<td>High Complexity</td>
<td>Single/Multi</td>
</tr>
<tr>
<td>Varying Complexity</td>
<td>Multi-Pass</td>
</tr>
</tbody>
</table>

Your Content Experiment
## Content Considerations

<table>
<thead>
<tr>
<th>Best Choice</th>
<th>Low Complexity</th>
<th>Single/Multi</th>
</tr>
</thead>
<tbody>
<tr>
<td>Varying Complexity</td>
<td>Multi-Pass</td>
<td></td>
</tr>
<tr>
<td>Your Content Experiment</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Content Considerations

<table>
<thead>
<tr>
<th>Complexity</th>
<th>Best Choice</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low Complexity</td>
<td>Single/Multi</td>
</tr>
<tr>
<td>High Complexity</td>
<td></td>
</tr>
</tbody>
</table>

**Your Content Experiment**
## Content Considerations

<table>
<thead>
<tr>
<th>Low Complexity</th>
<th>Single/Multi</th>
</tr>
</thead>
<tbody>
<tr>
<td>High Complexity</td>
<td>Single/Multi</td>
</tr>
</tbody>
</table>

Best Choice
## Content Considerations

<table>
<thead>
<tr>
<th>Complexity</th>
<th>Best Choice</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low Complexity</td>
<td>Single/Multi</td>
</tr>
<tr>
<td>High Complexity</td>
<td>Single/Multi</td>
</tr>
<tr>
<td>Varying Complexity</td>
<td>Multi-Pass</td>
</tr>
</tbody>
</table>
## Content Considerations

<table>
<thead>
<tr>
<th>Complexity</th>
<th>Best Choice</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low Complexity</td>
<td>Single/Multi</td>
</tr>
<tr>
<td>High Complexity</td>
<td>Single/Multi</td>
</tr>
<tr>
<td>Varying Complexity</td>
<td>Multi-Pass</td>
</tr>
</tbody>
</table>
## Content Considerations

<table>
<thead>
<tr>
<th>Complexity Type</th>
<th>Best Choice</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low Complexity</td>
<td>Single/Multi</td>
</tr>
<tr>
<td>High Complexity</td>
<td>Single/Multi</td>
</tr>
<tr>
<td>Varying Complexity</td>
<td>Multi-Pass</td>
</tr>
<tr>
<td>Your Content</td>
<td></td>
</tr>
</tbody>
</table>
## Content Considerations

<table>
<thead>
<tr>
<th>Complexity</th>
<th>Best Choice</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low Complexity</td>
<td>Single/Multi</td>
</tr>
<tr>
<td>High Complexity</td>
<td>Single/Multi</td>
</tr>
<tr>
<td>Varying Complexity</td>
<td>Multi-Pass</td>
</tr>
<tr>
<td>Your Content</td>
<td>Experiment</td>
</tr>
</tbody>
</table>
Summary
AVFoundation provides powerful APIs to operate on media
AVFoundation provides powerful APIs to operate on media
Video Toolbox APIs provide direct codec access
Summary

AVFoundation provides powerful APIs to operate on media
Video Toolbox APIs provide direct codec access
Multi-pass can provide substantial quality improvements
More Information

Evangelism
evangelism@apple.com

AVFoundation Documentation
AVFoundation Programming Guide

Apple Developer Forums
http://devforums.apple.com
### Related Sessions

<table>
<thead>
<tr>
<th>Session</th>
<th>Location</th>
<th>Date/Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mastering Modern Media Playback</td>
<td>Mission</td>
<td>Tuesday 11:30AM</td>
</tr>
<tr>
<td>Harnessing Metadata in Audiovisual Media</td>
<td>Pacific Heights</td>
<td>Tuesday 2:00PM</td>
</tr>
<tr>
<td>Camera Capture: Manual Controls</td>
<td>Marina</td>
<td>Wednesday 11:30AM</td>
</tr>
<tr>
<td>Introducing the Photos Frameworks</td>
<td>Nob Hill</td>
<td>Thursday 10:15AM</td>
</tr>
</tbody>
</table>
Labs

- AVFoundation and Camera Capture Lab
  Media Lab A
  Thursday 2:00PM