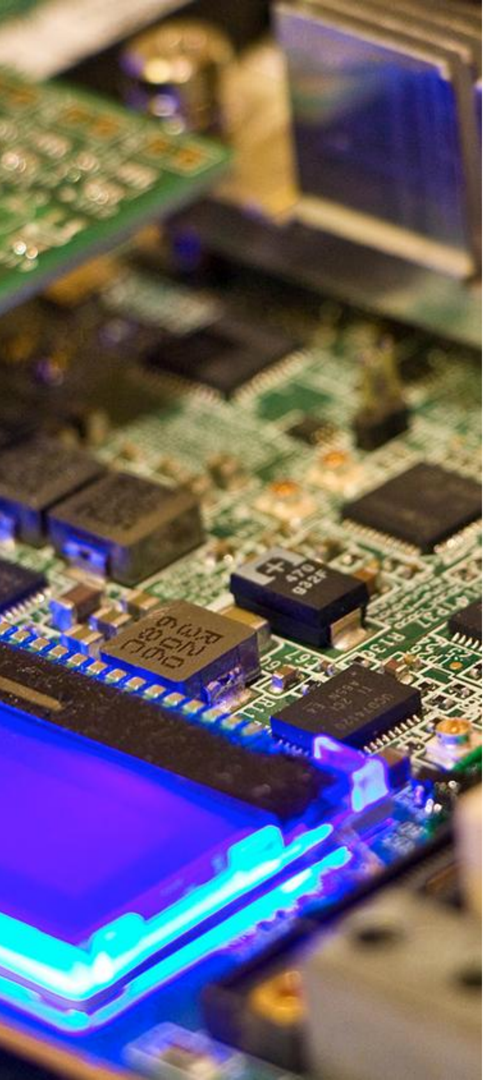
Bionic and musl - room for cooperation?

Presented by

Bernhard "Bero" Rosenkränzer

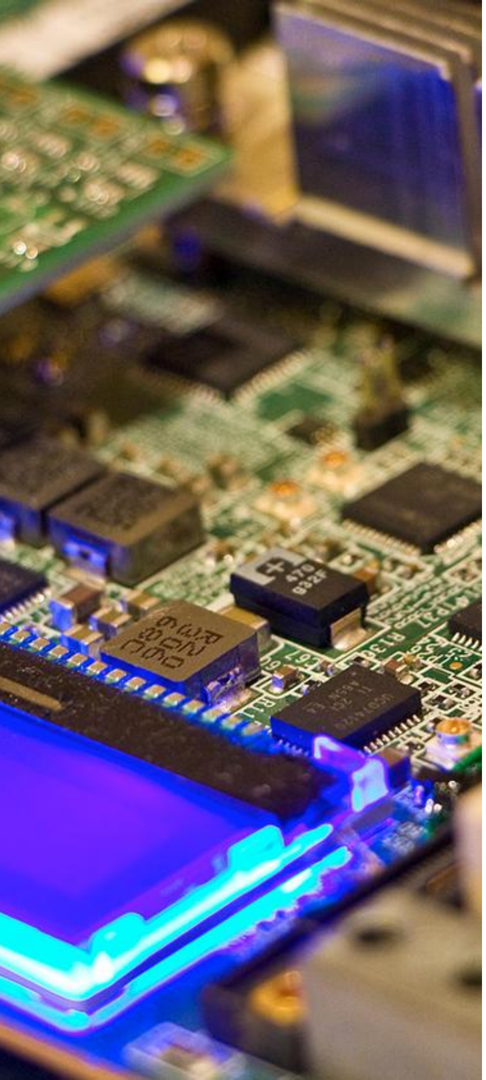
Date

Android Builders Summit 2015



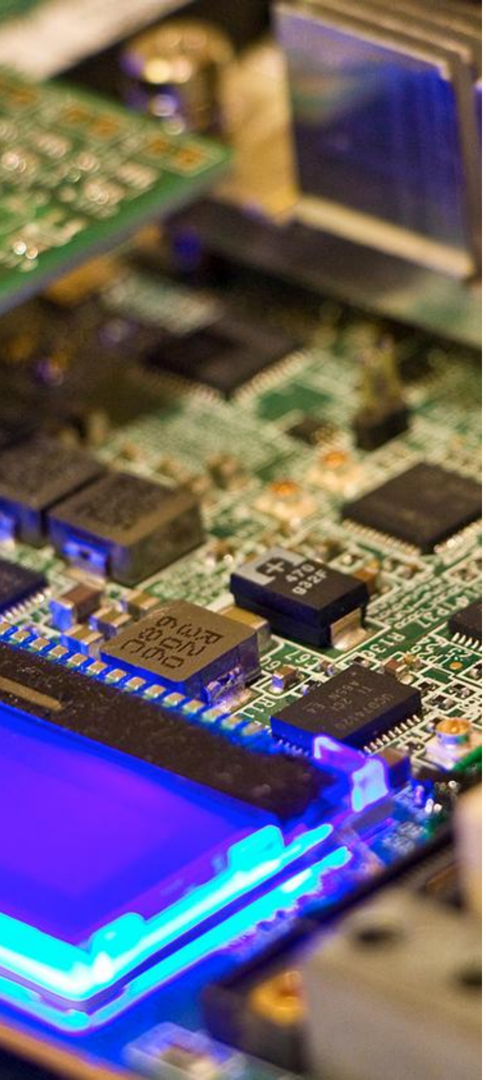
What are Bionic and musl?

- Bionic and musl are different implementations of the C standard library - along with glibc, uClibc, dietlibc, newlib, klibc and various other implementations.



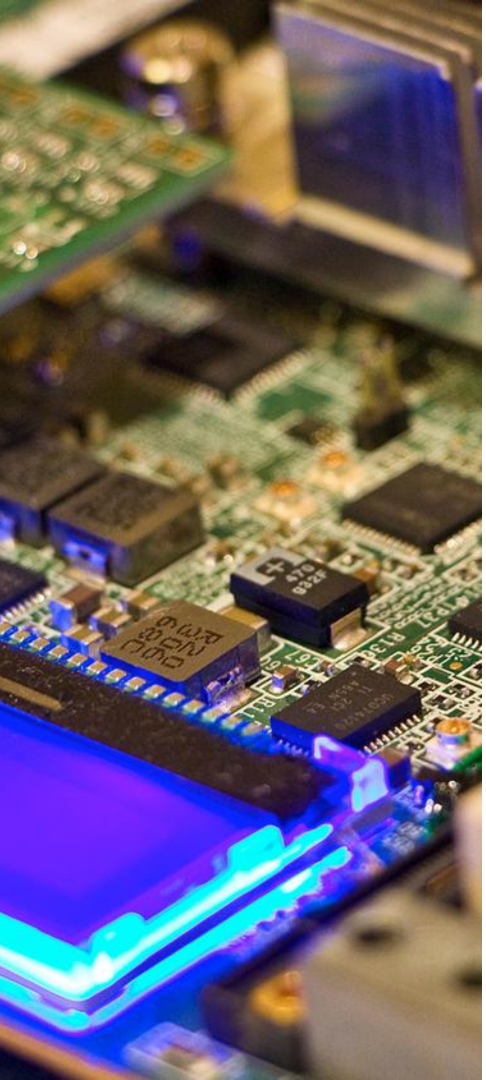
A bit of history

When Android started, no existing libc met the requirements: small, BSD licensed, runs on Linux kernel -- so Bionic was put together based on BSD libcs



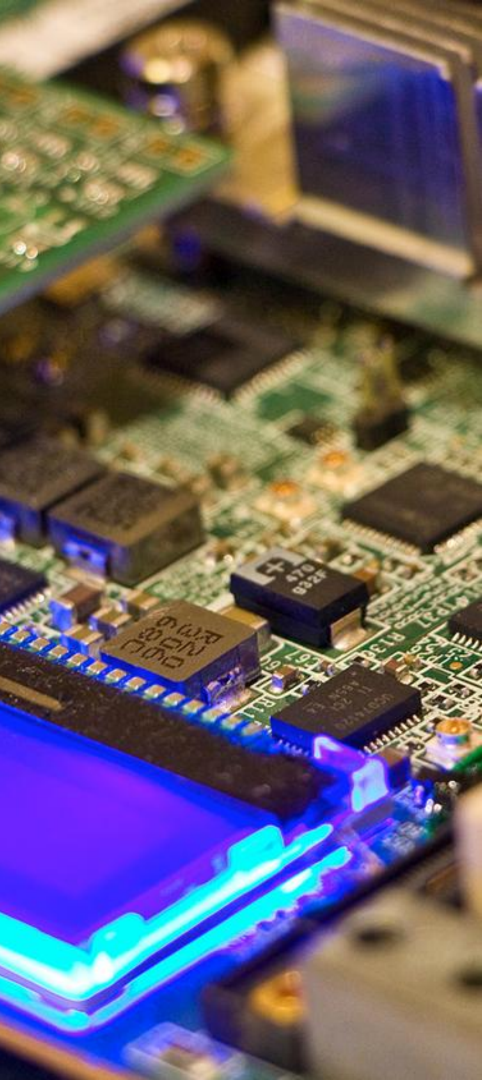
A bit of history

Today, musl fulfills the same requirements -- but there is no need to throw away existing Bionic, and ABI compatibility is important



Bionic's structure

Bionic is derived from several BSD libcs, and designed for pulling in their improvements already:



Bionic's structure

Source tree:

`libc/bionic`

  bits implemented for Bionic itself

`libc/upstream-freebsd`

  functions from FreeBSD libc

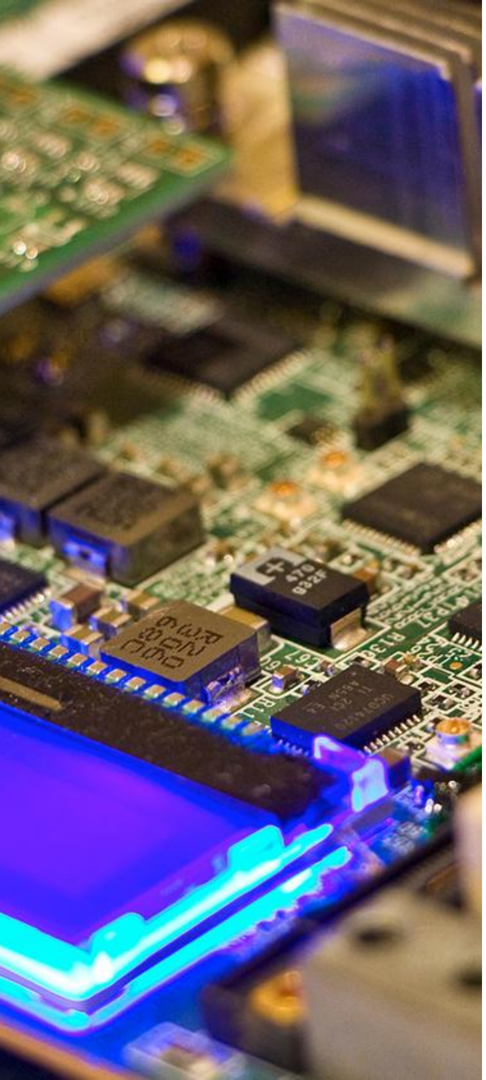
`libc/upstream-netbsd`

  functions from NetBSD libc

`libc/upstream-openbsd`

  functions from OpenBSD libc

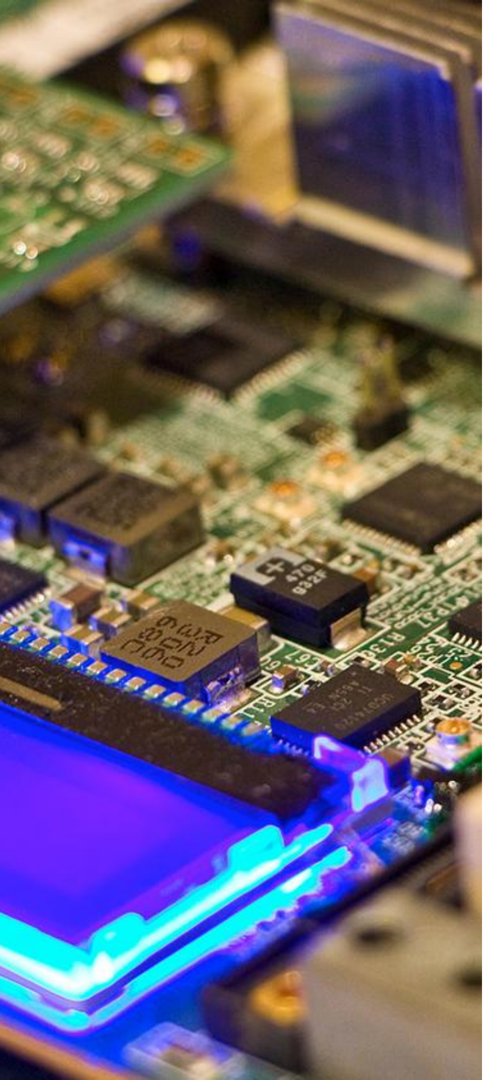
Linaro



Bionic’s structure

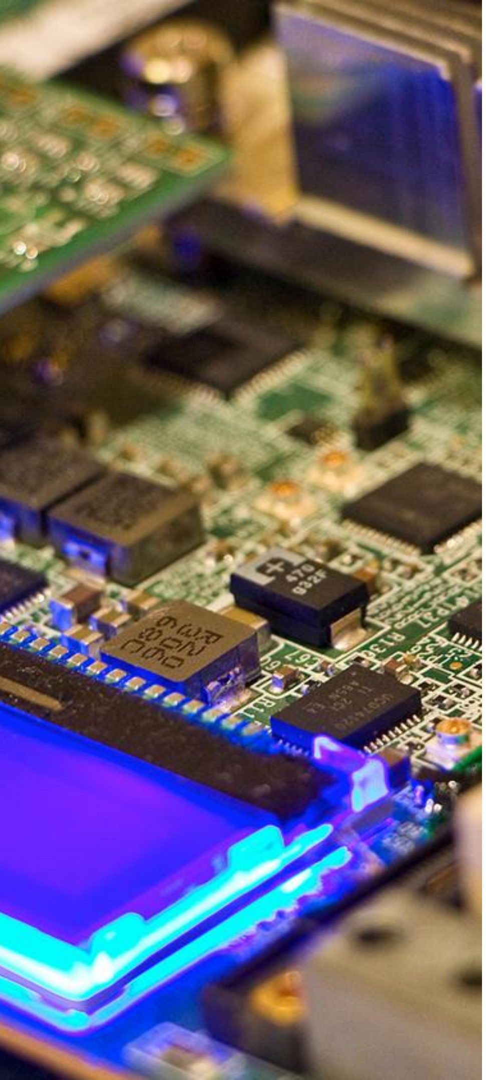
`libc/upstream-musl`

is a fairly obvious addition for merging musl’s implementations of libc functions



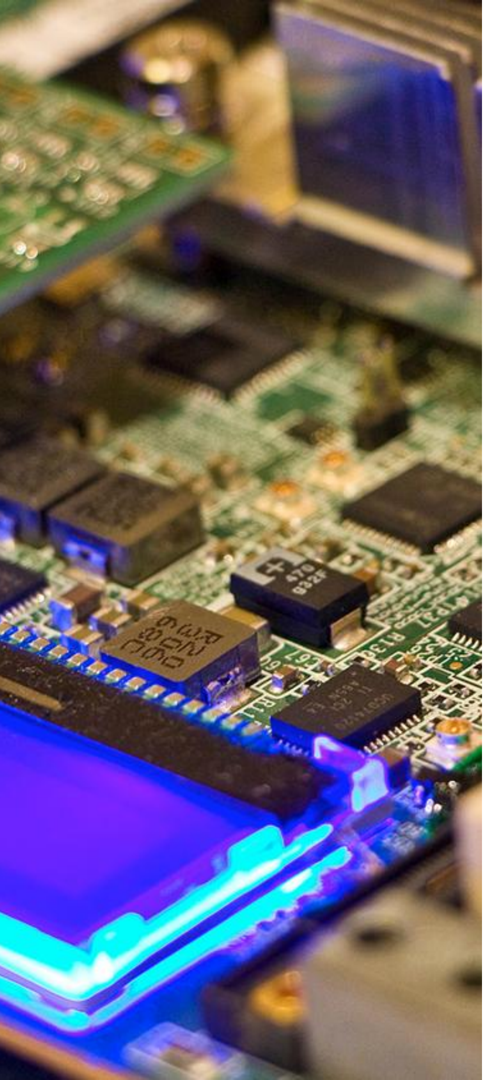
What needs to be done?

- Identify functions that are faster or smaller (or otherwise better) in musl
- Copy them to upstream-musl, edit Android.mk files
- build, run CTS, benchmark, upstream
- submit functions that are better in Bionic to musl -- let's not be leeches



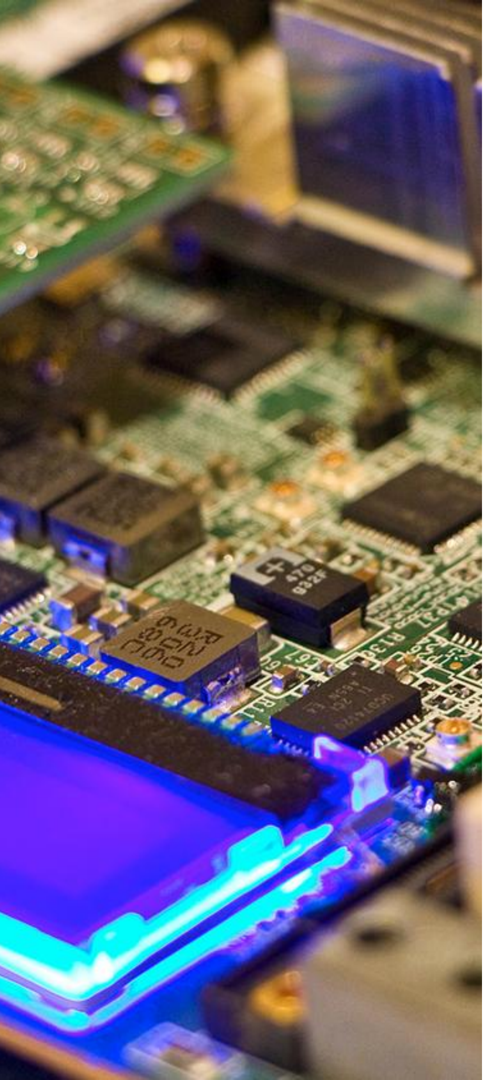
Handling tradeoffs

- Check if some simplifications done in musl (e.g. use ARM VFP `vsqrt.f32` and AArch64 `fsqrt` assembly instruction to implement sqrt()) change accuracy/break anything (compared to the much more complex and slower implementation in Bionic)



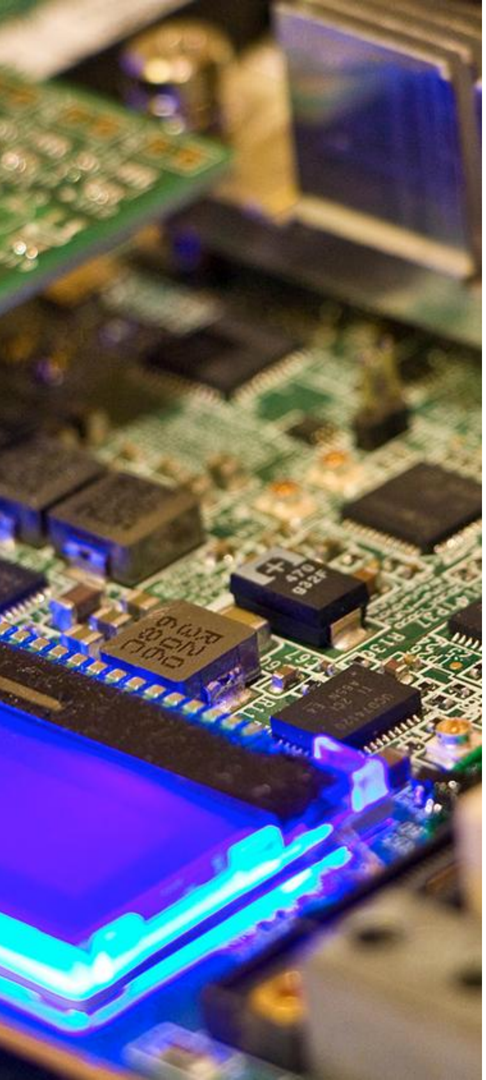
Handling tradeoffs

- The effect of such simplifications may be similar to `-ffast-math` - so it may be useful to make Bionic use the musl implementation if `-ffast-math` (or `-funsafe-math-optimizations`) is specified on the compiler command line and the traditional implementation if it's not



Current status

- Linaro has analyzed string handling functions (strcpy, memcpy, memset, …) on ARMv7 and ARMv8.
- So far, nothing for upstream-musl (but that’s not a big surprise, we’ve submitted optimized asm code for those to AOSP before)



Current status

- We expect improvements in musl over what's currently in Bionic in e.g. parts of libm
- musl's threading is interesting, but probably hard to fit into Bionic's pre-existing implementation - without breaking ABIs and assumed behaviors...

Questions? Suggestions?