Why the Yocto Project for my IoT Project?

Drew Moseley
Technical Solutions Architect
Mender.io

Session overview

- Motivation
- Challenges for Embedded, Linux and IoT developers
- Describe IoT workflow
- Overview of Yocto
- Benefits of Linux and Yocto for IoT

about.me

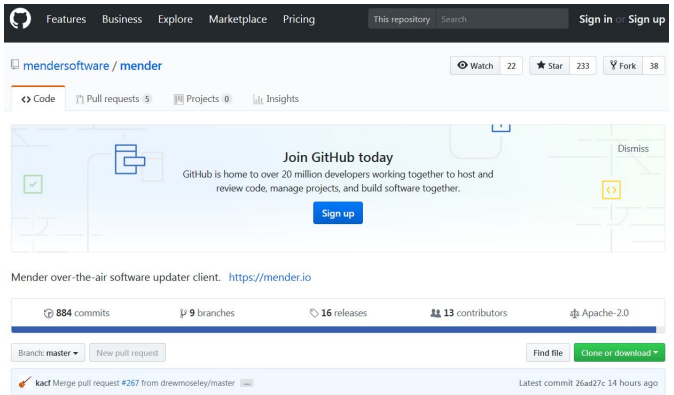
Drew Moseley

- 10 years in Embedded Linux/Yocto development.
- Longer than that in general Embedded Software.
- Project Lead and Solutions Architect.

drew.moseley@mender.io
https://twitter.com/drewmoseley
https://www.linkedin.com/in/drewmoseley/
https://twitter.com/mender_io

Mender.io

- Over-the-air updater for Embedded Linux
- Open source (Apache License, v2)
- Dual A/B rootfs layout (client)
- Remote deployment management (server)
- Under active development



Motivation

Embedded Projects increasingly use Linux:

- AspenCore/Linux.com[1]: Embedded Linux top 2 in current and planned use.

Huge IoT market opportunity:

- Forbes[2]: $267B by 2020

Linux is a big player in IoT

- Nodes & Gateways[3] - 17.18 Billion units by 2023
- Inexpensive prototyping hardware - Raspberry Pi, Beaglebone, etc
- Readily available production hardware - Toradex, Variscite, Boundary Devices
- Wide selection of chipsets - NXP, TI, Microchip, Nvidia

[1] https://www.linux.com/news/event/elce/2017/linux-and-open-source-move-embedded-says-survey
[2] https://www.forbes.com/sites/louiscolumbus/2017/01/29/internet-of-things-market-to-reach-267b-by-2020
[3] http://www.marketsandmarkets.com/PressReleases/iot-gateway.asp

Challenges for Embedded Linux/IoT Developers

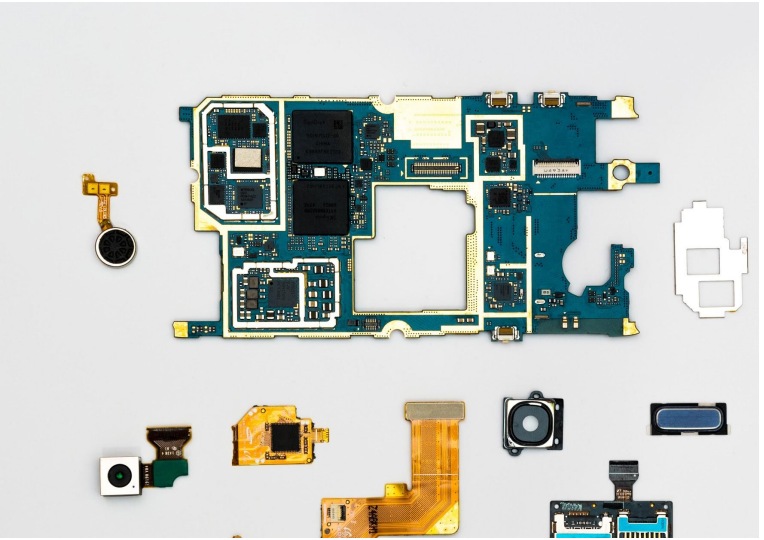
Hardware variety

Storage Media

Software may be maintained in forks

Cross development

Initial device provisioning



Getting Started Guide for Embedded/IoT Development

1. Buy Hardware[1]

[1] https://makezine.com/comparison/boards/

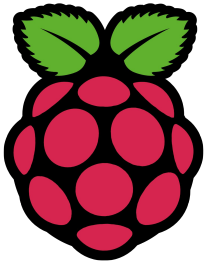
Getting Started Guide for Embedded/IoT Development

1. Buy Hardware[1]
2. Connect Hardware

[1] https://makezine.com/comparison/boards/

Getting Started Guide for Embedded/IoT Development

1. Buy Hardware[1]
2. Connect Hardware
3. Install OS
   a. Binary distribution
   b. Distribution build tools
   c. Build system



[1] https://makezine.com/comparison/boards/

Getting Started Guide for Embedded/IoT Development

1. Buy Hardware[1]
2. Connect Hardware
3. Install OS
   a. Binary distribution
   b. Distribution build tools
   c. Build system
4. Develop Application
5. Test Application

[1] https://makezine.com/comparison/boards/

Getting Started Guide for Embedded/IoT Development

1. Buy Hardware[1]
2. Connect Hardware
3. Install OS
   a. Binary distribution
   b. Distribution build tools
   c. Build system
4. Develop Application
5. Test Application
6. Test System

[1] https://makezine.com/comparison/boards/

Getting Started Guide for Embedded/IoT Development

1. Buy Hardware[1]
2. Connect Hardware
3. Install OS
   a. Binary distribution
   b. Distribution build tools
   c. Build system
4. Develop Application
5. Test Application
6. Test System
7. Deploy

[1] https://makezine.com/comparison/boards/

Getting Started Guide for Embedded/IoT Development

1. Buy Hardware[1]
2. Connect Hardware
3. Install OS
   a. Binary distribution
   b. Distribution h
   c. Bui
4. Develo
5. Test Ap
6. Test Sys
7. Deploy

[1] https://makezine.com/comparison/boards/

Build System Defined

_*Is*_

- Mechanism to specify and build
  - Define hardware/BSP components
  - Integrate user-space applications; including custom code
- Need reproducibility
- Must support multiple developers
- Allow for parallel processing
- (Cross) Toolchains
- License Management

*Is Not*

- An IDE
- A Distribution
- A deployment and provisioning tool
- An out-of-the-box solution

Yocto Project - Overview

"It's not an embedded Linux distribution -- it creates a custom one for you"[1]

- Recipes, metadata, dependencies and configuration
- Primary output: package feed
- Secondary output: boot images
- Builds all components from source
- Mechanism, not policy

Products:

- Root filesystem image
- Kernel, Bootloader, Toolchain
- Package Feed

[1]See more at https://www.yoctoproject.org

Yocto Project - Details

Organized into independent layers:

- Separation of functionality
- Allows different release schedules
- Expandability
  - Recipes developed in python and bash

SDK mechanism

- Separation of system and application devs
- Easily allows multiple developers to contribute

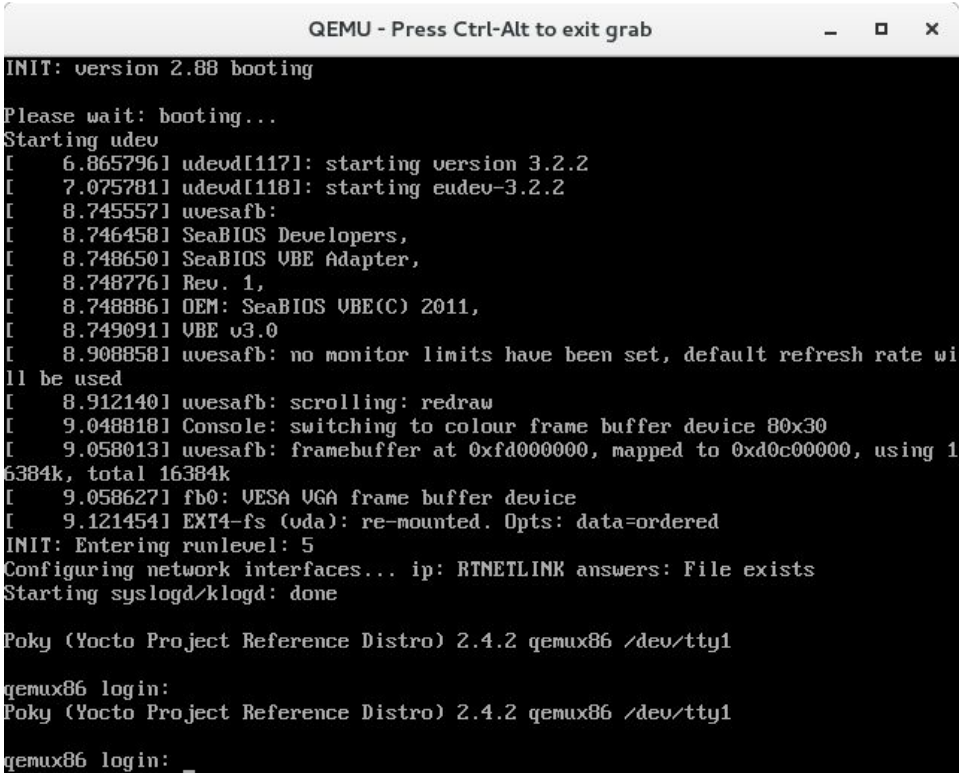
Optimizations:

- Faster build time reusing prebuilt binaries
- Parallel builds

Previous ELC talk estimated ~ 8400 software packages

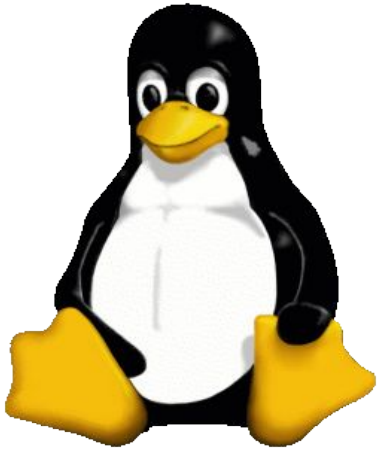
Yocto Project - Getting Started

```
$ git clone -b rocko \
    git://git.yoctoproject.org/poky.git
$ source poky/oe-init-build-env
$ MACHINE=qemux86 bitbake \
    core-image-minimal
$ runqemu qemux86
```



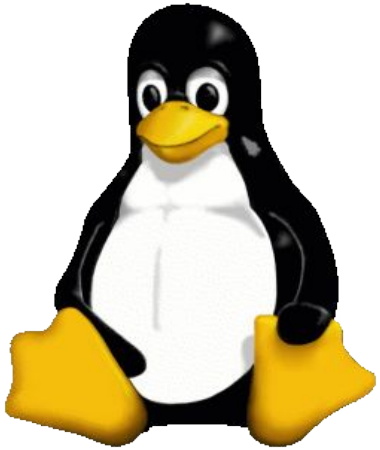
Why Linux for Embedded (1/2)?

- Ubiquity of:
  - Available Software
  - Expertise
  - Training Materials
- Broad device support
- Support for most common SOCs and reference boards
- Common system architecture on host and target
- Open Source



Why Linux for Embedded (2/2)?

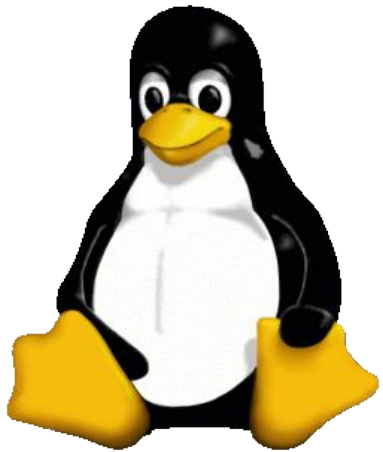
- Industry Support
  - Semiconductor manufacturers
  - Industry groups (Genivi, AGL, etc)
- High performance/low power
- Connectivity options
- Good cross-development support
  - Toolchains, including debug support
  - Serial consoles
  - JTAG



Why Yocto for IoT (1/2)?

- Strong support from major board and semiconductor vendors.
- Well defined workflow.
- Supports large developer teams .
- Easy access to IoT-specific protocols.
- Good connectivity and coexistence with RTOSes used in sensors and actuators.
- CVE updates in upstream stable branches.

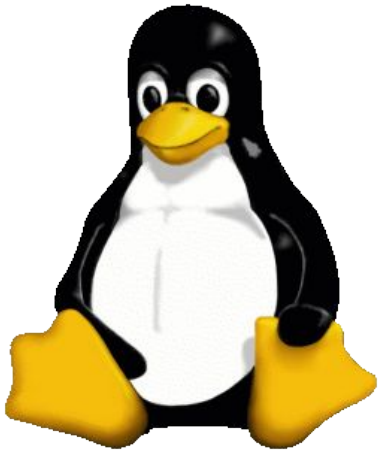
yocto PROJECT



Why Yocto for IoT (2/2)?

- IOT-Specific protocols well supported:
  - MQTT(python-paho-mqtt)
  - AMQP(rabbitmq)
  - CoAP(libcoap/python3-aiocoap)
  - ZeroMQ(zeromq)
  - GPS (gpsd)
  - Bluetooth(bluez5)
- Networking
  - systemd-networking
  - connman
  - NetworkManager
- Developer frameworks:
  - Python (meta-python layer)
  - NodeJS (meta-nodejs layer)

yocto PROJECT



Yocto IoT Layers

iot

Filter layers

Layer name	Description	Type	Repository
meta-vesta	Rigado BSP for the Vesta IoT Gateways	Machine (BSP)	https://git.rigado.com/vesta/meta-vesta.git
meta-refkit	IoT Reference OS Kit for Intel(r) Architecture distro layer	Distribution	https://github.com/intel/intel-iot-refkit
meta-flatpak	Flatpak support layer	Miscellaneous	https://github.com/intel/intel-iot-refkit
meta-refkit-computervision	IoT Reference OS Kit for Intel(r) Architecture profile for computer vision use cases	Miscellaneous	https://github.com/intel/intel-iot-refkit
meta-refkit-core	Core layer for the IoT Reference OS Kit for Intel(r) Architecture	Miscellaneous	https://github.com/intel/intel-iot-refkit
meta-refkit-extra	IoT Reference OS Kit for Intel(r) Architecture demo layer	Miscellaneous	https://github.com/intel/intel-iot-refkit
meta-refkit-industrial	IoT Reference OS Kit for Intel(r) Architecture profile for industrial use cases	Miscellaneous	https://github.com/intel/intel-iot-refkit
meta-intel-iot-middleware	Shared middleware recipes for Intel IoT platforms	Software	git://git.yoctoproject.org/meta-intel-iot-middleware
meta-iot-cloud	OpenEmbedded layer to add support for multiple cloud service provider solutions.	Software	https://github.com/intel-iot-devkit/meta-iot-cloud.git
meta-iot-web	IoT web components	Software	https://github.com/intel/iot-web-layers
meta-oic	Support for building the Open Interconnect Consortium Iotivity framework	Software	git://git.yoctoproject.org/meta-oic
meta-security-framework	adds higher-level security middleware and tools	Software	https://github.com/01org/meta-intel-iot-security.git
meta-security-smack	adds the Smack LSM to OpenEmbedded distros	Software	https://github.com/01org/meta-intel-iot-security.git

Source: https://layers.openembedded.org/layerindex/branch/master/layers/

Yocto Project for IoT; in summary

Pros:

- Widely supported by board and semiconductor vendors
- Active developer community
- Wide functionality and board support enabled by layer mechanism
- Customizable and expandable
- Minimal native tooling required
- Predictability of software contents

Cons:

- Steep learning curve
- Unfamiliar environment to non-embedded developers
- Resource-intensive
  - Long initial build times
  - Disk space

Q&A - Thank you!

Resources:

- https://bit.ly/2GIKlUQ - Previous ELC Talk comparing Embedded Linux build systems
- https://ubm.io/2Iazdfn - Deeper dive into the Yocto project
- https://hub.mender.io/t/raspberry-pi-3-model-b-b/57 - Building Yocto for Raspberry Pi with Mender.

@drewmoseley

drew.moseley@mender.io