To Advance through Presentation
Use Page Up and Page Down Keys

99 | Worldwide
Developers
Conference

Think Fast! Performance!

Glenn Fisher

Performance Marketing Manager

Introduction

- Tools to optimize performance of your software on Macintosh
  - The Instrumentation SDK
  - 4PM

Why Performance?

- Performance is the #1 customer issue
- Hardware advantage provides a clear competitive opportunity
- Performance is a direct competitive advantage for applications

Instrumentation SDK

Roger Pantos

Classic (Blue Box) Engineering

99 Worldwide Developers Conference

99 Worldwide Developers Conference

How to Use the Instrumentation SDK to Improve Application Performance

Outline

- An Overview
- How Instrumentation Improved My Life
- Dodging Questions from the Crowd

Turn This…


```
Whiskers:Code:EKG:Source:TimeLineViewer.cp

MPW Shell

OSStatus     TLPointContainer::ScanForward( const Nanoseconds& toTime, const Nanoseconds
{
ViewTraceData        traceRec;
Nanoseconds          curEnd, destEnd;

    m_InstLogTraceEvent( sFnTrace, 0, kInstStartEvent);

    curEnd = U64ADD( fCurTime, fVisRange);

    fVisiblePoints.RemoveItemsAt( fVisiblePoints.GetCount(), kFirstDynamicArrayIndex
    numVisStarts = numVisEnds = 0;

    if ( fMatchFlags && noErr == ( err = VPSetTraceIterPos( fOwnerRef, fTraceIter, &t
    {
        while ( noErr == ( err = VPGetNextTrace( fOwnerRef, fTraceIter, &traceRec)) &
                U64COMPARE( traceRec.fTimeStamp, toTime) < 0)
        {
            if ( !this->NodeIndex( traceRec.fOpaqueNodeRef, &mustMatch) || mustMatch
        }
    }
    if ( err == kViewStatusNoData)          // acceptable termination condition
        err = noErr;

    m_InstLogTraceEvent( sFnTrace, 0, kInstMiddleEvent);
```
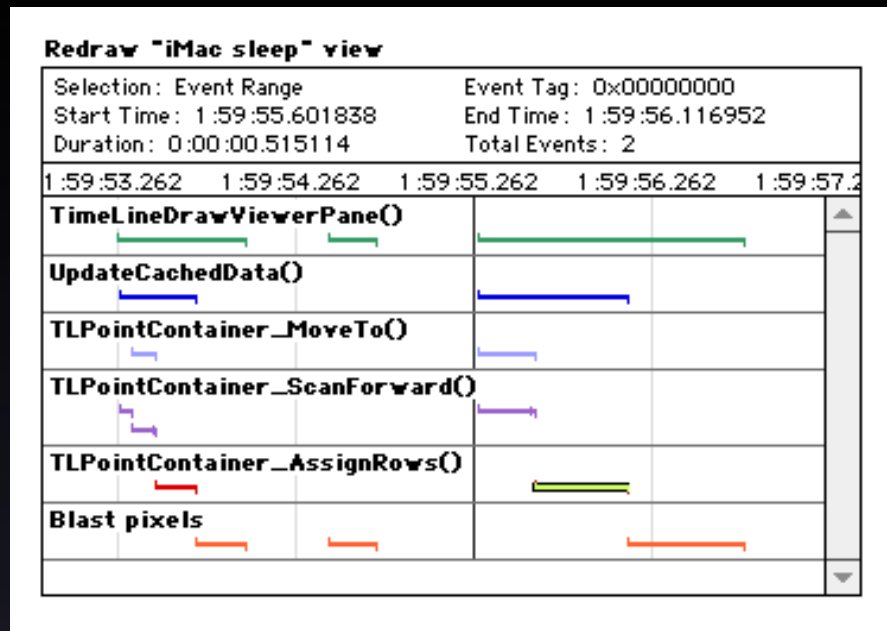
…Into This

Using the SDK

- Add instrumentation to your code
- Run the code and collect the data
- Use the viewer to analyze the data
- Make changes
- Retest!

Better Performance: A Five-Step Plan

- Identify the performance target
- Instrument your code
- Take the easy wins
- Redesign algorithms
- Micro-optimize

The Process

The Process

- Start with a goal

The Process

- Start with a goal
- Establish a baseline

The Process

- Start with a goal
- Establish a baseline
- Cover the baseline

The Process

- Start with a goal
- Establish a baseline
- Cover the baseline
- Drill down

The Process

- Start with a goal
- Establish a baseline
- Cover the baseline
- Drill down

- Select your target

The Process

- Start with a goal
- Establish a baseline
- Cover the baseline
- Drill down

- Select your target
- Optimize it

The Process

- Start with a goal
- Establish a baseline
- Cover the baseline
- Drill down

- Select your target
- Optimize it
- Retest

The Process

- Start with a goal
- Establish a baseline
- Cover the baseline
- Drill down

- Select your target
- Optimize it
- Retest

The Process

- Start with a goal
- Establish a baseline
- Cover the baseline
- Drill down

- Select your target
- Optimize it
- Retest
- **Party!**

The Results

- One day spent doing analysis and optimization
- 14x speed improvement on test case

The Results

- One day spent doing analysis and optimization
- 14x speed improvement on test case
- User-visible increase in browsing speed:
  - Release version took 18s
  - Required 9MB app partition

The Results

- One day spent doing analysis and optimization

- 14x speed improvement on test case

- User-visible increase in browsing speed:
  - Release version took 18s
  - Required 9MB app partition

  - Optimized version took 6s
  - In a 3MB partition

Instrumentation SDK

- Can be used to graphically illustrate your application's behavior

- Reveals significant opportunities to improve application performance

- Available today on the Mac OS SDK CD series

Introduction

- Hardware and software interaction for performance

- Performance is impacted by:
  - Processors, caches, memory, I/O, co-processors, compilers, OS…

- Focus of this talk:
  - Determine system performance leaks

Performance Metrics

- Must represent your application's goals
  - Execution time
  - Through-put
  - Bandwidth
  - Frequency
  - Response time
  - Frames per second
  - …

Tools for Performance Analysis

- Hardware counters
  - Processor
    - 4 counters (G3 and G4)
  - Memory controller
    - 4 counters (G3 system)

- 4PM: HW counter instrumentation

4PM

Justin Bishop

Performance Engineer,
Architecture and
Performance Engineering

4PM

- A tool for using processor and memory controller counters
  - Select events to be monitored
  - Save counter values after measurement
  - Turn counters on/off using the 4PM Library or a 'hot key'
  - Generate a single total number or histogram of the selected events

4PM Library

- Library provides four calls:
  - Init4PMLibrary
  - Delete4PMLibrary
  - Start4PM
  - Stop4PM

Processor Events

- Instructions executed
- Branches taken
- TLB misses
- L1 misses
- L2 misses
- And more…

Memory Controller Events

- Processor memory transactions
- PCI master memory transactions
- Cycles memory is busy
- Cycles PCI is busy
- And more…

4PM Shortcuts

- Shortcuts can be used to provide high level information such as:
    - Memory bandwidth
    - PCI bandwidth
    - L2 miss ratios

Demo

Summary

- Define the goal: performance metric
- Understand hardware and software trade-off
- Architect for high performance
- Measure and verify

Action Items

- Get Instrumentation SDK and 4PM
- Determine performance issues
- Optimize your code

Q&A

**To Advance through Presentation
Use Page Up and Page Down Keys**

99 | Worldwide Developers Conference