Piece of cake

testing remote embedded devices
made easy with MuxPi

Paweł Wieczorek
February 3, 2018

Samsung R&D Institute Poland
Agenda

1. Introduction
2. Previous efforts
3. Idea
4. Hardware
5. Software
6. Next steps
7. Conclusion
Introduction
Use cases

Release engineering

- Continuous platform development
- QA step prior pulling new changes
- Package internal tests are not enough
• Easy to store in a secure manner
• Less effort than per developer
• Better utilized when shared
Piece of cake (with MuxPi)
Piece of cake (with Dryad)
Piece of cake (with Dryad)
Piece of cake (with Dryads)
Piece of cake (with Dryads)
Previous efforts
• Linaro Automated Validation Architecture
• Automation system for deploying operating systems
• Virtual and physical hardware supported
• Allows running boot, bootloader and system level tests
Kernel CI

https://kernelci.org/stats/
https://linux.codehelp.co.uk/the-problem-of-sd-mux.html
SD MUX – open hardware

https://git.tizen.org/cgit/tools/testlab/sd-mux
Autohat board (SD MUX-based)

https://github.com/resin-io/autohat-board
SD MUX issues

$ dmesg | tail -12
[ 98.375599] usb 3-1: new full-speed USB device number 12 using xhci_hcd
[ 98.487663] usb 3-1: device descriptor read/64, error -71
[ 98.703656] usb 3-1: device descriptor read/64, error -71
[ 98.919658] usb 3-1: new full-speed USB device number 13 using xhci_hcd
[ 98.919969] usb 3-1: Device not responding to setup address.
[ 99.123998] usb 3-1: Device not responding to setup address.
[ 99.327681] usb 3-1: device not accepting address 13, error -71
[ 99.439718] usb 3-1: new full-speed USB device number 14 using xhci_hcd
[ 99.440049] usb 3-1: Device not responding to setup address.
[ 99.644028] usb 3-1: Device not responding to setup address.
[ 99.847719] usb 3-1: device not accepting address 14, error -71
[ 99.847819] usb usb3-port1: unable to enumerate USB device
Idea
Constraints

- Only replaceable media
- No single point of failure parts
- No USB involvement (from test server)
Requirements

- **Minimum** external connections
- **Unified** remote access to target devices
- **Easy** setup and maintenance
Features

- User interface (often requested)
- Power measurement (increasing demand)
- Writing EDID to HDMI
Hardware
MuxPi components

NanoPi NEO
- 4-wire UART Level Shifter
- Watchdog Timer
- Power Control & Current Measurement
- μC Cortex-M0
  - CTRL, ADC, DyPers, HDMI, GPIO, UI

2-Port USB Hub
- SD-READER

SD-MUX
- USB ETH
- 2x USB-A

Connectors
- ETH IN
- ETH OUT
- 2x USB-A
- μSD Adapter
- HDMI
- ADD-ONS
- 4x DyPer

Power Supply
- 4x LED, 2x Button, OLED Display

UI
- 2x 2-channel DyPers
MuxPi components

- NanoPi NEO
  - USB / UART SWITCH & ID
  - 4-wire UART Level Shifter
  - Watchdog Timer
  - Power Control & Current Measurement
  - μC Cortex-M0
  - CTRL, ADC, DyPers, HDMI, GPIO, UI

- USB ETH
  - 2-Port USB HUB
  - SD-READER
  - SD-MUX
    - 2x 2-channel DyPers

- Connectors
  - ETH T5
  - USB OTG
  - 5Pin USB
  - UART
  - 2x PWR
  - BARREL

- Connectors
  - ETH OUT
  - 2x USB-A
  - μSD Adapter
  - HDMI
  - ADD-ONS
  - 4x DyPer

Power Supply
- 4x LED, 2x Button, OLED Display

20/49
MuxPi components

NanoPi NEO

USB / UART SWITCH & ID
4-wire UART Level Shifter
Watchdog Timer

Power Control & Current Measurement

μC Cortex-M0
CTRL, ADC, DyPers, HDMI, GPIO, UI

2x PWR
BARREL

Power Supply

UI
4x LED, 2x Button, OLED Display

SD-MUX
2-Port USB HUB
SD-READER

USB ETH

Connectors

ETH IN
USB OTG
5Pin USB
UART

Connectors

ETH OUT
2x USB-A
μSD Adapter
HDMI
ADD-ONS
4x DyPer

20/49
Essential MuxPi functions

- Switching a microSD card between DUT and TS
- Switching power supply for DUT
- Switching jumpers/buttons of DUT
- Measuring power consumption of DUT
- Writing EDID to DUT over HDMI connection
- Providing DUT connection (UART, USB, ETH, microSD card) over Ethernet
- Interacting with farm maintainer
Easy maintenance
Extensibility
Major improvements

- Independent (standalone)
- Aware of its state
- Easy to maintain
- Extensible from start
Building your own

NanoPi NEO ≈ $10
Building your own

NanoPi NEO \approx \$10
Parts \approx \$80

https://git.tizen.org/cgit/tools/muxpi
Building your own

NanoPi NEO ≈ $10
Parts ≈ $80
Soldering skills *High*
Building your own

<table>
<thead>
<tr>
<th>Item</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>NanoPi NEO</td>
<td>≈ $10</td>
</tr>
<tr>
<td>Parts</td>
<td>≈ $80</td>
</tr>
<tr>
<td>Soldering skills</td>
<td>High</td>
</tr>
<tr>
<td>Patience</td>
<td>A LOT</td>
</tr>
</tbody>
</table>
Building your own

NanoPi NEO  ≈ $10
Parts  ≈ $80
Soldering skills High
Patience A LOT

https://git.tizen.org/cgit/tools/muxpi
Software
Multitier architecture

• “Do One Thing and Do It Well”
• RESTful HTTP APIs
• Homogeneous solution stack
Responsibilities

• Who knows what requires verification?

• Who knows which actions are necessary?

• Who knows where can it be done?

• Who knows how to do it?
Responsibilities

• Who knows what requires verification? **Perun**

• Who knows which actions are necessary?

• Who knows where can it be done?

• Who knows how to do it?
Responsibilities

- Who knows what requires verification? **Perun**
- Who knows which actions are necessary? **Weles**
- Who knows where can it be done?
- Who knows how to do it?
Responsibilities

• Who knows what requires verification? **Perun**

• Who knows which actions are necessary? **Weles**

• Who knows where can it be done? **Boruta**

• Who knows how to do it?
Responsibilities

- Who knows what requires verification? **Perun**
- Who knows which actions are necessary? **Weles**
- Who knows where can it be done? **Boruta**
- Who knows how to do it? **MuxPi**
MuxPi (farm)

- Manages single DUT
- Fully aware of its capabilities
- Requires **only two** interfaces
  - Power supply
  - Network connection (Ethernet)
$ fota --help
Usage of fota:
  -card string
  path to SDcard
  -map string
  path to JSON formatted mapping
  -md5 string
  URL to MD5SUMS file
  -quiet
  suppress logging

$ stm --help
Usage of stm:
  -dut
  connect SD card to DUT
  -m duration
  time delay for tick command
  -tick
  power off and on after 'm' (s)
  -ts
  connect SD card to test server
$ fota --help
Usage of fota:
  -card string
    path to SD card
  -map string
    path to JSON formatted mapping
  -md5 string
    URL to MD5SUMS file
  -quiet
    suppress logging

$ stm --help
Usage of stm:
  -dut
    connect SD card to DUT
  -m duration
    time delay for tick command
  -tick
    power off and on after 'm' (s)
  -ts
    connect SD card to test server
$ fota --help
Usage of fota:
  -card string
    path to SDcard
  -map string
    path to JSON formatted mapping
  -md5 string
    URL to MD5SUMS file
  -quiet
    suppress logging

$ stm --help
Usage of stm:
  -dut
    connect SD card to DUT
  -m duration
    time delay for tick command
  -tick
    power off and on after 'm' (s)
  -ts
    connect SD card to test server
$ fota --help
Usage of fota:
  -card string
    path to SDcard
  -map string
    path to JSON formatted mapping
  -md5 string
    URL to MD5SUMS file
  -quiet
    suppress logging

$ stm --help
Usage of stm:
  -dut
    connect SD card to DUT
  -m duration
    time delay for tick command
  -tick
    power off and on after 'm' (s)
  -ts
    connect SD card to test server
• Dryad farm management system
• Schedules requests
  • Priority
  • Device groups
  • Delayed access
• Provides convenient access to selected Dryad
Boruta on stack
Dryad life cycle in Boruta

- **Prepares environment**
- **Sets up tunnel**
- **Actions performed**
- **Maintenance mode**
- **Unallocated**
- **Matches requirements**
• Lightweight testing framework
• Provides LAVA-like interface
• YAML job definition →
  actions executed on DUT
  • Deploy
  • Boot
  • Test
  • Collect
Weles on stack
Weles purpose

- Actions performed
- Unallocated
- Matches requirements
- Prepares environment
- Sets up tunnel
- Maintenance mode
Weles action sequence

- Parse YAML
- Collect assets
- Request DUT
- Perform tests
• OS images testing system
• Schedules verification (per new set of OS images)
• Automates QA step of Release Engineering Duty
Perun on stack
Perun action sequence

Crawl URL

Report changes

Submit Weles jobs

Collect artifacts

Interpret results
Keeping it simple
Keeping it simple (and decoupled)
Keeping it simple (and decoupled)
Keeping it simple (and decoupled)
Keeping it simple (and decoupled)
Next steps
Future plans

**Hardware**
- Audio I/O
- USB Type C investigation
- NanoPi serial console on USB

**Software**
- Web interfaces for current layers
- Service state management
- Release engineer's layer
Further details

- MuxPi
  https://wiki.tizen.org/MuxPi

- SD MUX (deprecated – lesson learnt)
  https://wiki.tizen.org/SD_MUX
• Mailing list
general@lists.tizen.org
• #tizen on Freenode
https://webchat.freenode.net/?channels=tizen
Conclusion
• Quick setup
• Easy maintenance
• Responsibilities division
• Execution parallelization
• Environment unification
Questions?
Thank you!

Paweł Wieczorek
p.wieczorek2@samsung.com
Samsung R&D Institute Poland
• **Metropolis** – simple, modern Beamer theme
Pictures used

- https://commons.wikimedia.org/wiki/File:Tux.svg
- https://commons.wikimedia.org/wiki/File:Wayland_Logo.svg
- https://commons.wikimedia.org/wiki/File:Enlightenment_logo_black.png
- https://developer.tizen.org/sites/default/files/images/about_tizen_1.png
- https://commons.wikimedia.org/wiki/File:ColoredBlankMap-World-10E.svg
- https://commons.wikimedia.org/wiki/File:Nuvola_Korean_flag.svg
- https://commons.wikimedia.org/wiki/File:Nuvola_Polish_flag.svg
- https://commons.wikimedia.org/wiki/File:Nuvola_USA_flag.svg
- https://validation.linaro.org/static/docs/v2/_images/lava.svg
- https://wiki.linaro.org/Platform/LAB/LMP_in_practice
- https://forums.resin.io/uploads/resin/original/1X/88ab2e061cd644b18b95fa99ede9ce6b98adfa44.jpg
- https://commons.wikimedia.org/wiki/File:Italian_traffic_signs_-_fermarsi_e_dare_precendenza_-_stop.svg
- https://farm9.staticflickr.com/8263/28955874330_d1b1202ae8_k_d.jpg
- https://commons.wikimedia.org/wiki/File:PEO-smiley_smile.svg