March 2018 ELC 2018 Portland

Speeding up Linux Development with Debian and Open Embedded on DragonBoard™ 410c

Mark Charlebois

Director Engineering
Qualcomm Technologies, Inc.

Qualcomm® Snapdragon™ 410E Embedded Platform

Brief Overview

Snapdragon 410E Embedded Platform Overview

Snapdragon 410E

CPU: 1.2 GHz quad-core ARM v8 Cortex-A53, 32/64-bit capable

Connectivity: Integrated Wi-Fi, Bluetooth 4.xLE and GPS

DSP: Qualcomm® Hexagon™ DSP

Graphics: Qualcomm® Adreno™ 306 400MHz GPU

Interfaces: 1x USB2.0, 2x MIPI-CSI, MIPI-DSI, SD3.0 & eMMC v4.5 with DDR support

Qualcomm snapdragon

410E
(APQ8016E)
802.11b/g/n

1920x1080

Up to 13MP

DragonBoard 410c Development Board

ARROW

Linaro

96Boards

- Upstream kernel HW support for graphics, video, audio, IO
- Bootloader loads kernel from Android boot image

Snapdragon 410E Embedded Platform

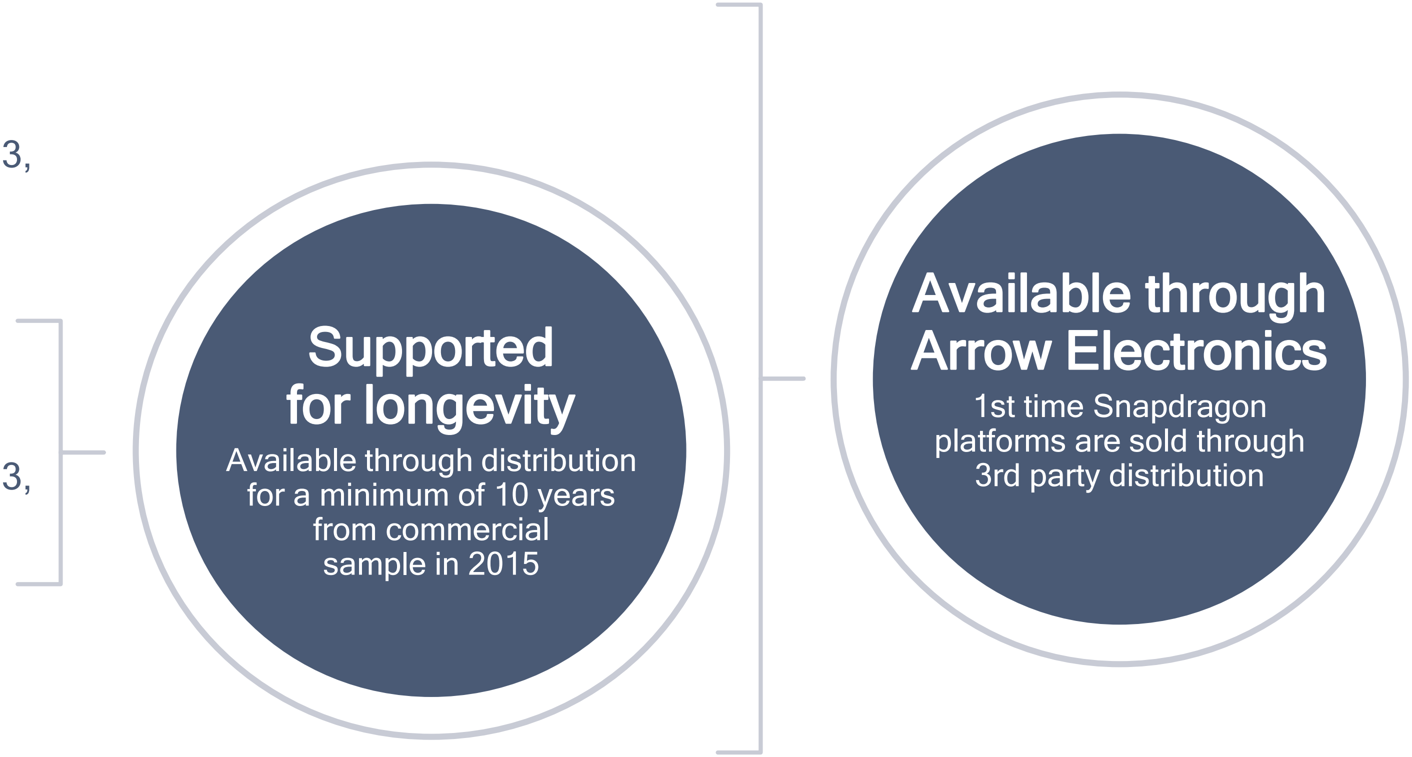
Path for prototyping today to many years of availability for production

DragonBoard 410c

1.2 GHz quad-core ARM v8 Cortex-A53,
32/64-bit capable

Snapdragon 410E

1.2 GHz quad-core ARM v8 Cortex-A53,
32/64-bit capable



Supported for longevity

Available through distribution for a minimum of 10 years from commercial sample in 2015

Available through Arrow Electronics

1st time Snapdragon platforms are sold through 3rd party distribution

https://developer.qualcomm.com/qfile/33927/snapdragon-e-selection-guide_0817_web.pdf

Debian Development

Pros and Cons of Debian

Huge repository (over 50K) of prebuilt packages for over a dozen architectures

- arm64 and armhf are well supported
  - See https://www.debian.org/mirror/size

Large, active community of supporters and contributors

- https://www.debian.org/intro/why_debian

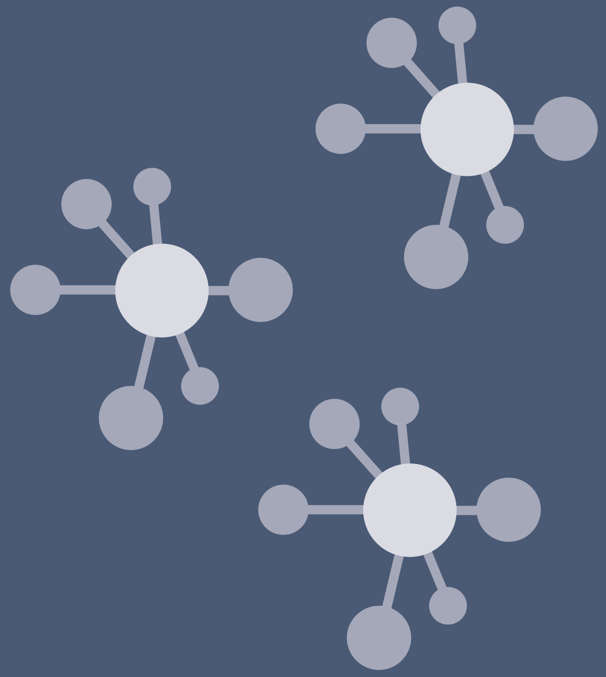
Debian Social Contract

- https://www.debian.org/social_contract
- *contrib* and *non-free* still supported by bug tracking system and mailing lists

Documentation is not particularly newbie friendly IMO

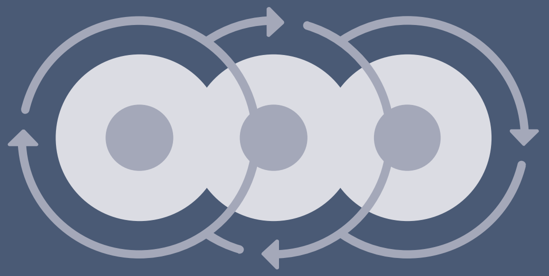
- Packaging and cross build docs assume prior knowledge of Debian specific tools

Vibrant Community



Pros and Cons of Debian

Versions and Support



https://wiki.debian.org/DebianReleases

At any given time, there is one stable release of Debian, which has the support of the Debian security team. When a new stable version is released, the security team will usually cover the previous version for a year or so, while they also cover the new/current version.

https://wiki.debian.org/LTS

Debian Long Term Support (LTS) is a project to extend the lifetime of all Debian stable releases to (at least) 5 years. Debian LTS will not be handled by the Debian security team, but by a separate group of volunteers and companies interested in making it a success.

Debian Development Methodology

- Native build
  - Many autoconf scripts and Makefiles were not designed for cross build
  - Easy to build small packages on target
  - ARM development machines are here now!
- "Native" Build in QEMU
  - Often used to build complex SW projects for Debian/Ubuntu
  - Arm64 sysroot using multistrap
    - Install additional packages into the sysroot
  - Use QEMU linux-user-mode to chroot into sysroot
    - At least 5x slower than native system
- Cross build
  - https://wiki.debian.org/Arm64Port#Building_packages

Debian SW Packaging

Creating src packages and packaging prebuilt binaries

- Deb pkg format
  - https://en.wikipedia.org/wiki/Deb_(file_format)
- Confusing evolution of packaging helpers:
  - See https://www.debian.org/doc/manuals/packaging-tutorial/packaging-tutorial.en.pdf
  - git-buildpackage (most modern)
  - pbuilder
  - sbuild
  - dh (debhelper 7)
- Generate binary package from src package
  - http://man7.org/linux/man-pages/man1/dpkg-buildpackage.1.html
- Packaging prebuilt files
  - Documented workflows expect to create a src deb then build it
  - I Use "dpkg-deb -build" for quick SW testing

Commercial Deployment of Debian

- Friendly terms for commercial deployment
  - https://www.debian.org/doc/manuals/debian-faq/ch-redistrib.en.html
  - Able to pin packages
- License compliance tools
  - https://wiki.debian.org/CopyrightReviewTools
- Commercial Support Options
  - https://www.debian.org/consultants/
- 5 years support of stable branches from LTS Project
- Linaro Debian is based on Testing
  - May align with productization timeframe

Yocto/ OpenEmbedded Development

Yocto vs OpenEmbedded

- Simplest Overview with description of layers
  - https://en.wikipedia.org/wiki/OpenEmbedded
- Good overview of the differences and terminology
  - https://elinux.org/images/d/de/Elc2011_kooi.pdf

TLDR;

- OpenEmbedded (OE) is a build system based on BitBake, it is not a distro
- OE is made up of collections of BitBake build recipes for packages organized in layers
- Yocto provides a reference distro (Poky) using OE (and additional tools and recipes)

Pros and Cons of Yocto/OE

- Best for minimal build, BOM cost sensitive products
- Flexibility and control
  - Choice of toolchain, C library, busybox, etc
- Yocto provides SW Compliance tools and some SPDX tooling
- HUGE learning curve and can be difficult to debug build issues
- Rebuilding a package may mean regenerating the rootfs image
- Large startup time cost, needs lots of storage and processing power

Flexible

and Complex

Yocto and OE Development Methodology

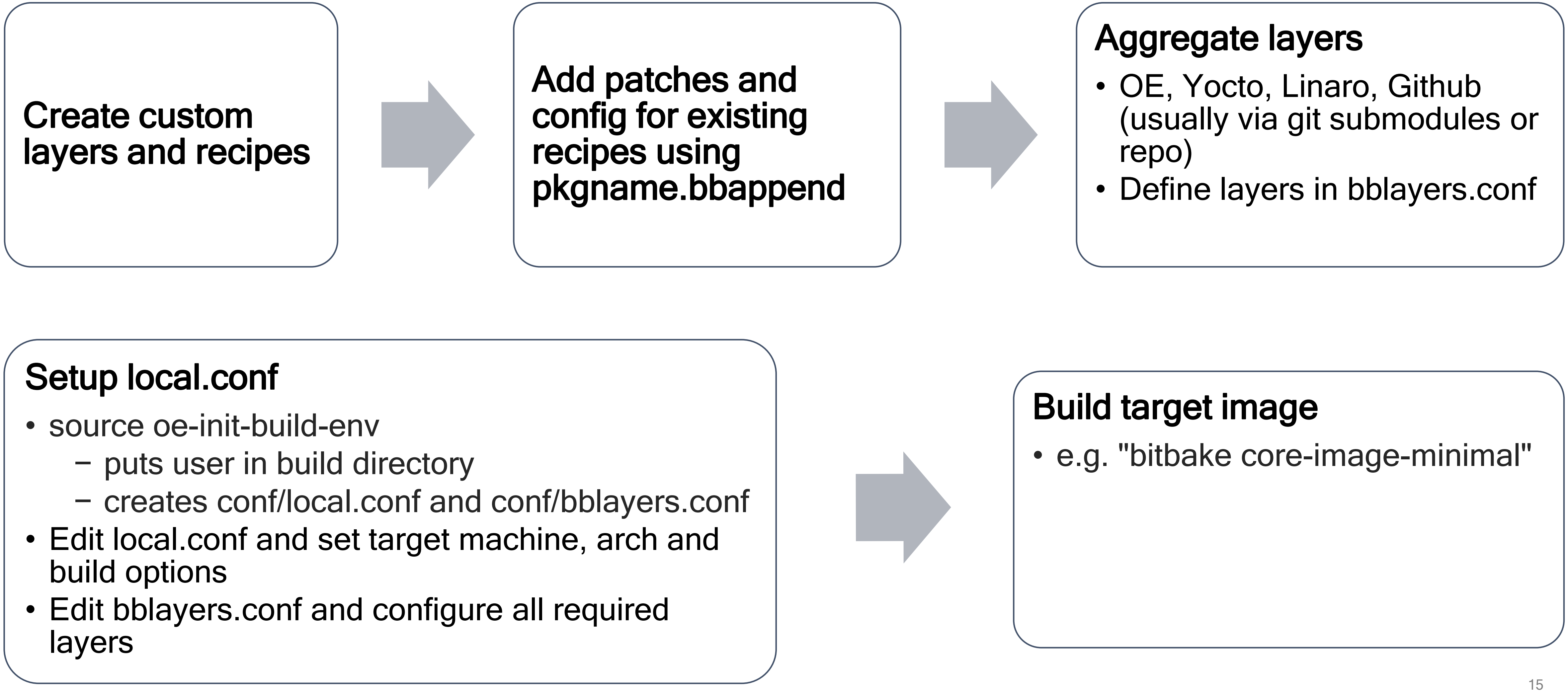
Wait, What? I'm a Distro Maintainer!?

- With OE you are building a Linux Distro
  - You control system updates, SW upgrades, critical fixes, user installable packages, (or not...)
  - No 3rd party SW ecosystem unless you create it
  - You manage the SDK for your distro

- Yocto SDK vs Extensible SDK
  - External developers require SDK to make compatible SW
  - Standard SDK
    - No packaging option using standard SDK, just sysroot
  - Extensible SDK
    - Richer (and more complicated) developer experience, and more work for distro maintainer
    - Great overview from previous ELC - https://www.youtube.com/watch?v=d3xanDJuXRA

Yocto and OE Development Methodology

Distro build process



Create custom layers and recipes

Add patches and config for existing recipes using pkgname.bbappend

Aggregate layers

- OE, Yocto, Linaro, Github (usually via git submodules or repo)
- Define layers in bblayers.conf

Setup local.conf

- source oe-init-build-env
  - puts user in build directory
  - creates conf/local.conf and conf/bblayers.conf
- Edit local.conf and set target machine, arch and build options
- Edit bblayers.conf and configure all required layers

Build target image

- e.g. "bitbake core-image-minimal"

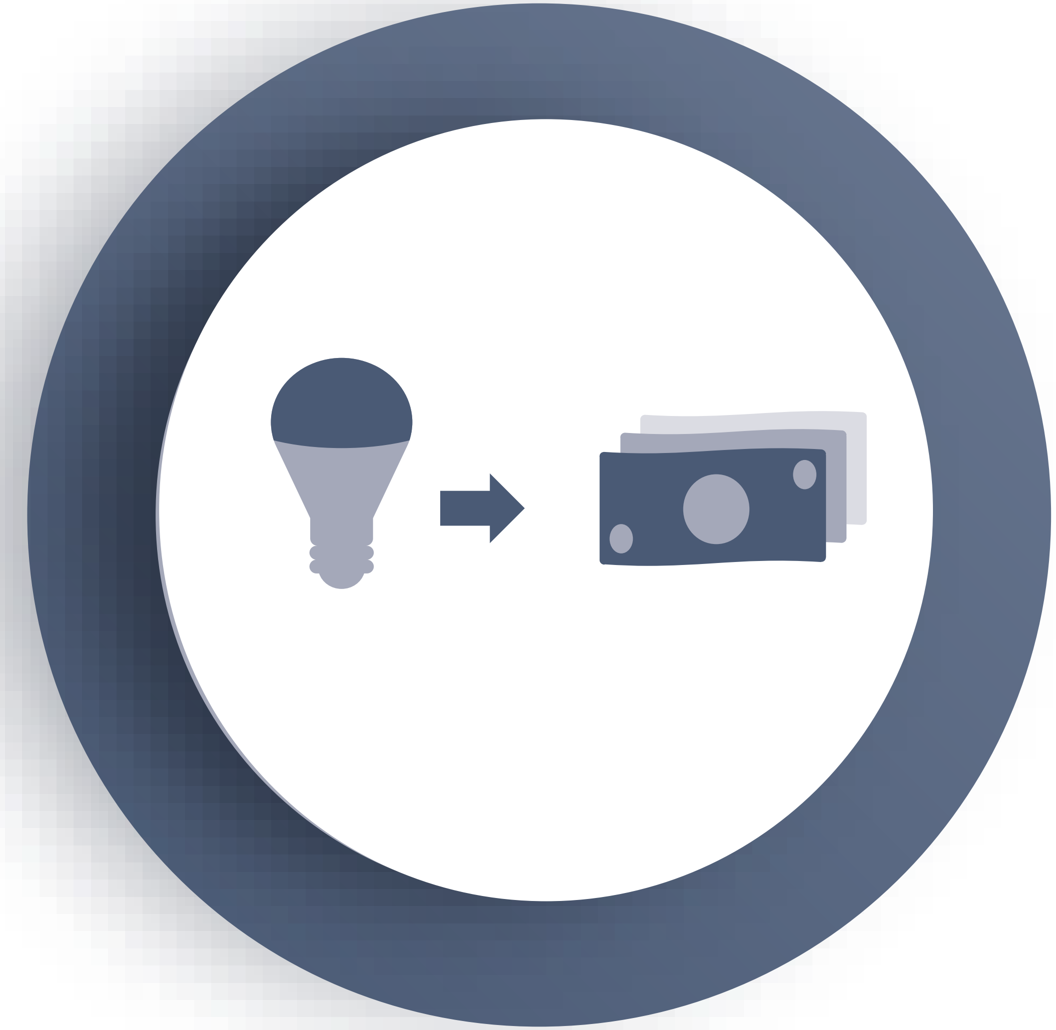
Alternative Yocto/OE Debian Hybrid Approaches

- https://elinux.org/images/6/67/Isar-Deby_jamboree61_r1.pdf
  - ISAR vs Deby
- https://www.cip-project.org/blog/2017/10/23/cip-launches-cip-core
  - The implementation of the CIP Core is based on the Deby distribution, a reproducible and maintainable embedded Linux distribution based on poky. Deby overlays meta-debian on top of poky to build file system images out of Debian long-term supported source code packages.

Commercial Deployment of OE/Yocto

- Yocto defines a reference Poky Distro
  - Not used for Linaro OE RPB
- Yocto makes 2 releases a year and each release is supported for one year
  - https://wiki.yoctoproject.org/wiki/Releases
  - Community may provide additional support for older releases
- Commercial support for Options for Yocto/OE from 3rd parties
  - e.g. Intrinsyc, TimeSys other OSVs
  - https://www.openembedded.org/wiki/CommercialSupport

Prototyping to Productization



Planning for Productization

Potential Gotcha's

- Moving from upstream to downstream kernel based BSP (or vice versa)
  - Proprietary middleware and HW acceleration
  - Not a problem on DragonBoard 410c (unless you were using Android)
- Conflicting prebuilt libs and system toolchain with different C++ ABI
  - Pre gcc 5 vs post (and pre 3.8 clang)
    - _GLIBCXX_USE_CXX11_ABI - non-zero value means use the new C++11-conforming ABI introduced in GCC 5, vs the previous ABI
- Commercial support and SW updates
  - LTS Kernels
  - Frozen vs rolling OS and kernel versions
- Open Source License Compliance
  - SPDX Report

Software ecosystem

Supporting great flexibility for architecting IoT solutions

OS*

- Linux
  - Debian
  - Yocto/ OpenEmbedded
  - Ubuntu Core
- Windows 10 IoT Core

Middleware

- AWS Greengrass
- IBM Watson IoT
- ROS
  (Robotics Operating System)

Cloud

- AT&T M2X
- AWS IoT
- IBM Bluemix
- Microsoft Azure IoT

*Android supported on DragonBoard 410c for hobbyist projects, not for commercial use in Snapdragon 410E and 600E designs

Useful Links

- Main SW page for DragonBoard 410c
  - https://www.96boards.org/product/dragonboard410c/
    - Links to Debian and OpenEmbedded pages
- Qualcomm® Developer Network projects
  - https://developer.qualcomm.com/project
- Debian Resources
  - https://www.debian.org/support
- OpenEmbedded
  - https://github.com/96boards/documentation/blob/master/consumer/guides/open_embedded.md
- Yocto
  - https://www.yoctoproject.org/docs/2.4.1/dev-manual/dev-manual.html
- Arrow Electronics: DragonBoard 410C and/or accessories
  - https://www.arrow.com/en/campaigns/the-dragonboard-is-here

Qualcomm

Thank you!

Follow us on: f 🐦 in

For more information, visit us at:

www.qualcomm.com & www.qualcomm.com/blog

Nothing in these materials is an offer to sell any of the components or devices referenced herein.

References in this presentation to “Qualcomm” may mean Qualcomm Incorporated, Qualcomm Technologies, Inc., and/or other subsidiaries or business units within the Qualcomm corporate structure, as applicable. Qualcomm Incorporated includes Qualcomm’s licensing business, QTL, and the vast majority of its patent portfolio. Qualcomm Technologies, Inc., a wholly-owned subsidiary of Qualcomm Incorporated, operates, along with its subsidiaries, substantially all of Qualcomm’s engineering, research and development functions, and substantially all of its product and services businesses, including its semiconductor business, QCT.