Porting Graphical Stacks to RISC-V using QEMU and Yocto

Alistair Francis <Alistair.Francis@wdc.com>

Presented by: Atish Patra <Atish.Patra@wdc.com>

18th of July 2018
RISC-V Workshop Chennai
GUIs are important

- For RISC-V to compete with other incumbent architectures it will need support to run a Graphical User Interface (GUI).
- Users are accustomed to GUIs everywhere. From cars to smart fridges everything has a GUI
- The ones presented by Android™ and iOS™ are the most ubiquitous

Android is a trademark of Google LLC.
IOS is a trademark or registered trademark of Cisco in the U.S. and other countries.
Privacy and Open Source are also important

- With the rise of open source software and people becoming increasingly privacy conscious there is a growing number of open source smartphone software stacks that compete against Android and iOS
- Users are interested in more open software stacks that take their privacy into consideration
Don’t we already have GUIs?

- What we have are some standard Desktop distribution (Debian, Fedora) environments
- This presentation is instead focused on
  - Trying to run a mobile distribution (Plasma Mobile)
  - Building the distribution ourselves (with Yocto)
  - Testing on QEMU with virtual GPUs and virtual displays
The software and hardware layers

- RISC-V Board/QEMU
- Linux
- Matchbox
- XFCE
- KDE Runtime
- Plasma Mobile
Plasma Mobile and KDE

• Plasma Mobile is the mobile version of the KDE Plasma Environment

• It has an active development community behind it backed by the KDE desktop environment project developers

• It is running on real ARM hardware (Nexus 5) today

• Offers a complete software system for mobile devices with a strong focus on user’s privacy protection

• Builds on top of Qt5
QEMU

- QEMU is a very quick open source (mostly GPLv2) emulator and hypervisor
- It is not cycle accurate, but it is functionally accurate
- It uses Tiny Code Generator (TCG) to translate different architecture instructions (guest) to the host PC (host)
- QEMU supports RISC-V machines!
- QEMU also supports PCIe, virtual GPUs, displays and input devices
Yocto

- Yocto is a Linux Foundation backed project whose goal is to allow the creation of Linux distributions specifically for embedded devices
- Yocto allows developers to create their own distribution or their own SDK
- It builds all the software from the cross compiler to the Linux kernel itself
- Allows us to add/remove required packages as desired and make any source changes as required
- Functionality is added via meta layers
Yocto layers

- meta-riscv
- meta-qt5
- meta-oe
- OE-Core
- Bitbake
- meta-kf5
- meta-kde
QEMU Model

- CPU
- PLIC
- Other standard RISC-V devices
- Xilinx PCIe Root Complex
- PCIe STD VGA
- PCIe USB Host
Building a full GUI stack in Yocto

- Working on top of other Yocto meta layers
  - meta-riscv: Allows cross compiling for RISC-V and includes specific RISC-V fixes
  - meta-qt5: Adds all the Qt5 packages, required for Plasma
  - meta-kf5: Base KDE framework packages required for KDE
  - meta-kde: KDE layer, includes plasma mobile

- Unluckily, this doesn’t just work there are still cross compile errors
  - Changes have been made to meta, meta-qt5, meta-kf5 and meta-kde and corresponding packages to allow cross-compile
    - Mostly small changes to specify that RISC-V is a supported ISA
    - Also changes to some of the way everything is packaged
  - Upstreaming effort is ongoing
All this work, for the Matchbox Terminal
Next step, run XFCE
root@freedom-u540:~# DISPLAY=:0 startkde
/usr/bin/startkde: line 106: kapplymousetheme: command not found
QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-root'
libEGL warning: DRI2: failed to authenticate
QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-root'
[ 231.098066] kspashqml[301]: unhandled signal 11 code 0x1 at 0x0000000000000000 in
libQtCore.so.5.11.1[2000924000+50a00000]

[ 231.098763] CPU: 0 PID: 361 Comm: ksplashqml Tainted: G W 4.17.0 #1
[ 231.099198] sepc: 0000000005615c ra: 0000000001671e sp: 0000003ff6930
[ 231.100048]  s0: 0000002000b5615c s1: 0000000000019670 a0: 0000000000019d20
[ 231.102060]  s2: 0000003ff6a30 s3: 0000003ff6a20 s4: 0000002000be47a0
[ 231.102455]  s5: 0000000000016f70 t3: 000000000023215c t4: 0000000000000002
[ 231.102836]  s6: 0000003ff6ad808 s7: 000000000001a016 a1: 0000002000be47a0

startkde: Starting up...
QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-root'
kdeinit5: preparing to launch '/usr/libexec/kf5/klauncher'
kdeinit5: Launched KLauncher, pid = 381, result = 0
QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-root'
libEGL warning: DRI2: failed to authenticate
QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-root'
What doesn’t work?

• Input. We don’t have any input to the graphics stack via QEMU
  – This appears to be a configuration issue between QEMU and Linux
    • Linux can see the USB devices

• KDE. KDE starts up and we see the mouse theme change, but we never get the full graphics up
What next?

- Upstream! This work is being upstreamed so others can use it and build on it
- Keyboard and Mouse support. Need to setup input support in QEMU
- Full KDE stack. More debugging on why the display doesn’t start is required
  - Trying to debug a seg fault in a graphics library is really hard
- Port and test on real hardware
Can I try it at home?

- Not really.
- At the moment focusing on upstreaming the changes to the meta layers and fixing hacks
- If anyone is interested in having a go reach out to me and I can send you information: alistair.francis@wdc.com