Developing for Android on ARM

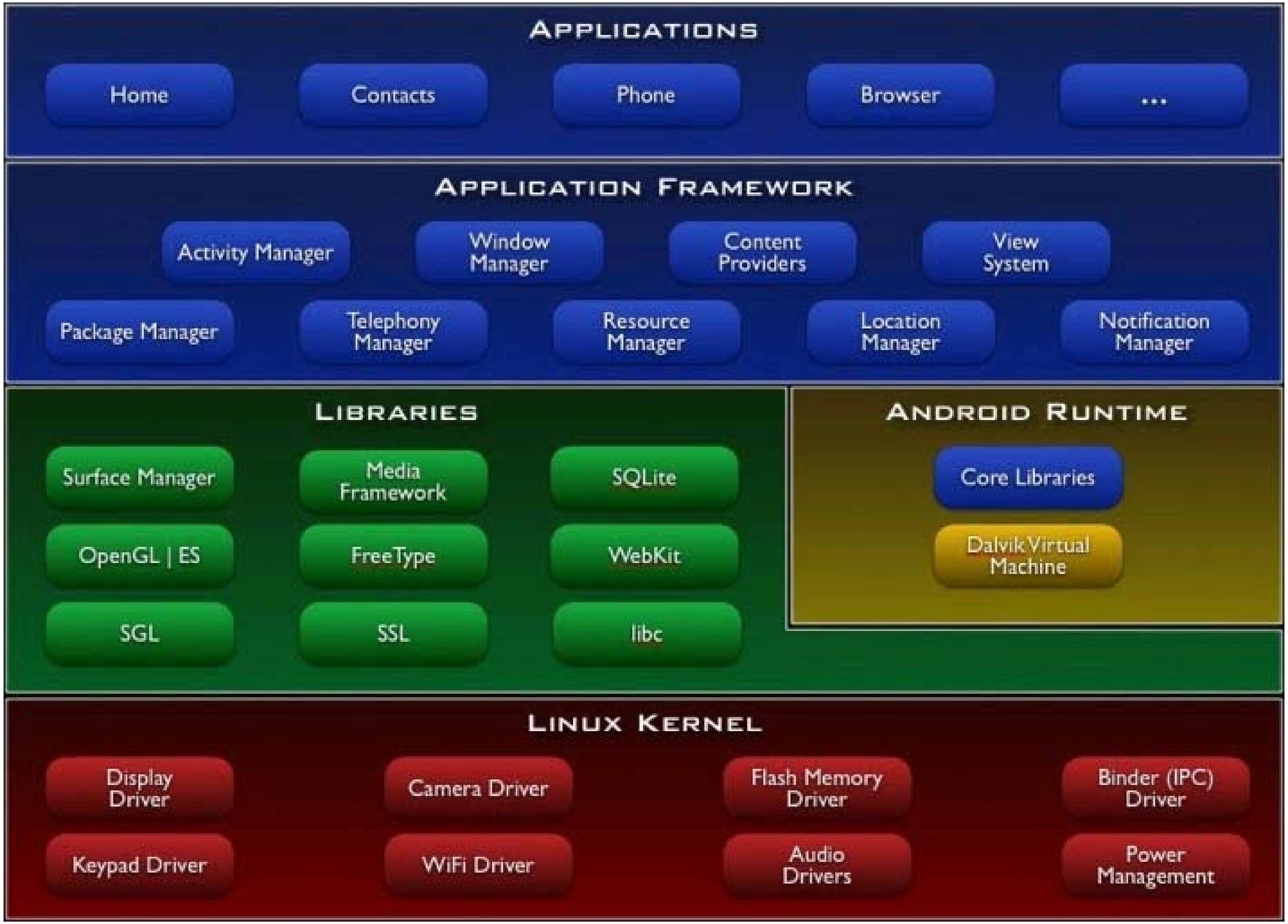
Ashley Stevens- Solution Architect

Google Android

- 10’s of phone models
  - 18-20 by year end
- 1000’s of apps
  - Over 12,000 at Nov 5th
- Millions of users

How to get your app to them…?

Android Architecture



APPLICATIONS

Home

Contacts

Phone

Browser

...

APPLICATION FRAMEWORK

Activity Manager

Window Manager

Content Providers

View System

Package Manager

Telephony Manager

Resource Manager

Location Manager

Notification Manager

LIBRARIES

Surface Manager

Media Framework

SQLite

OpenGL | ES

FreeType

WebKit

SGL

SSL

libc

ANDROID RUNTIME

Core Libraries

Dalvik Virtual Machine

LINUX KERNEL

Display Driver

Camera Driver

Flash Memory Driver

Binder (IPC) Driver

Keypad Driver

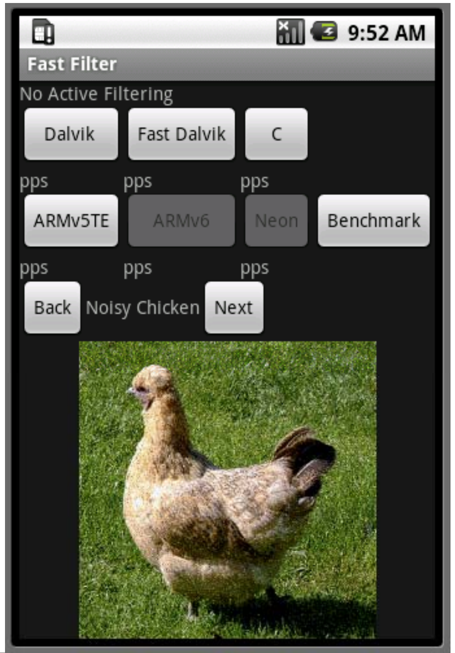
WiFi Driver

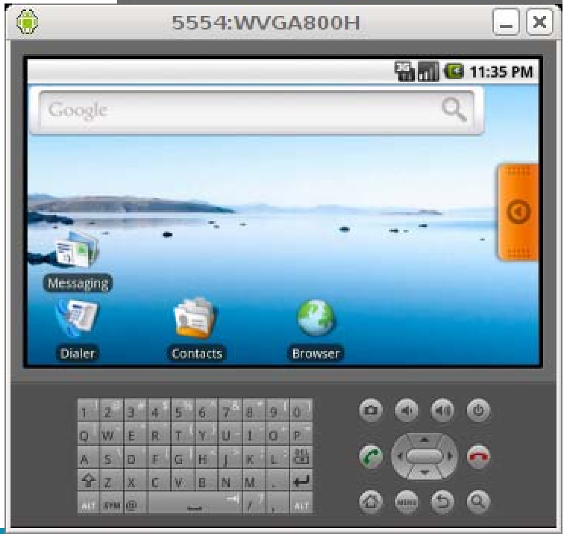
Audio Drivers

Power Management

Developing Using Android/Dalvik

- The UI is compiled from an XML file
  - You can use the GUI in the Android tools or hand craft the XML
- Simple to attach code to UI elements
- Emulator enables easy testing of code
  - Based on QEMU with a model of an ARM926
  - Real devices are typically ARM11, Cortex-A8 or Cortex-A9 cores
- Testing anything that relies on touch, accelerometer, GPS or network capabilities easier on real hardware



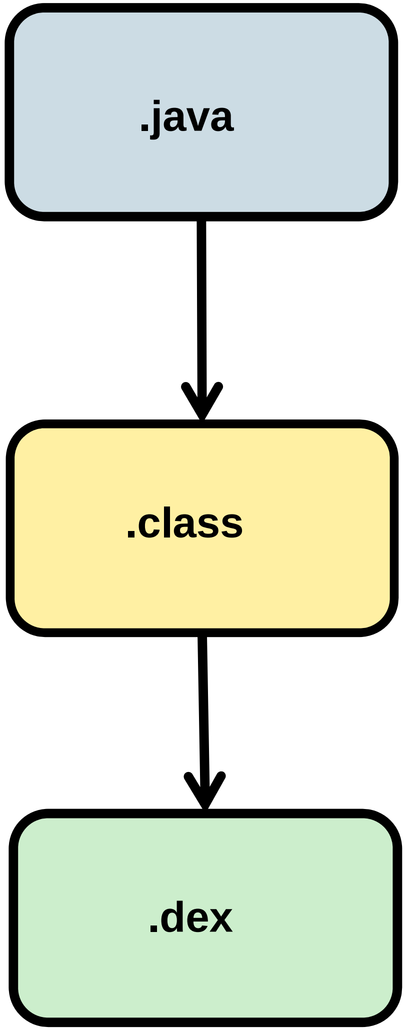


Dalvik VM Optimizations

- Dalvik JIT announced publically at Google-IO May ’09
- Google has asked ARM to engage on Dalvik open source project
  - ARM has initially worked on interpreter improvements ***>10% Uplift***
- When Google open source JIT ARM will contribute to code generation
  - Thumb2 and Thumb-2 EE (as appropriate)
  - Goal is for 3x-8x performance improvement for Java intensive apps

Dalvik Virtual Machine

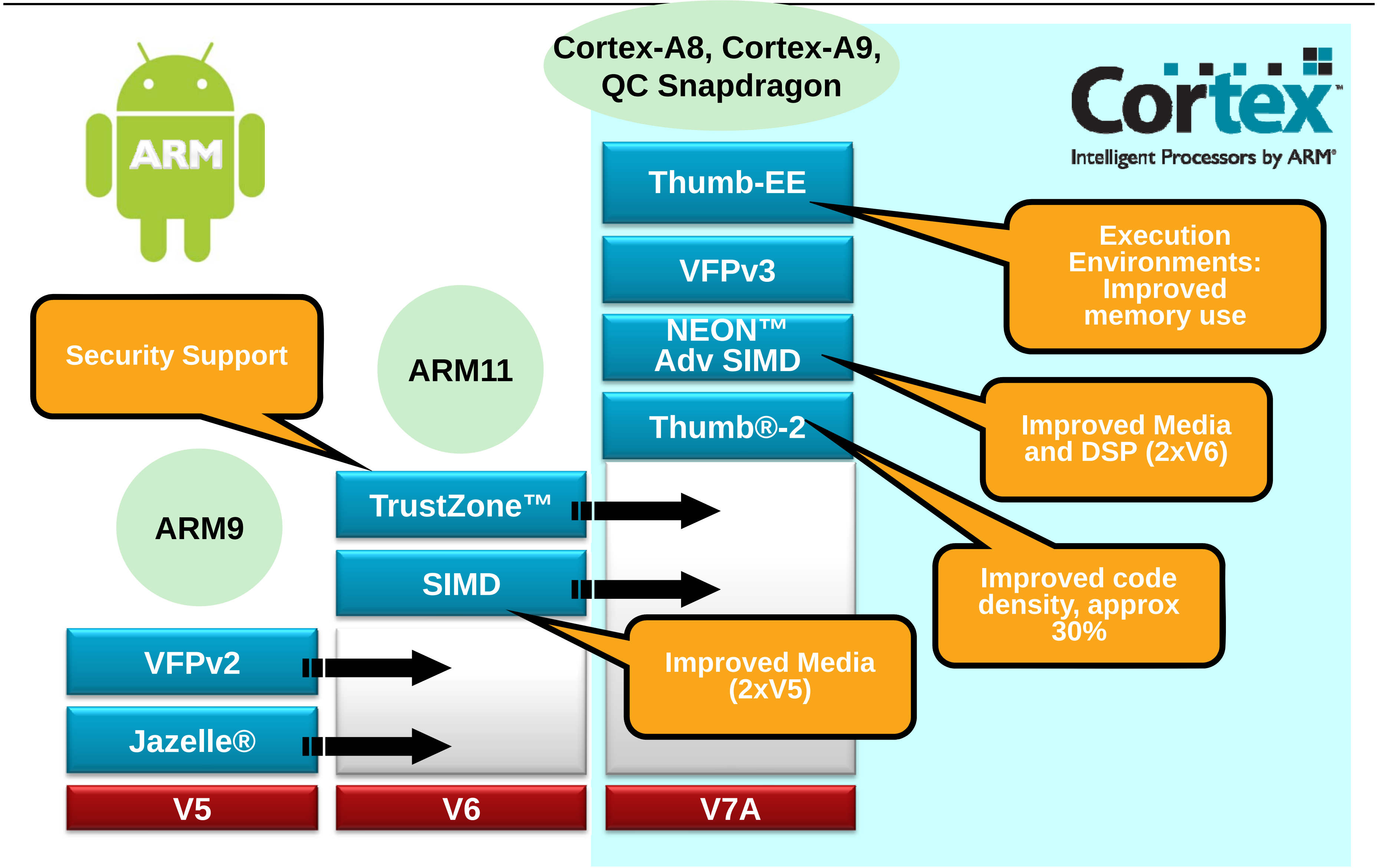
- Optimized for low-end, low-memory systems
  - Designed to run on systems with min 64MB memory
- Register-based VM (Java VM stack based)
  - Register-based VM better interpreted performance
  - Minimum instruction size 2 bytes
  - 30% fewer instructions than Java VM
  - Avoid overhead of instruction dispatch in interpreter
  - Higher semantic density, fewer instructions
  - 35% more bytes in instruction stream than Java VM
- Each VM runs in a separate OS process
  - Dalvik relies on OS to provide process management
- Development process
  - Develop in Java, compiled to Java class file then convert to dex
  - Automatically handled in Make files



ARM Optimization of Dalvik VM

- Port to Thumb2 and ARM Architecture v7A
- Optimization of Dalvik interpreter
- NEON and VFP optimizations

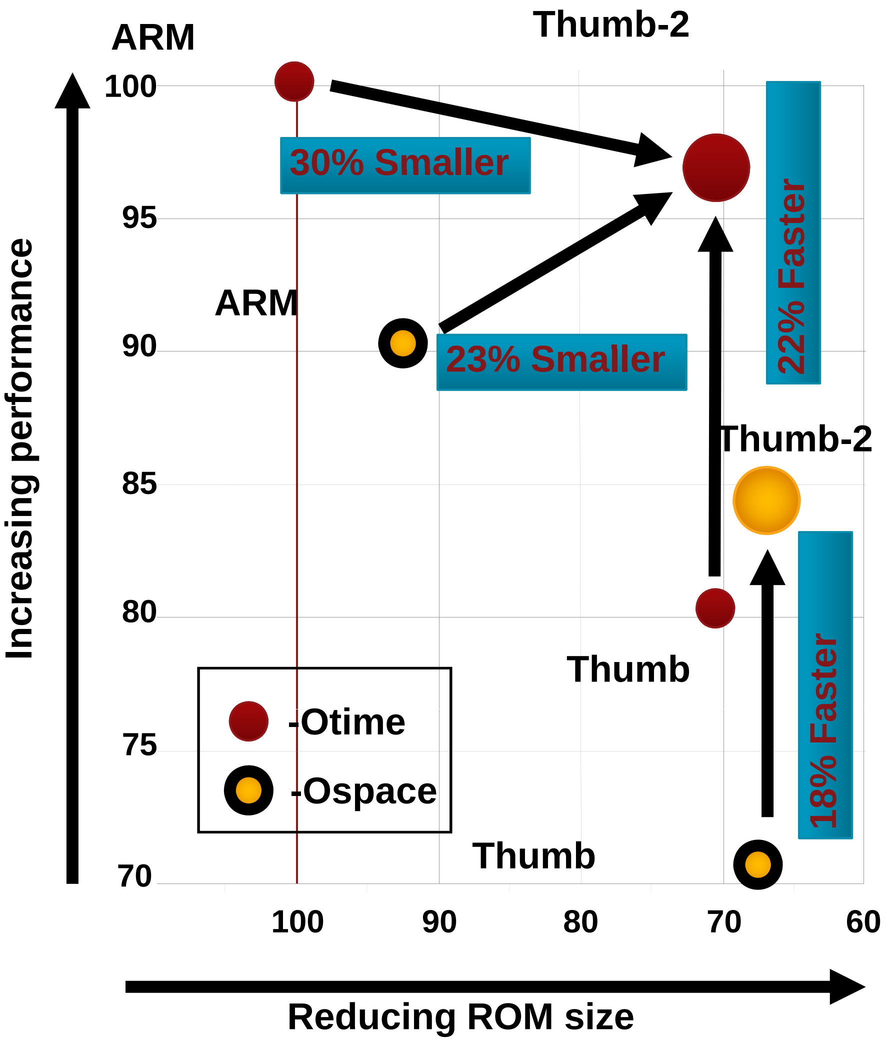
Android Architecture Evolution on ARM



Thumb-2 Zero Wait State Memory

- Improves code size when starting from ARM code
- Improves performance when starting from Thumb code

Benchmarking	Perf	Size
`-Otime ARM	100	100
`-Otime T-2	96	71
`-Otime T-1	79	71
`-Ospace ARM	90	92
`-Ospace T-2	85	68
`-Ospace T-1	72	67

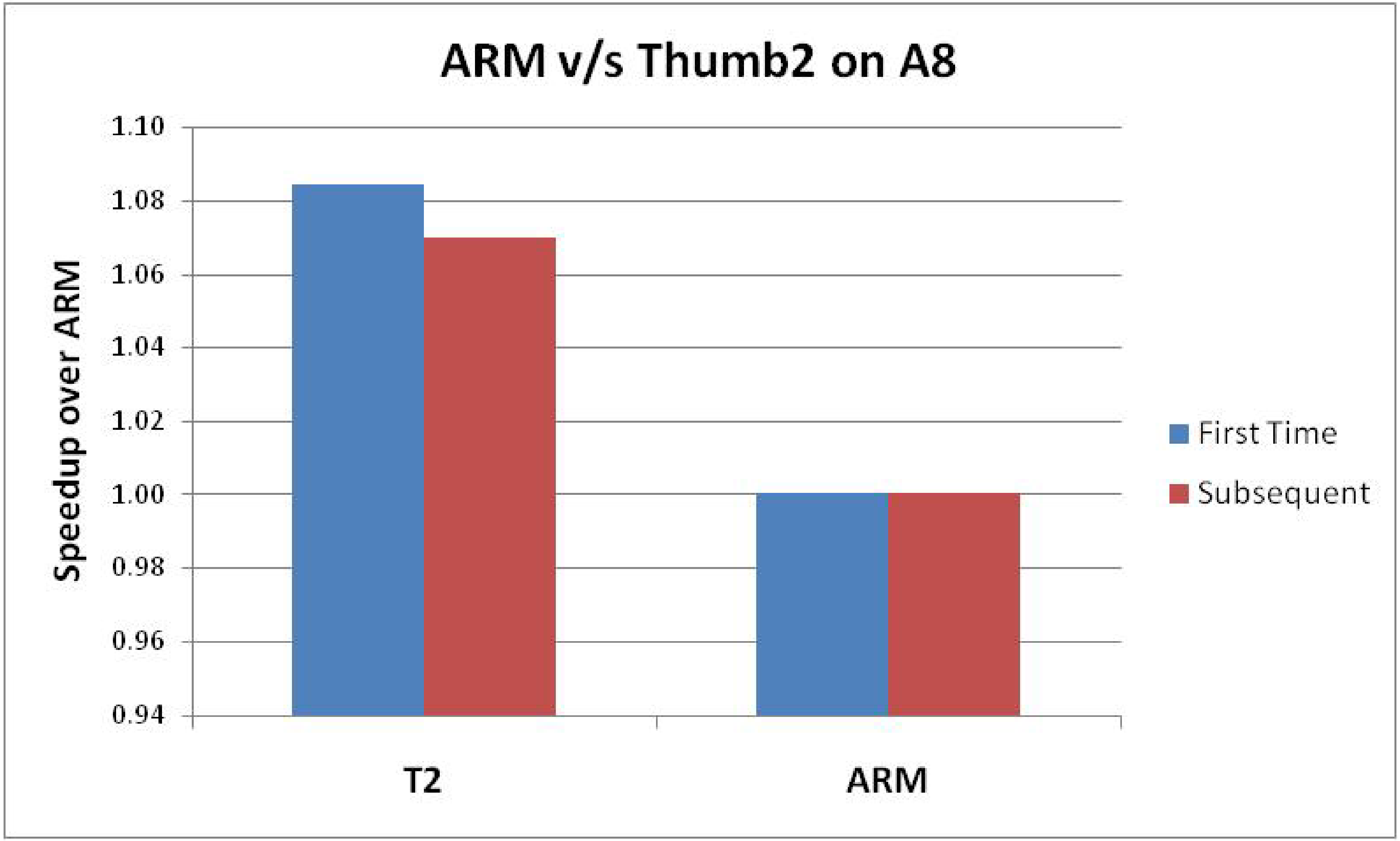


Performance numbers based on uncertified EEMBC numbers
Code Size Based On Real World Application Data

- 46 benchmarks
- 48 applications
- 9 Mbytes ROM

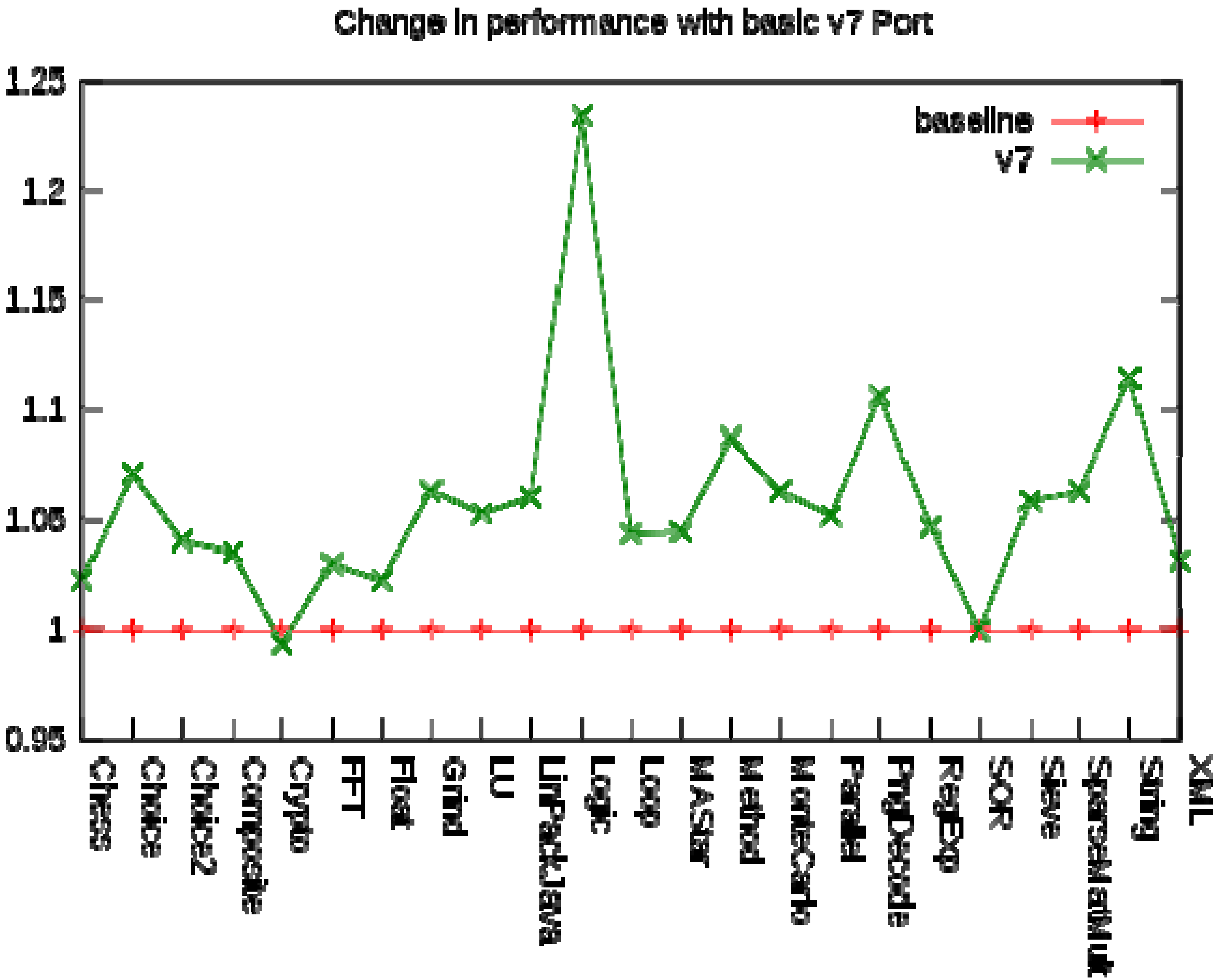
All numbers based on ARM1156T2 compiled using RVCT2.2.1 build 503

ARM vs T2 on Cortex-A8 Real Memory



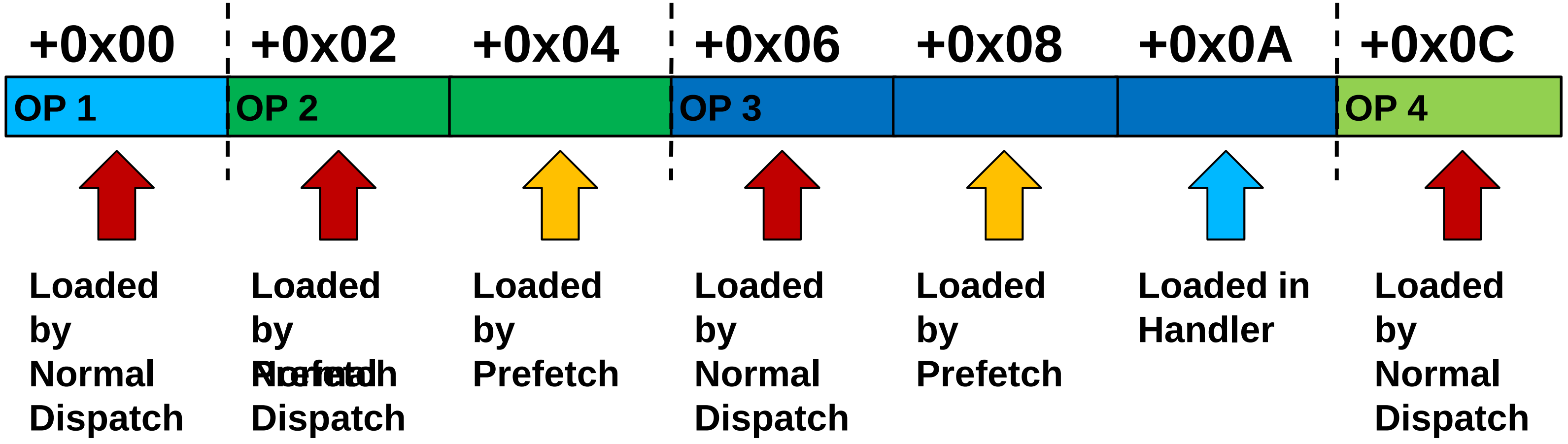
On real memory systems with latency, the additional instructions in I-cache make Thumb2 higher performance than ARM

Porting Android/Dalvik to Thumb2



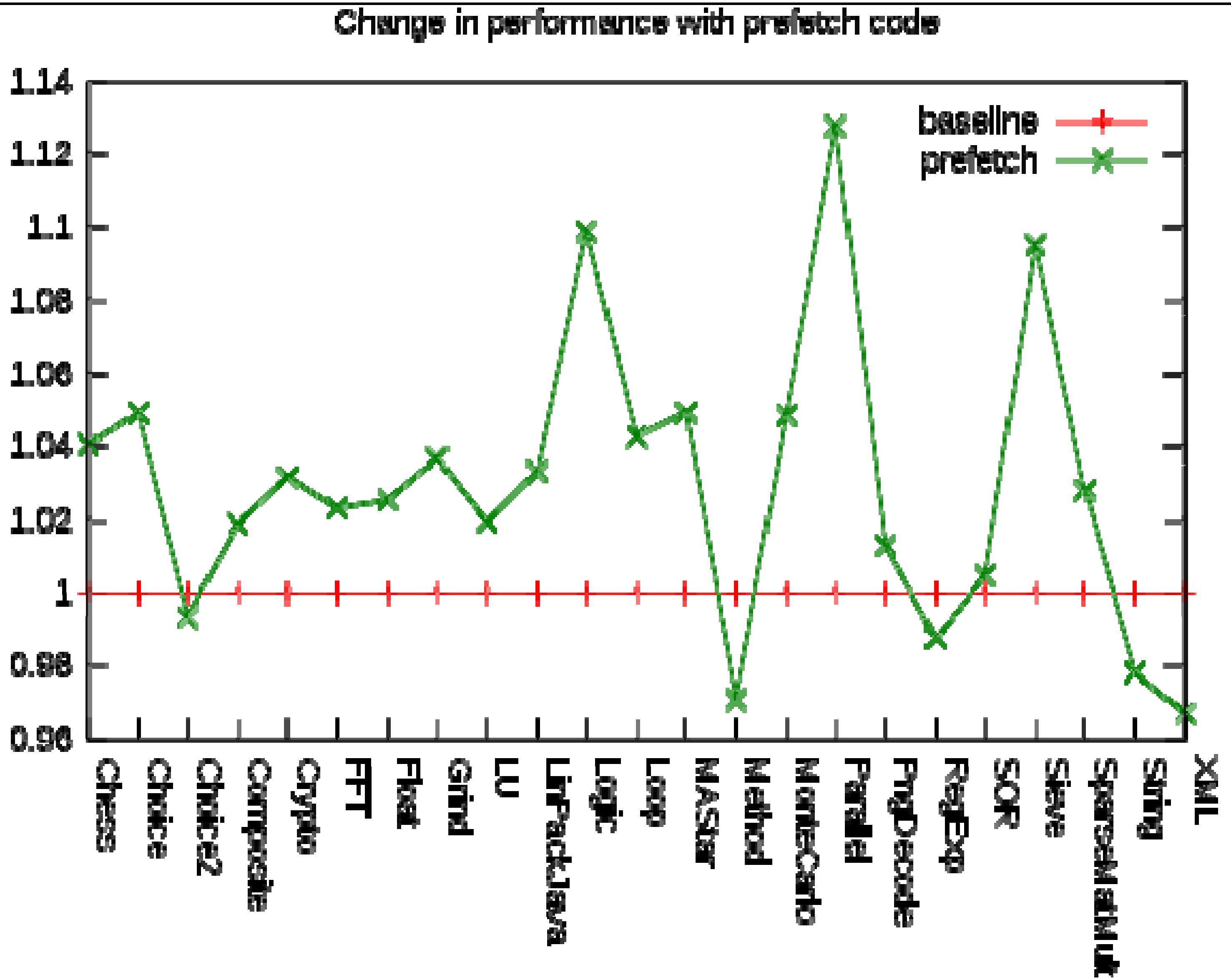
Prefetch Optimization

- The Dalvik instruction dispatch fetches a half-word at a time
  - One byte is the operation
  - One byte is usually data for that operation
- Many ops have one or more further half-words of data
- Frequently, the whole 'bytecode' is a word long
  - Requires another half-word load in the handler

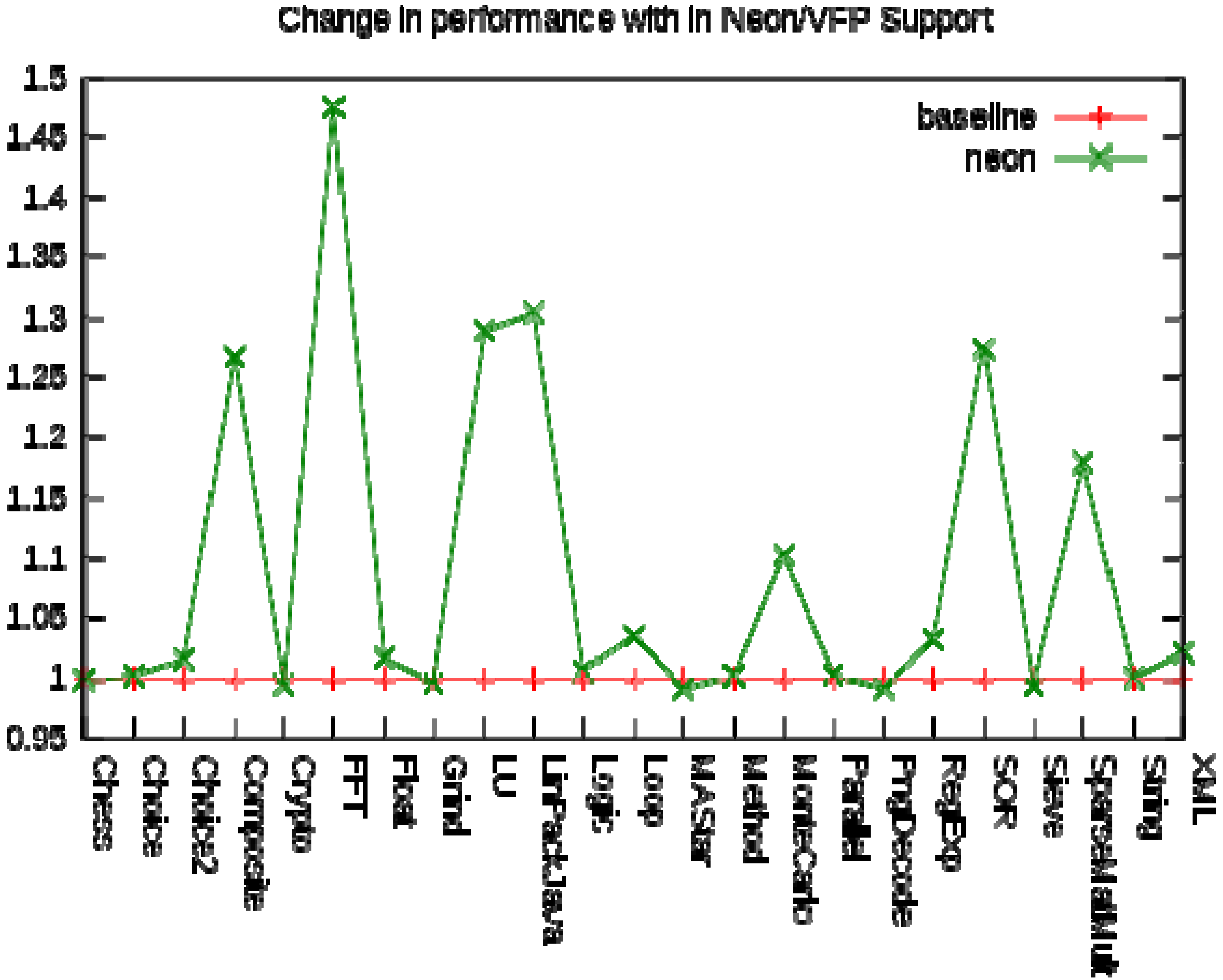


+0x00	+0x02	+0x04	+0x06	+0x08	+0x0A	+0x0C
OP 1	OP 2		OP 3			OP 4
Loaded by Normal Dispatch	Loaded by Normal Prefetch Dispatch	Loaded by Prefetch	Loaded by Normal Dispatch	Loaded by Prefetch	Loaded in Handler	Loaded by Normal Dispatch

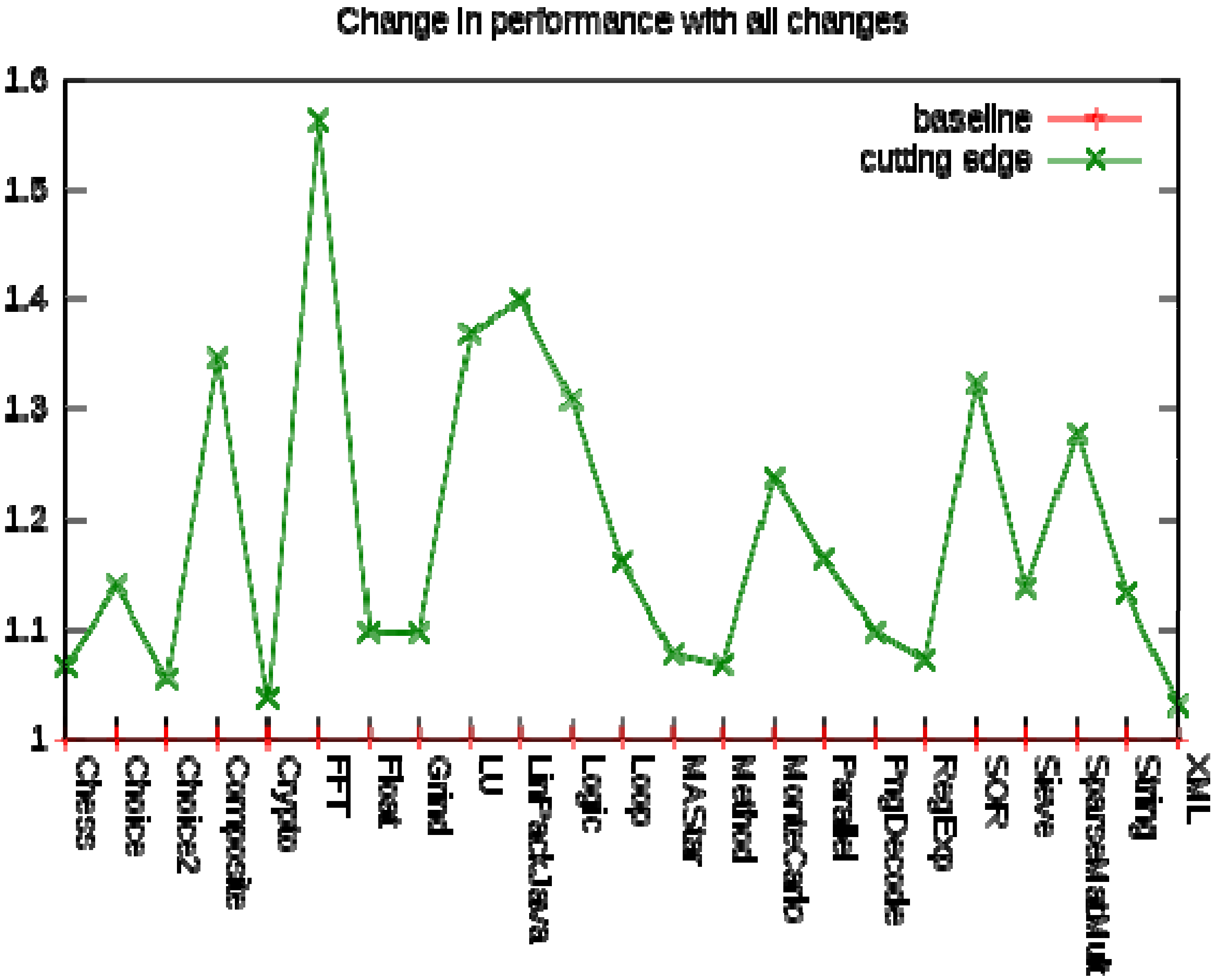
Prefetch Optimization of Dalvik VM



NEON / VFP Optimization of Dalvik



Overall Dalvik Optimization Result



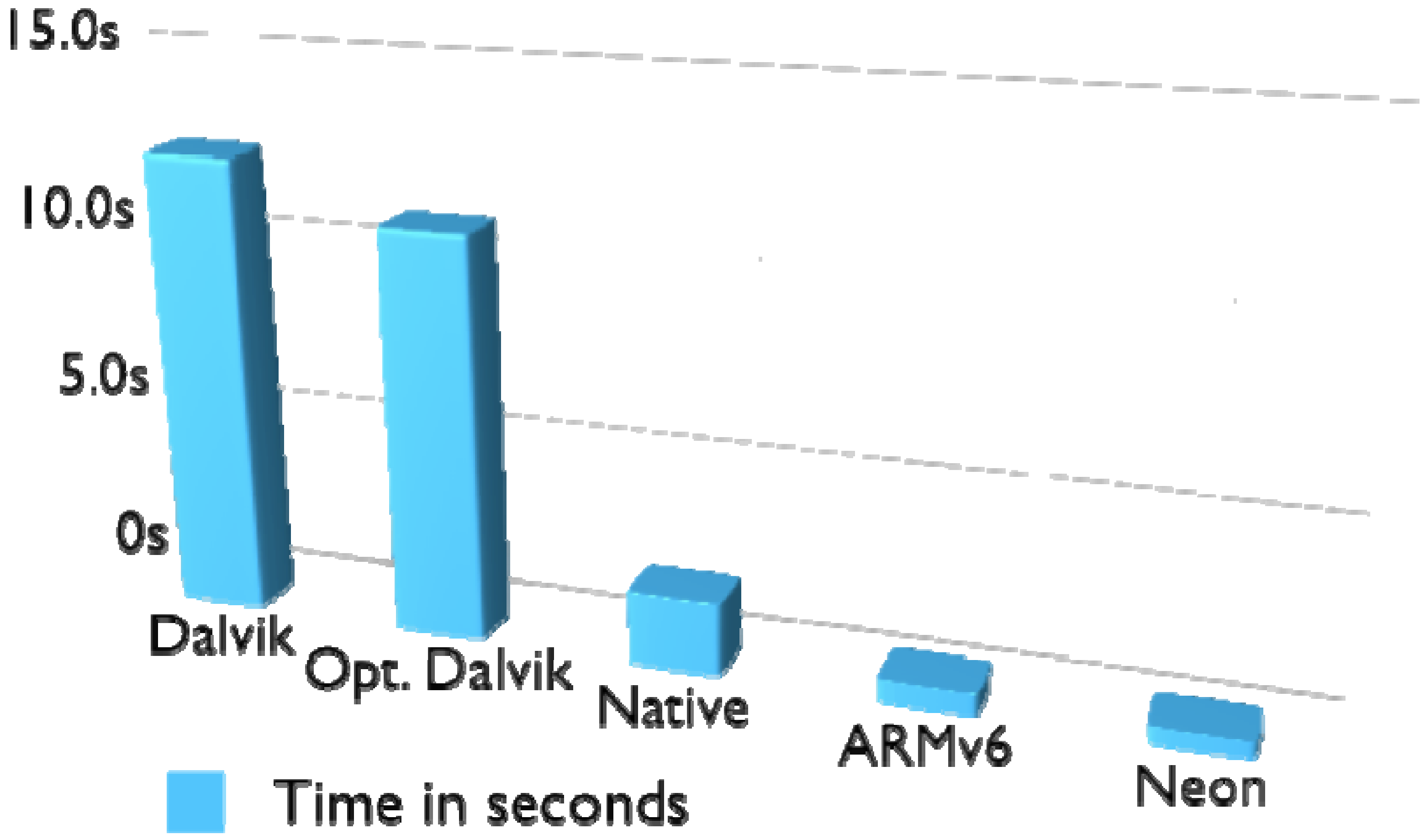
Native Development Kit (NDK)

- Based on GCC
  - Defaults to ARMv5TE
  - Supplied GCC version actually supports ARMv6 and VFP
  - Possible to overide to support v6 / VFP in Makefiles
  - Possible to replace GCC with newer GCC eg CodeSourcery with v7 support

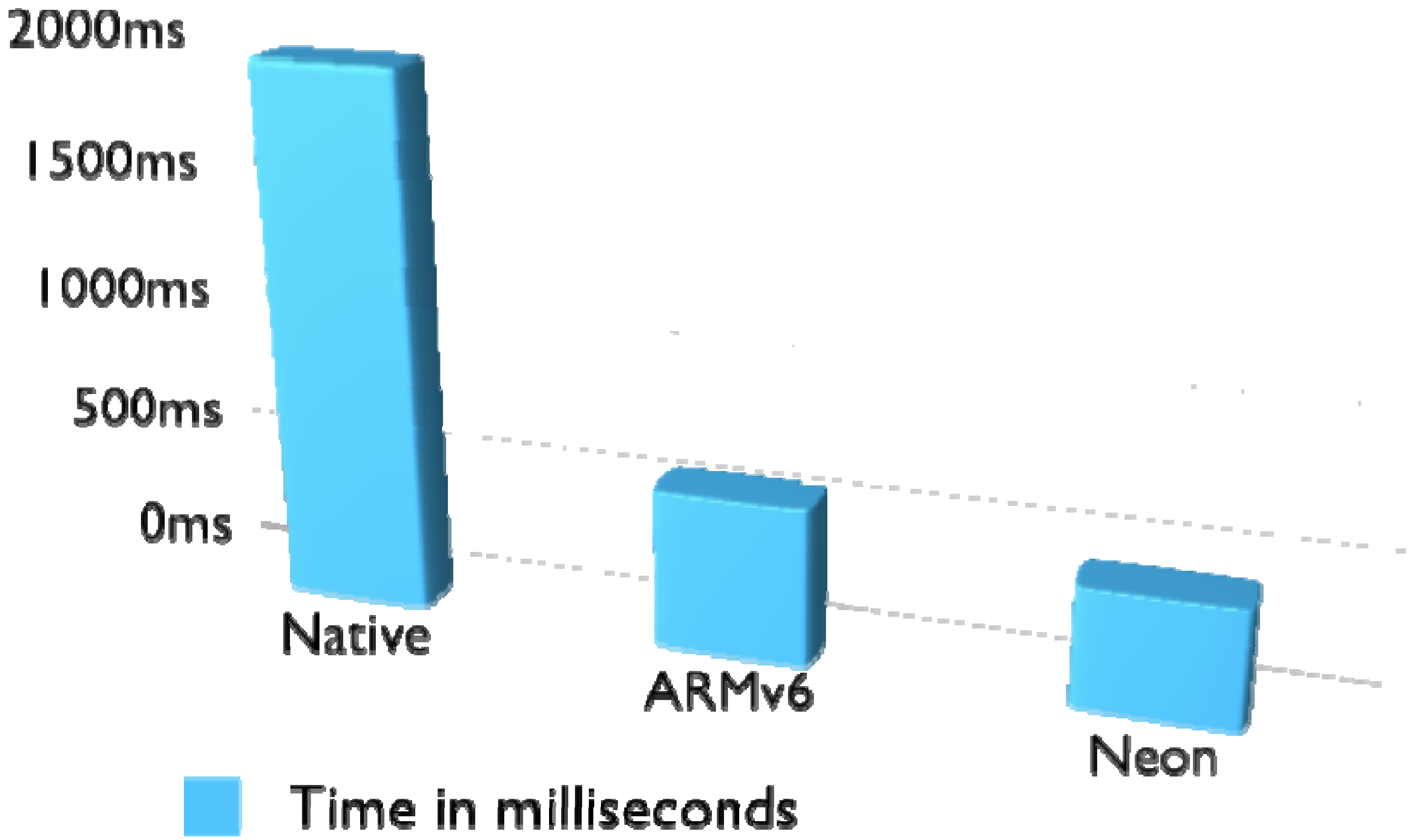
- Uses JNI (Java Native Interface)
  - Enables compiled code from C/C++ or assembler be called (or call) Java VM

- Best used for speeding up critical code sections
  - Code that touches hardware will not be portable
  - Good examples is iterating over arrays (eg video/audio)
  - Use supplied audio and video libraries & frameworks where possible

Benchmark Results



Benchmark Results - Just Native



DESIGN TO THE POWER OF THREE

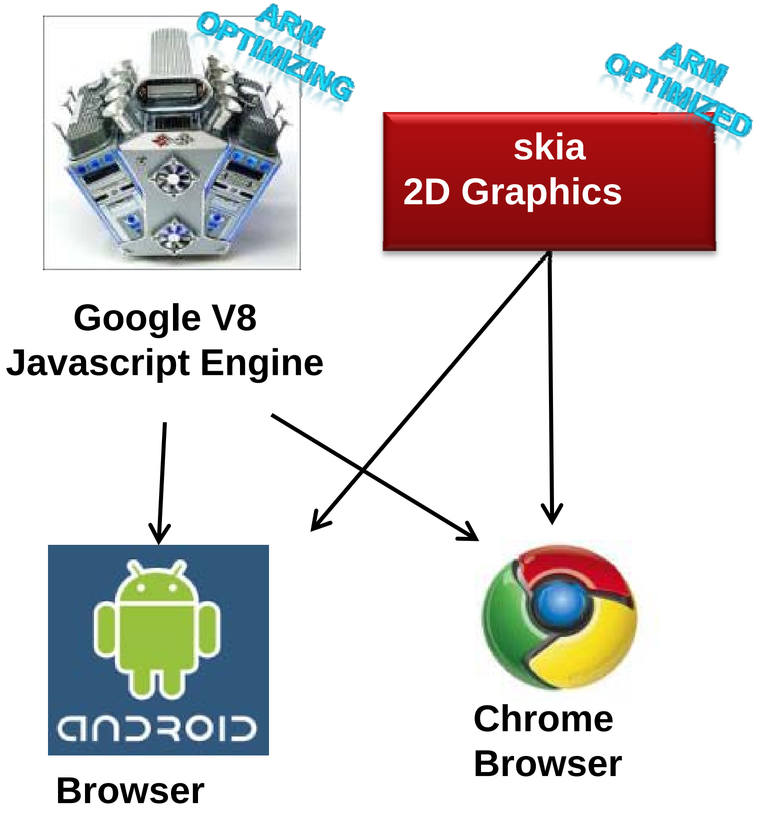
ARM Continuing Investment in Android

- NEON
  - Skia: Delivering 30% improved UI speed today
  - Pixelflinger software renderer NEON optimizations next
  - NEON for CODEC support

- Google V8 Javascript engine optimization

- SMP support
  - SMP compliance testing
  - SMP optimization

- Android is becoming even more optimized for the ARM architecture taking advantage of the latest and greatest features



skia
2D Graphics

Google V8
Javascript Engine

Browser

Chrome
Browser

Skia Graphics Engine Optimizations

- Graphics core for ***both*** Chrome and Android
- Includes SW implementation of OpenGLES1.1 (libagl)
  - Will load a hardware accelerator if present (libhgl)
- ARM optimized Skia using NEON intrinsics (and assembler)
  - Provides ***10%-30%*** performance uplift on browser use-cases
- ARM will release to Skia Open Source pool shortly
- OpenGL ES 2.0 API coming in Éclair NDK

Where to Get SDK / NDK Components

- Install Eclipse:
  - http://www.eclipse.org/downloads/

- Install the Android SDK and Eclipse Plugin
  - http://developer.android.com/sdk/index.html

- Run SDK and select packages to download
  - You can you Android 1.5, 1.6, 2.0

- Install and configure the NDK (only 1.6 at present)
  - http://developer.android.com/sdk/ndk/1.6_r1/index.html

Summary

- Google investing in ARM Architecture because they know that ARM is the key to the Internet beyond the PC
- ARM investing in optimizing Android on ARM to further improve user experience
- Easy to use SDK enables developers to quickly develop apps
- NDK enables native code optimization for ARM