Android Debugging ART

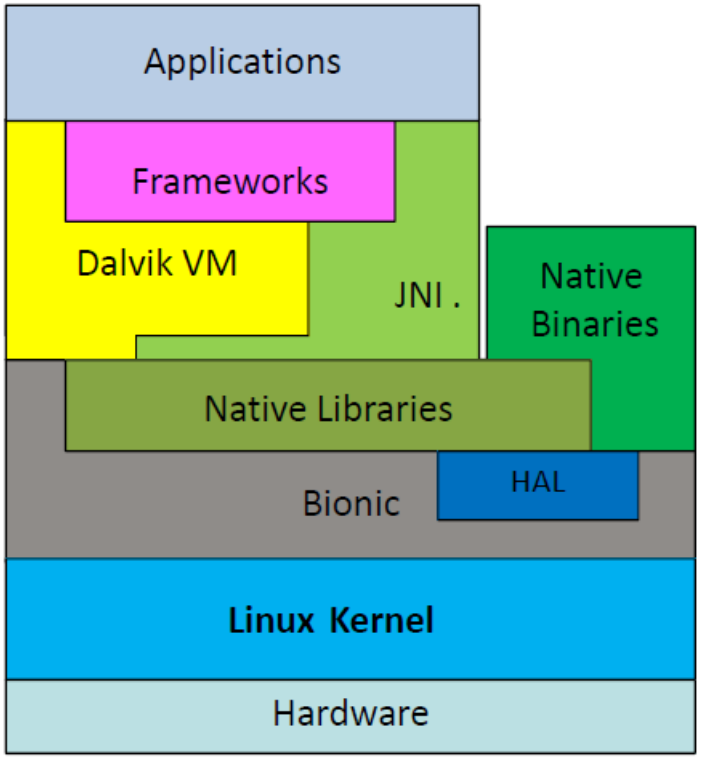
▪ Khaled JMAL ▪

2016 / 11 / 17

LAUTERBACH
DEVELOPMENT TOOLS

The Dalvik Virtual Machine

- Up to version 4.4 „KitKat“, Android was based on the Dalvik Virtual Machine
- Java compiles into DEX code
- DEX code is compiled just-in-time or interpreted by the Dalvik VM interpreter



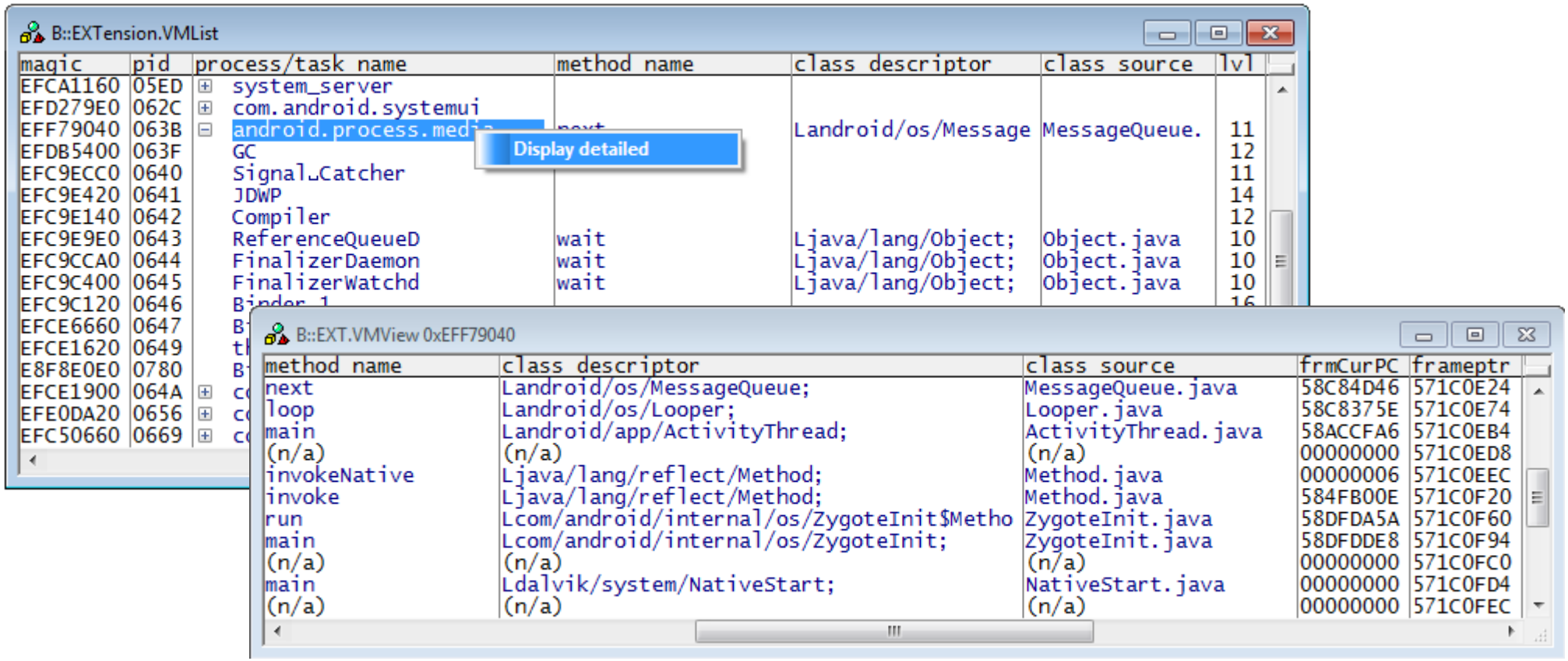
Source: „Dalvik and ART“, Jonathan Levin

The TRACE32 Dalvik Awareness

- Written by HAP
- Based on the symbols of the Dalvik VM (libdvm.so)
- Features:
  - List VM threads with their class descriptors and names
  - Display the VM stack for a single VM thread
  - List the Dalvik (Java) source code for each method
  - Display the stack frame with Java to native and native to Java transitions
  - Stepping through the DEX code not possible!

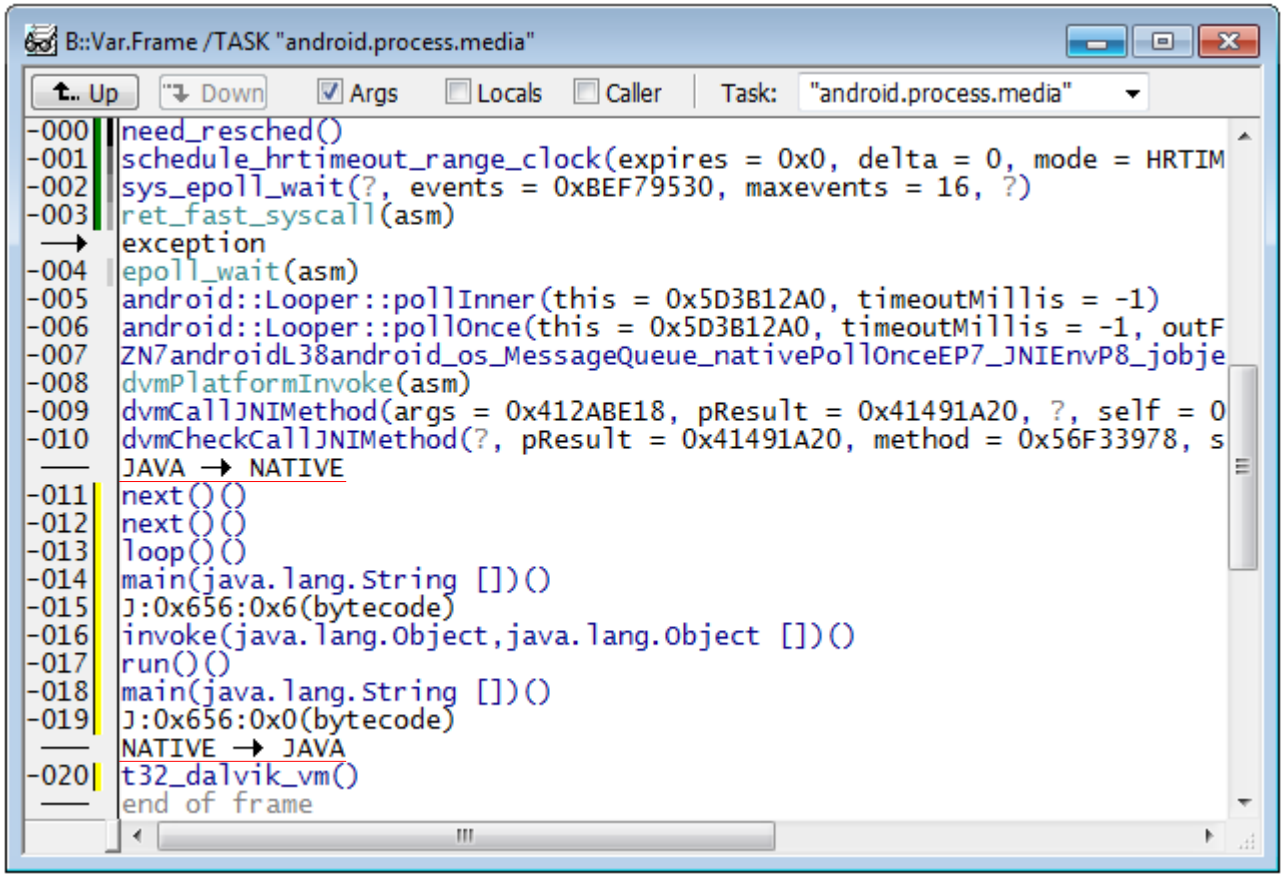
The TRACE32 Dalvik Awareness

- VM threads and thread stack



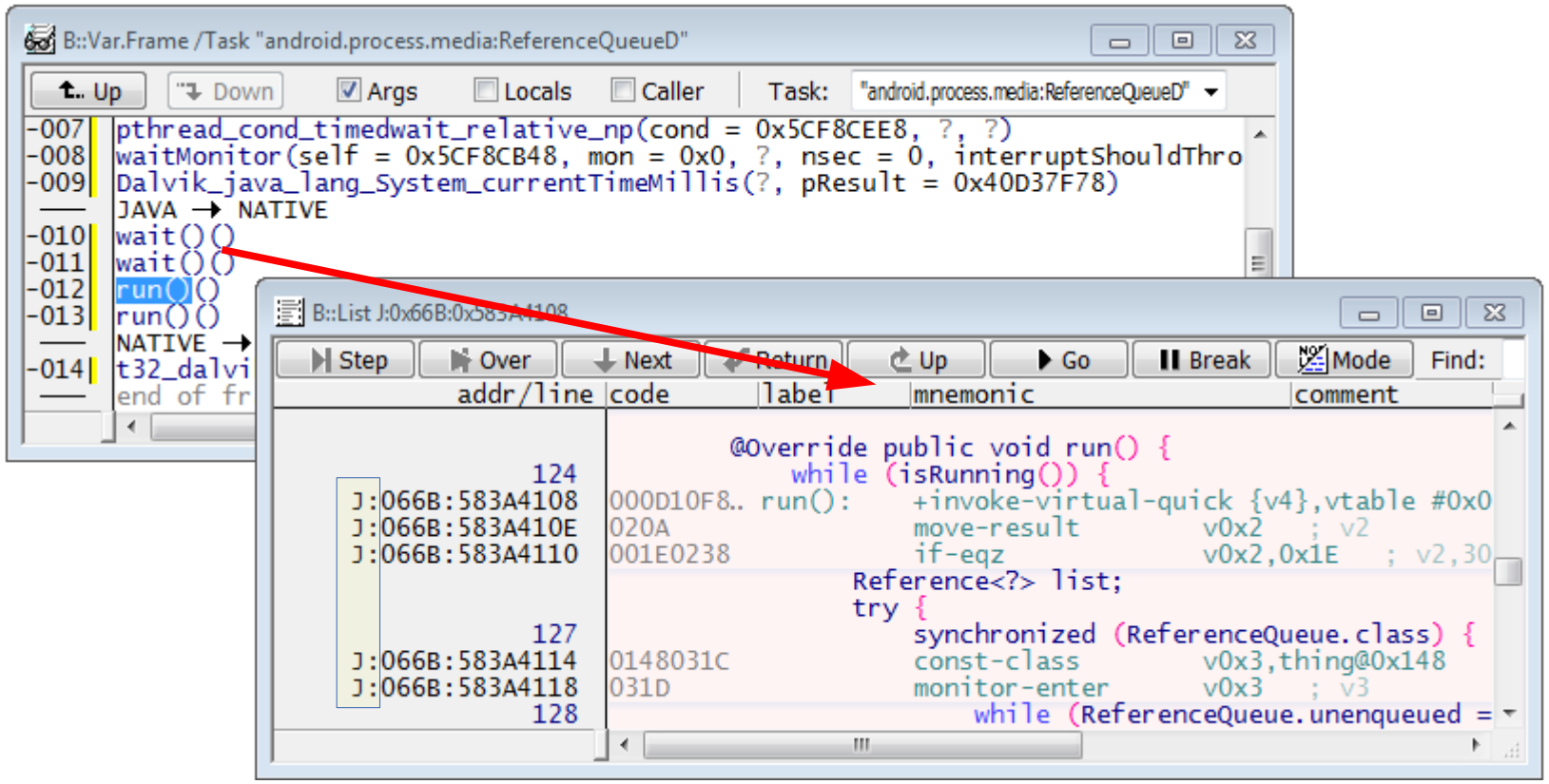
The TRACE32 Dalvik Awareness

- JAVA to NATIVE / NATIVE to JAVA transitions



The TRACE32 Dalvik Awareness

- DEX code disassembly

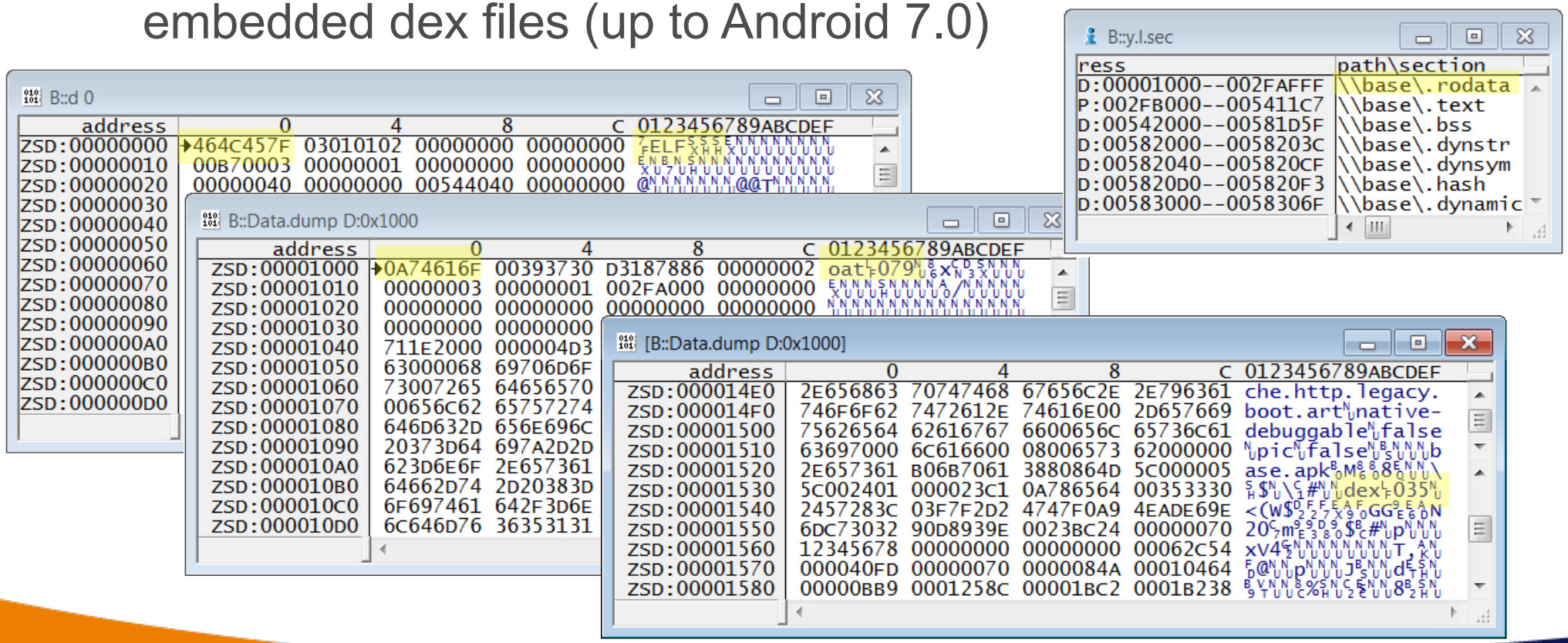


The Android RunTime (ART)

- Introduced in 4.4 „KitKat“ (available only through developer options)
- Supersedes Dalvik since 5.x Lollipop
- Introduces ahead-of-time (AOT) compilation instead of just-in-time (JIT)
  - Android Framework compiled to native code when building Android
  - Apps are compiled at install time

The Android RunTime (ART)

- ART compiles DEX to native code (OAT files)
- OAT files are ELF files with additional OAT data and embedded dex files (up to Android 7.0)



The Android RunTime (ART)

- OAT files can also be compiled with DWARF debug info
- DWARF info can be enabled for apps before installation with

```
$ adb shell setprop debug.generate-debug-info true
```

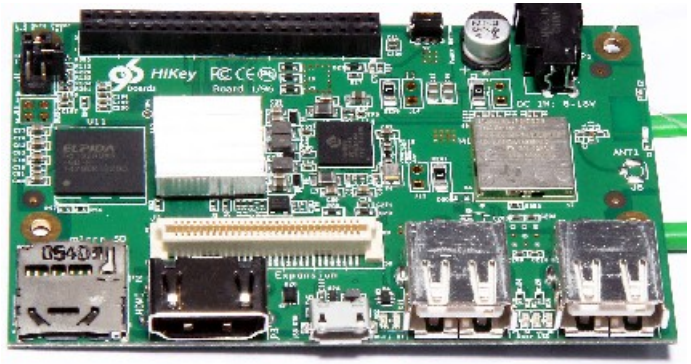
- Most debug info can also be extracted from the OAT data

  → new TRACE32 command: Data.LOAD.OAT

- Autoloader first tries to load the OAT file as ELF using Data.LOAD.ELF
- If no debug info is found then the Autoloader loads the files using Data.LOAD.OAT

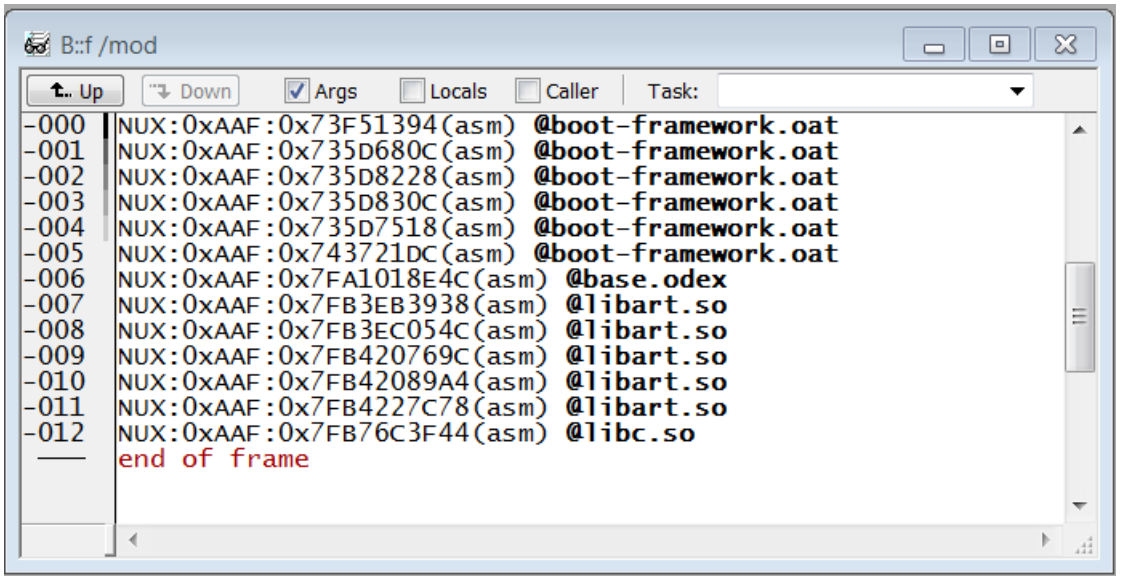
The Android RunTime (ART) - Demo

- RAM Dump from the Hikey board (ARM Cortex-A53) Loaded on the Simulator
- Android 6.0 Marshmallow



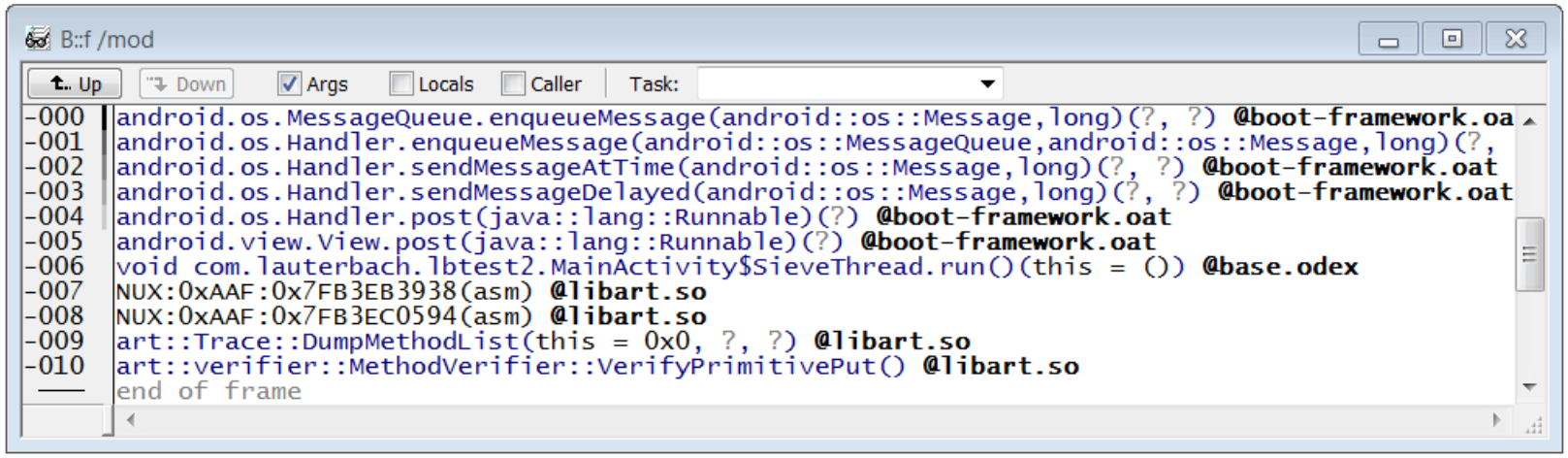
The Android RunTime (ART) - Demo

- Stack frame for the current task (sieve app) before loading the symbols. The app code has been compiled to base.odex and Android framework to boot-framework.oat. Both files are OAT files.

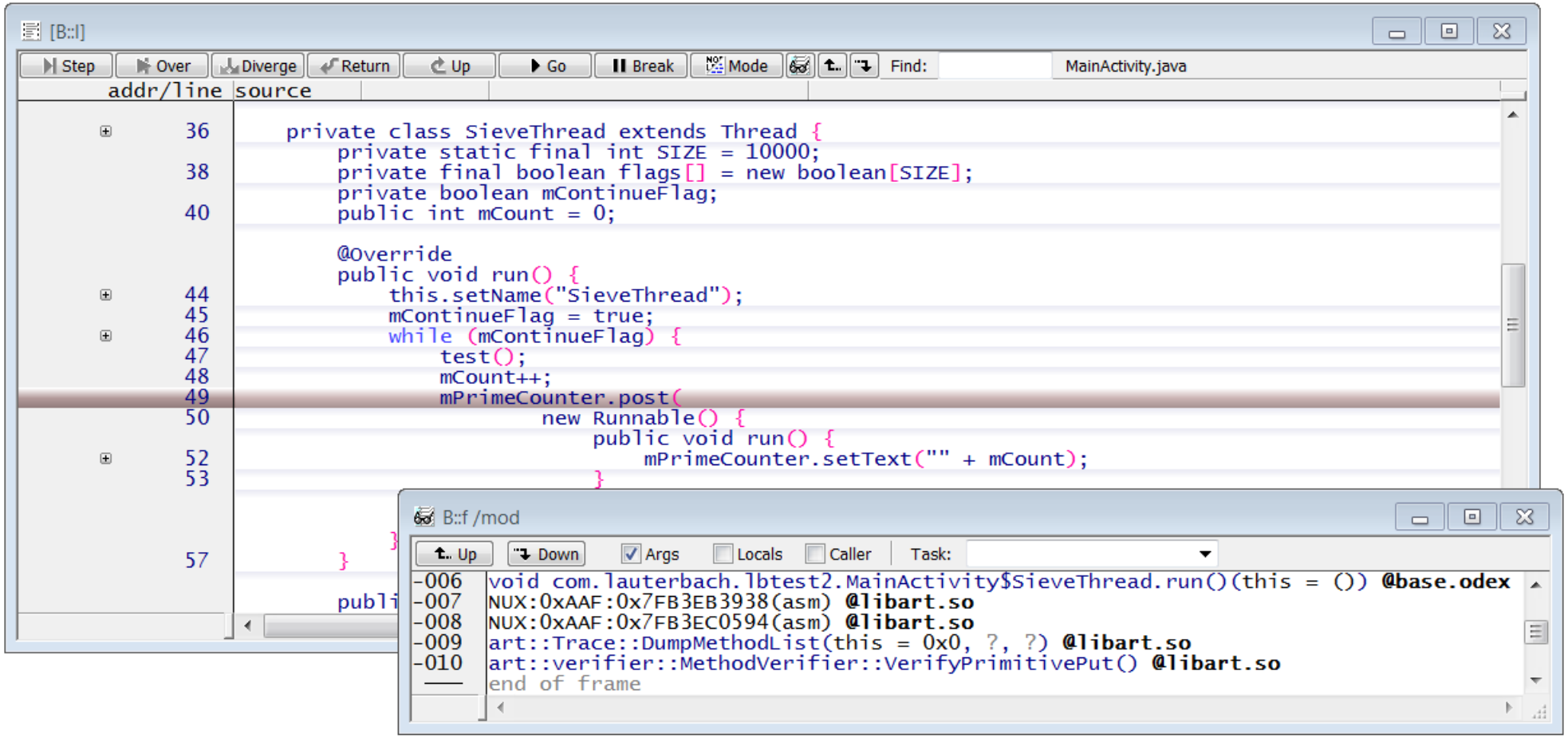


The Android RunTime (ART) - Demo

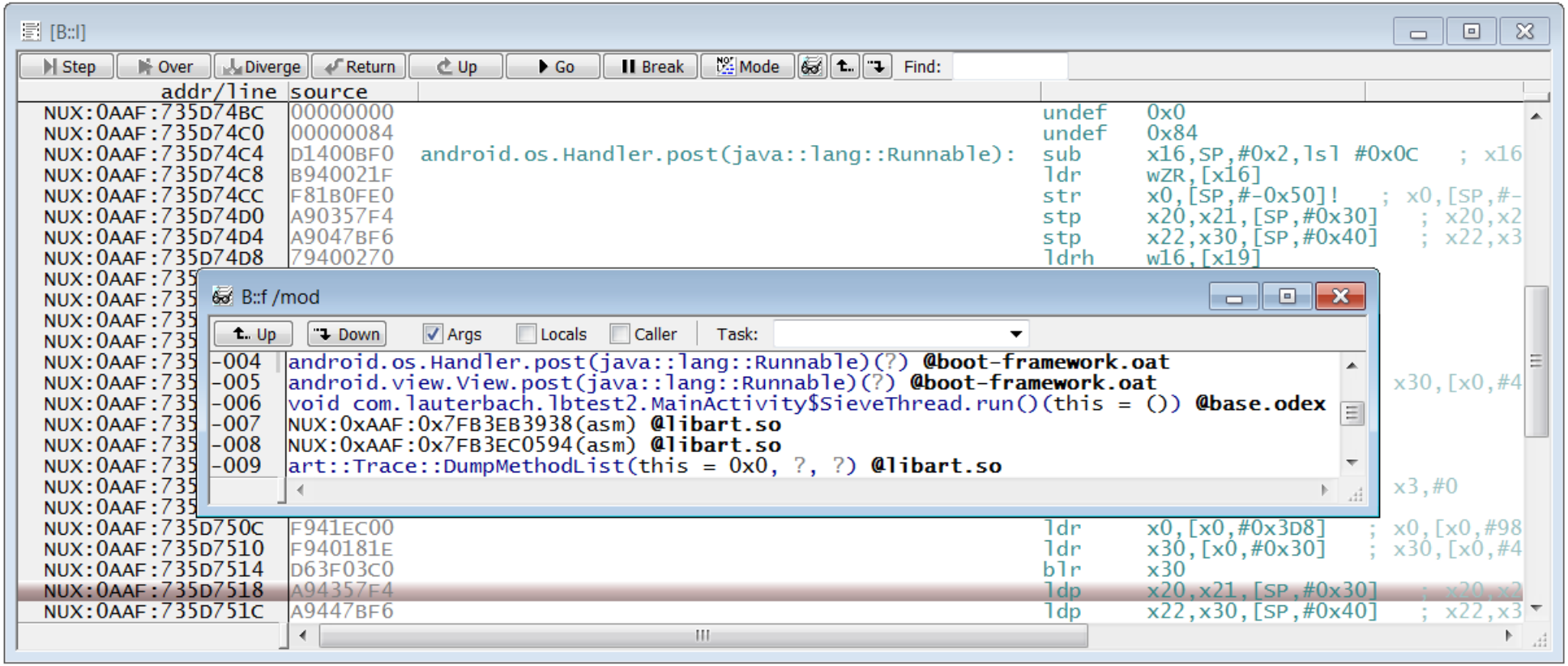
- The symbol autoloader load base.odex as ELF/DWARF since it contains DWARF info and boot-framework.oat as OAT.



The Android RunTime (ART) - Demo



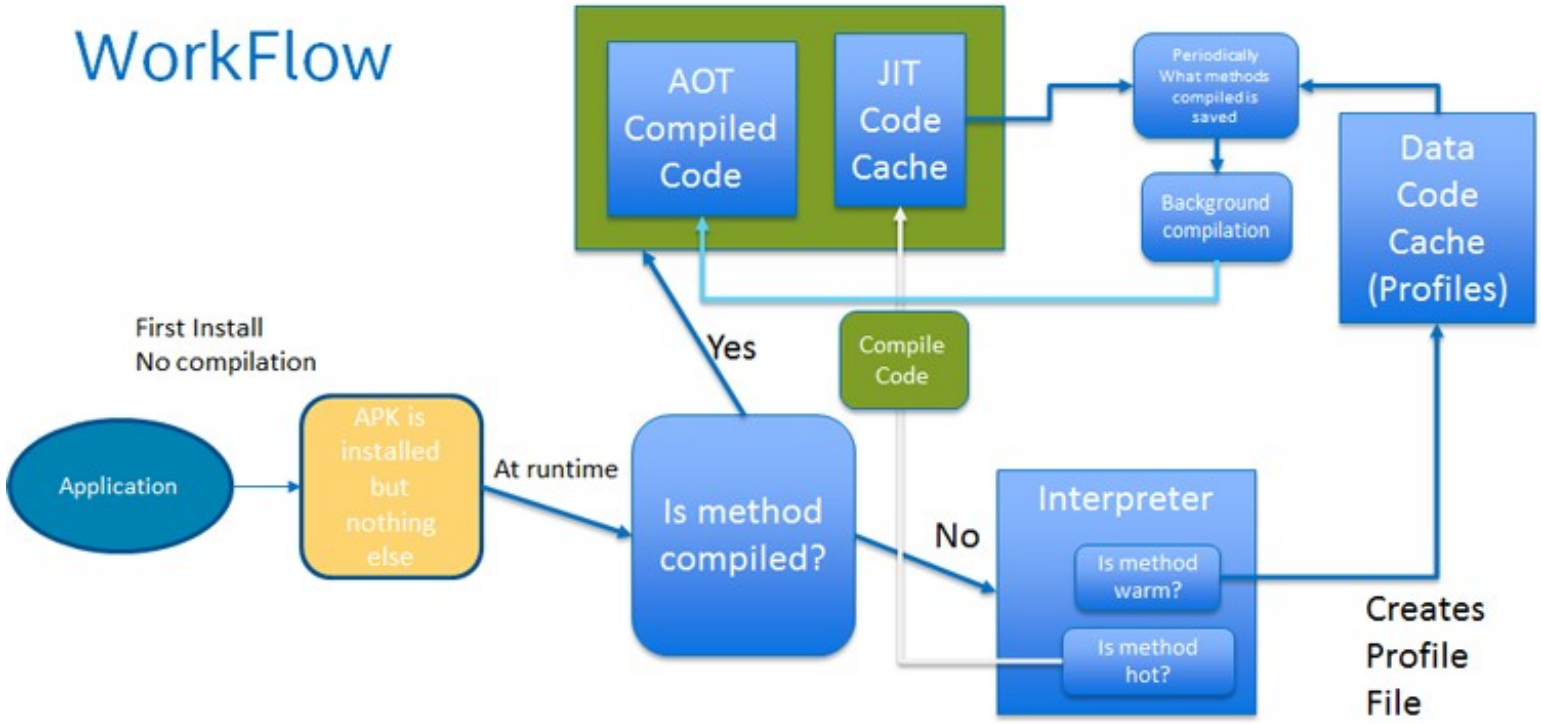
The Android RunTime (ART) - Demo



ART and Hybrid Compilation

- Android 7 (Nougat) introduces JIT/AOT hybrid compilation
- The Android Framework is still compiled ahead-of-time
- Apps are per default not compiled at install time:
  - An interpreter initially runs all the byte code and profiles often-executed methods („hot“)
  - „hot“ methods are compiled by the JIT compiler into native executable code which stored in the JIT cache along with the collected profile information
  - When the device is unused and charging for over long duration, a service will compile the hot methods and save the generated code

ART and Hybrid Compilation



Source: Android: The Road to JIT/AOT Hybrid Compilation-Based Application User Experience By Rahul K. (Intel), Jean Christophe Beyler (Intel), Paul H. (Intel)

ART and Hybrid Compilation

- JIT can be disabled with following commands:

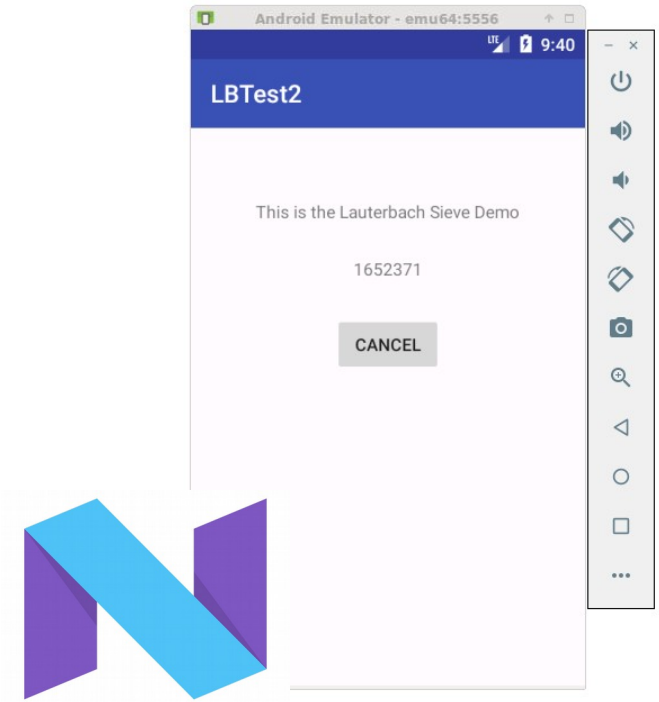
```
$ adb shell stop
$ adb shell setprop dalvik.vm.usejit false
$ adb shell start
```

- Ahead-Of-Time compilation for apps can be enabled using the following setup before installation:

```
$ adb shell setprop pm.dexopt.install everything
```

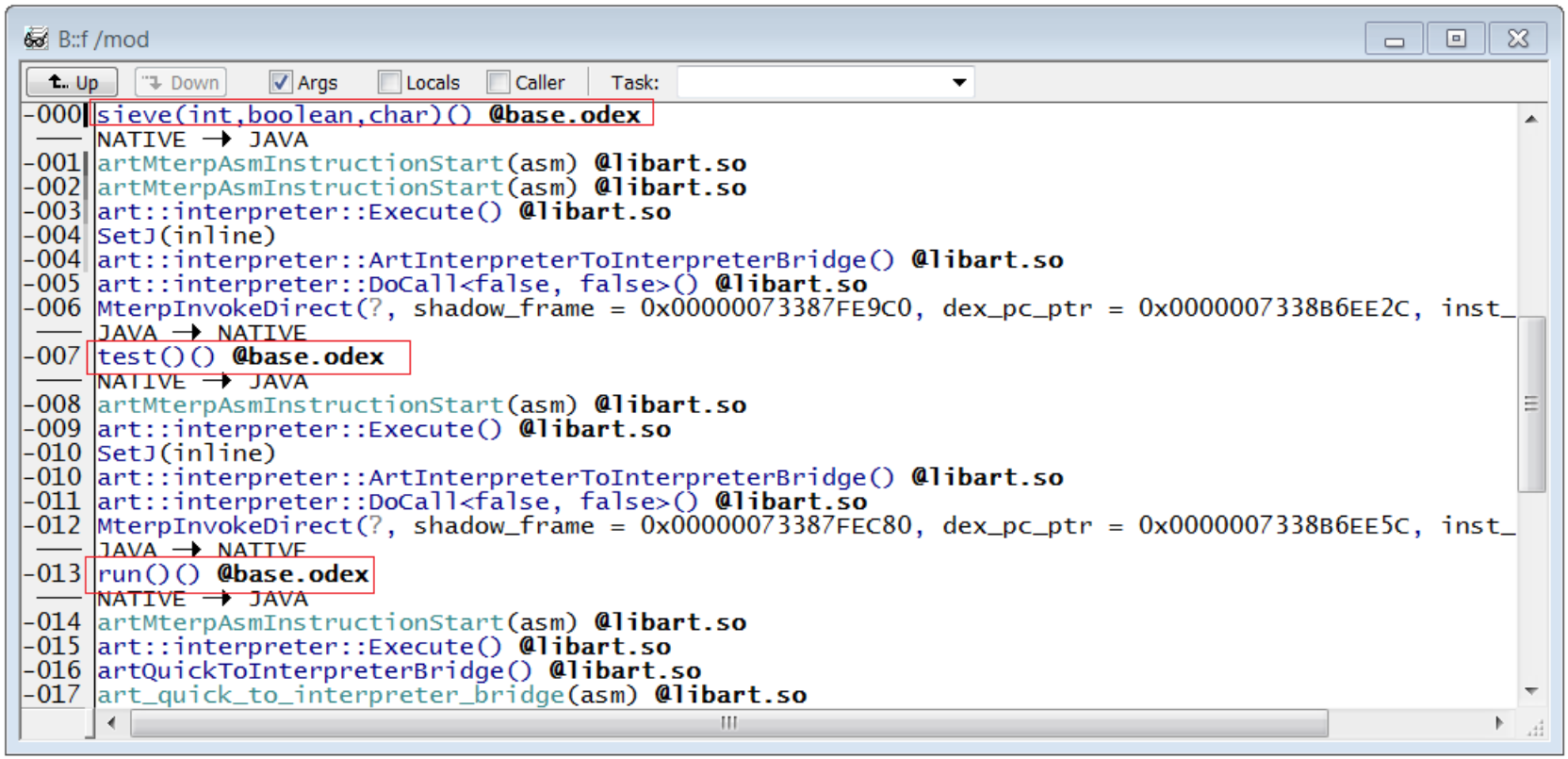
ART and Hybrid Compilation - Demo

- Android 7.0
- RAM dumps from the Android Emulator (QEMU)
- Disabled JIT



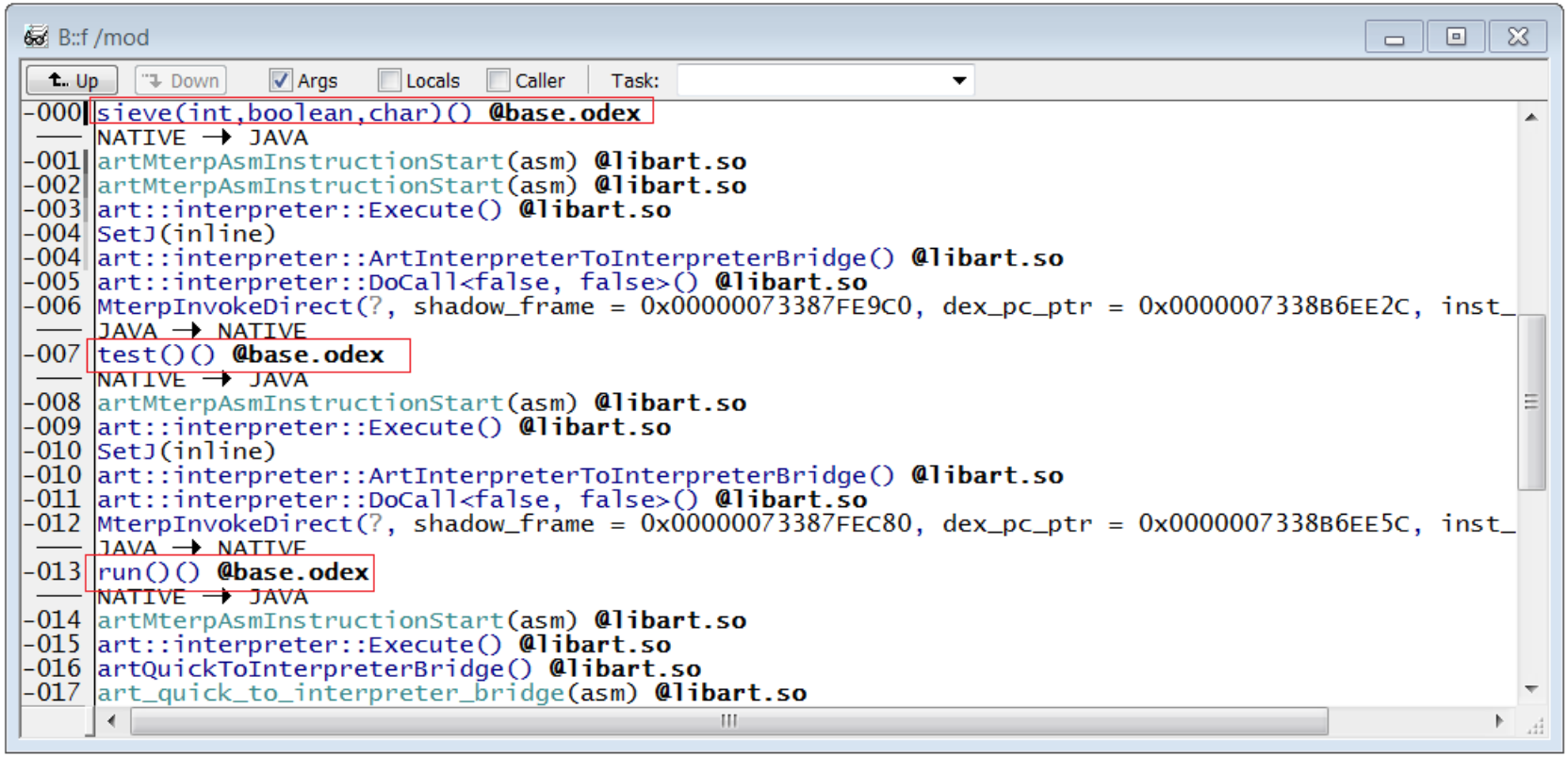
Android 7.0 – Enabled JIT

- TRACE32 can display the stack frame with the Java to native and native to Java transitions.



Android 7.0 – Enabled JIT

- TRACE32 can display the stack frame with the Java to native and native to Java transitions.

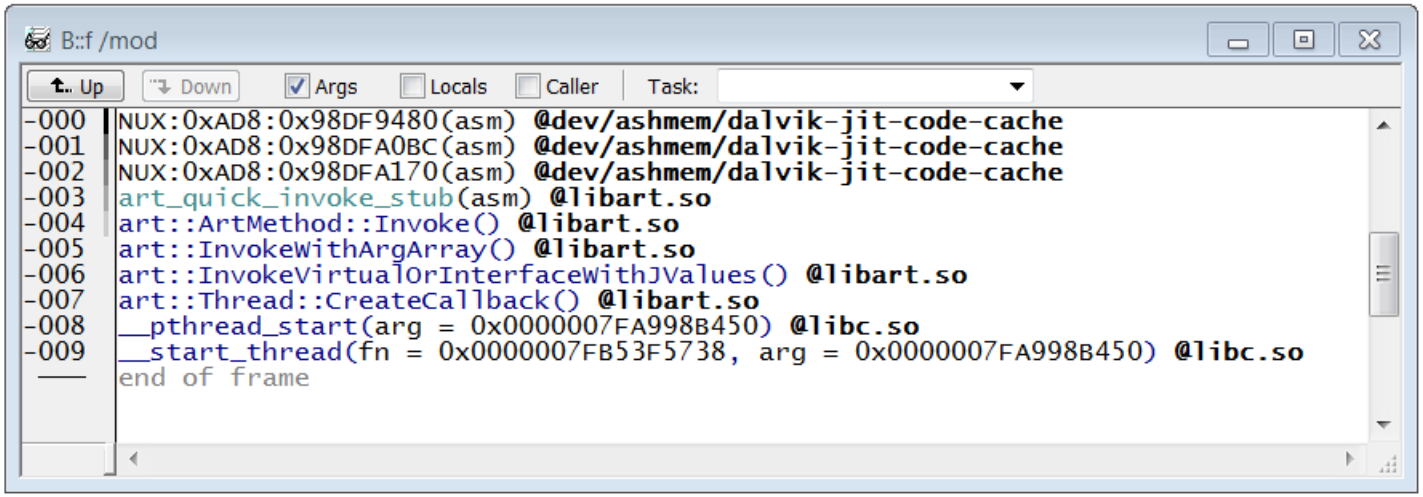


ART and Hybrid Compilation - Demo

- Android 7.1
- RAM dumps from HiKey board
- Enabled JIT

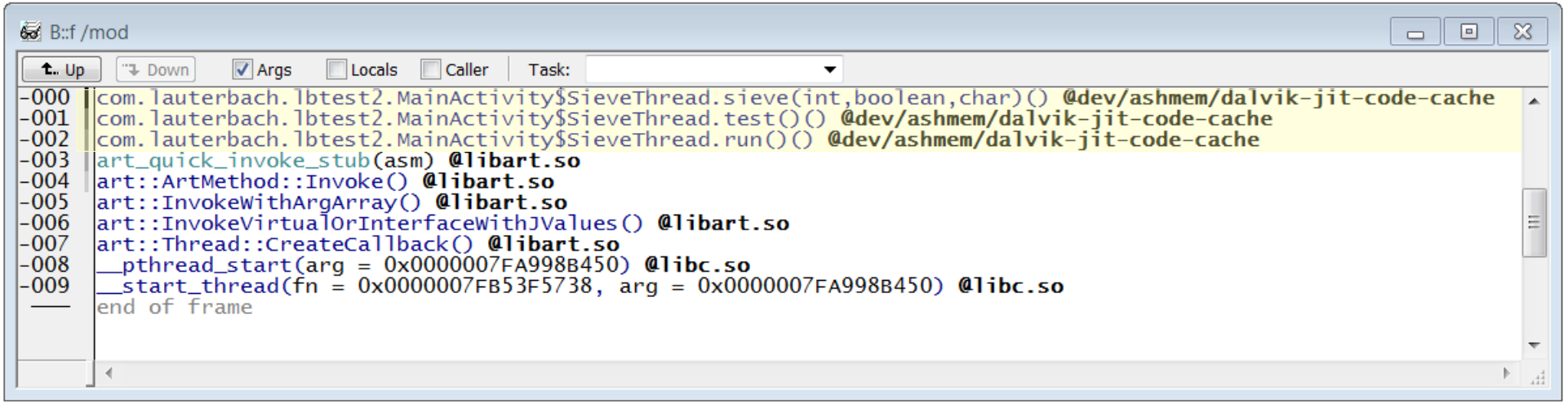
Android 7.1 – Enabled JIT

- Frame window shows that the application is executing in the dalvik jit code cache.



Android 7.1 – Enabled JIT

- Android ART Awareness can get the name of the methods corresponding to the jit code cache addresses **(work in progress)**



Thank you!

▪ Khaled JMAL ▪

2016 / 11 / 17

Questions?

LAUTERBACH
DEVELOPMENT TOOLS