Toradex

Swiss. Embedded. Computing.

Getting Started with Embedded Development using Toradex SoM

Maker Board – Right way?

- Often wrong way of doing things
- Is it reproducible?
- Use of distributions for Embedded?

Why Linux?

- Open source
- Huge number of contributors driving it forward
- Probability of drivers being available is very high
- Large number of user space software packages

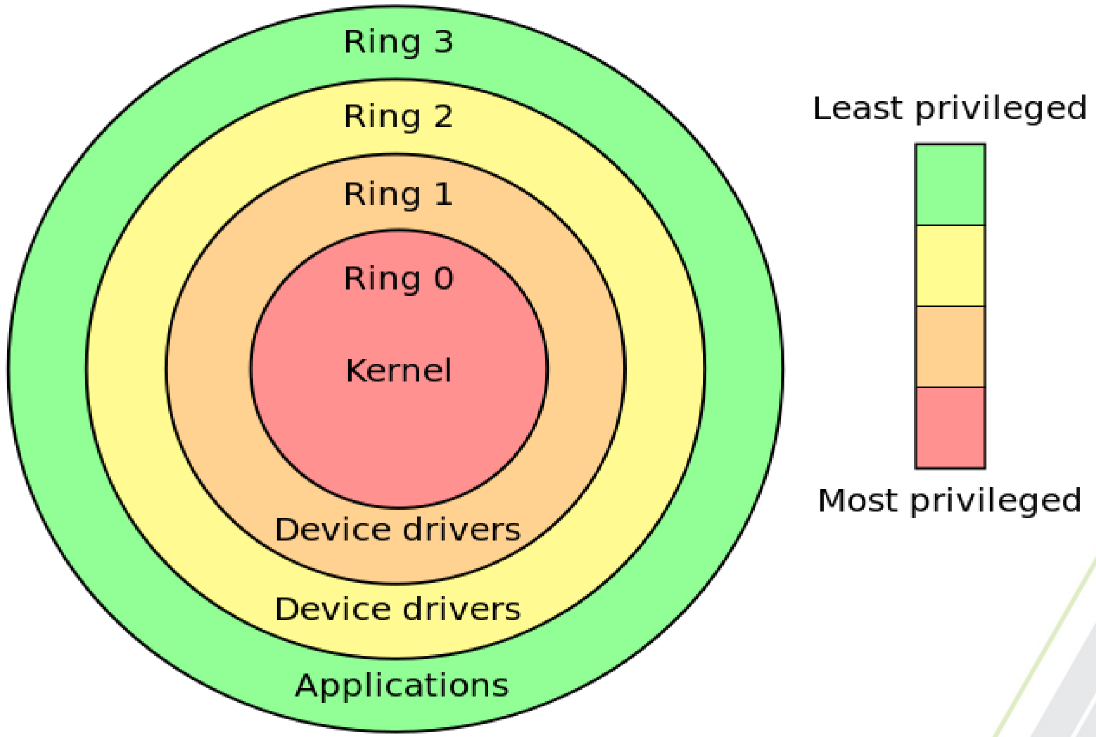
Toradex
Swiss. Embedded. Computing.

Where is Linux used?

- SpaceX
- International Space Station (ISS)
- Android
- Servers
- High Performance Computing Clusters
- High Performance Supercars
- Networking Equipment
- Embedded

The Linux approach to Embedded

- How is it different from microcontrollers?
  - It’s an operating system!
  - Monolithic kernel
  - Unix philosophy
  - Customisable to no end (subject to?)
  - Designed to be general purpose and maximise throughput



Is Linux right for you?

- Team skills? Background?
- Does lack of GUI tools pose a hindrance?
- Driver requirements?
- Latency requirements if any?
- Do you require a “non-embedded” language? Python? Nodejs? Java?
- Software component requirements?
- Understand the source of software components?

Before starting Linux?

- Install Linux (Not in VM!)
- Get well versed with command line
- What is cross compilation? Host? Target?
- Bootloader? Kernel? Rootfs?
- Deploying application to a target?

Embedded Build Systems

- Buildroot
  - Focuses on simplicity. Small and simple.
  - Special cases are handled in extension
  - Minimal by default making builds fast
  - Output is a root filesystem image and nothing more
- OpenEmbedded
  - Versatile and supports a wide range of embedded systems.
  - Defines builds in recipes and supports concept of layers (recipe collections)
  - Output is “a distribution”. Package feeds, package management, generation full disk images and SDK

Toradex
Swiss. Embedded. Computing.

Recommended Development Flow?

- OpenEmbedded SDK
  - Toolchain (compilers, debugger, assembler)
  - Header files
  - Libraries
- Eclipse setup
- Pinmultiplexing in kernel and u-boot if required
- Application development
- Custom image generation with OpenEmbedded

GPIO

- How does it differ from microcontrollers?
  - Method of access?
  - Interrupts handled?
  - Multiplexing?
- sysfs access (/sys/class/gpio)
- libsoc
- Drivers live in: drivers/gpio/

I2C

- i2cdev interface
  - open
  - read
  - write
  - close
- libsoc
- Drivers live in: drivers/i2c/busses

SPI

- spidev
- libsoc
- Drivers live in: drivers/spi

PWM

- /sys/class/pwm
- libsoc
- Drivers live in: drivers/pwm

Industrial IO subsystem

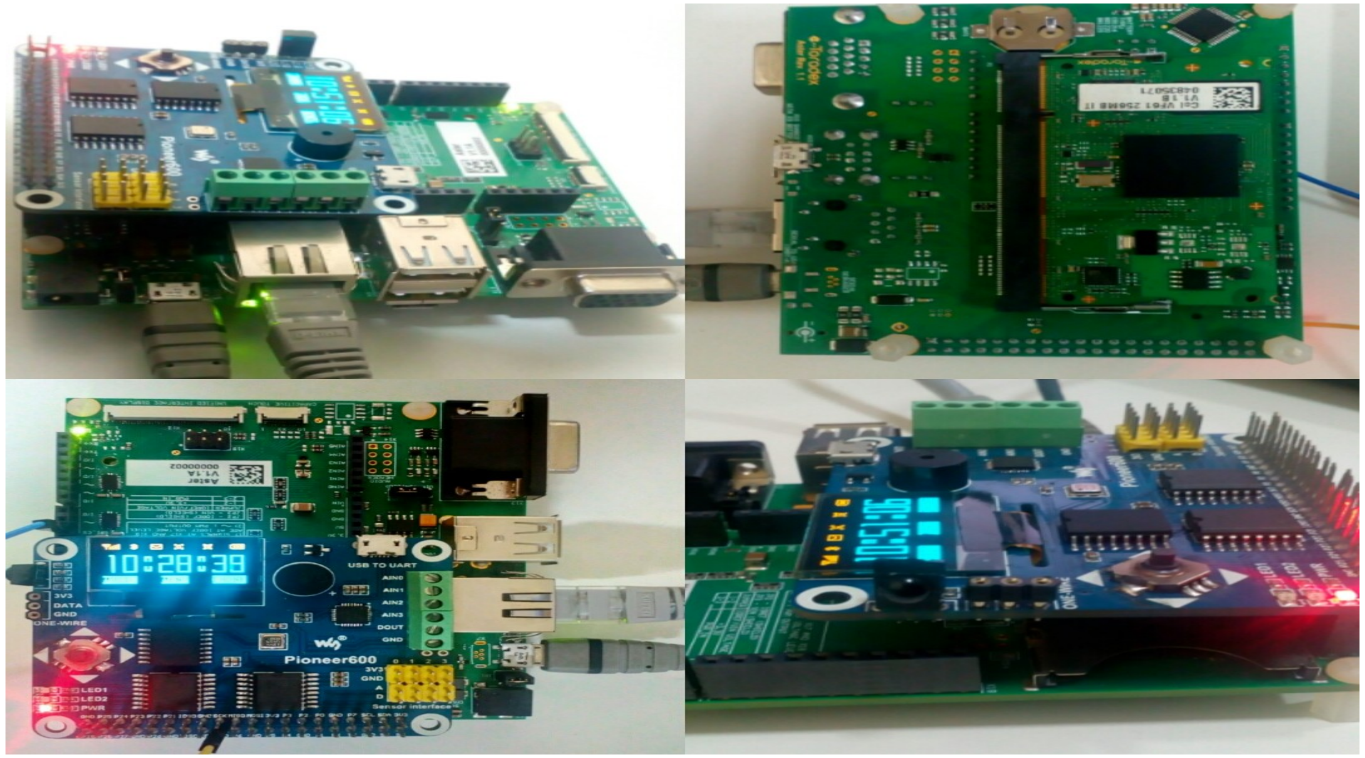
- Drivers live in: drivers/iio ; drivers/staging/iio
- /sys/bus/iio
  - ADC
  - DAC
  - Frquency
  - Gyro
  - Humidity

Linux Workshop Codes

Github repo: https://github.com/SanchayanMaity/LinuxWorkshop.git

Swiss. Embedded. Computing.

The Hardware



Toradex
Swiss. Embedded. Computing.

Thank you