Die wirre Welt der kleinen Computer

Daniel Maslowski
Agenda

- Introduction
- Understanding hardware
- Looking at the SoC
- Tracking upstream
- Cool projects
- Finding a community
Introduction
So many options…

There are hundreds of "single board computers" (SBCs). There are compute modules (SoMs) and carrier boards. There are add-ons, lots of form factors. There are microcontrollers and application processors. There are different instruction sets: RISC-V, ARM 32-bit + 64-bit, MIPS…

People often look at computers and ask: 

"Does/can it run "Linux"?"

That is a very tough question. Let's see! :‐( 
So many options…

There are hundreds of “single board computers” (SBCs).
So many options…

There are hundreds of “single board computers” (SBCs).
There are compute modules (SoMs) and carrier boards.
So many options…

- There are hundreds of “single board computers” (SBCs).
- There are compute modules (SoMs) and carrier boards.
- There are add-ons, lots of form factors.
So many options…

There are hundreds of “single board computers” (SBCs).
There are compute modules (SoMs) and carrier boards.
There are add-ons, lots of form factors.
There are microcontrollers and application processors.
So many options…

There are hundreds of “single board computers” (SBCs).
There are compute modules (SoMs) and carrier boards.
There are add-ons, lots of form factors.
There are microcontrollers and application processors.
There are different instruction sets: RISC-V, Arm 32bit + 64-bit, MIPS…
So many options…

There are hundreds of “single board computers” (SBCs).
There are compute modules (SoMs) and carrier boards.
There are add-ons, lots of form factors.
There are microcontrollers and application processors.
There are different instruction sets: RISC-V, Arm 32bit + 64-bit, MIPS…
People often look at computers and ask:

*Does/can it run “Linux”?*
So many options…

There are hundreds of “single board computers” (SBCs).
There are compute modules (SoMs) and carrier boards.
There are add-ons, lots of form factors.
There are microcontrollers and application processors.
There are different instruction sets: RISC-V, Arm 32bit + 64-bit, MIPS…

People often look at computers and ask:
  *Does/can it run “Linux”?*

That is a very tough question. Let’s see! :-)
Compatibility
Official ROCK Pi system images can also be downloaded from [ROCK Pi BaiduPan](https://baidu.com) or [Radxa Github Release](https://github.com).

For user names and passwords please check the [FAQ](https://faq.com).

<table>
<thead>
<tr>
<th>Description</th>
<th>Linux</th>
<th>MacOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Etcher - A user friendly Image Writer</td>
<td>Linux 64bit</td>
<td><a href="https://www.linux.com">Linux 32bit</a></td>
</tr>
</tbody>
</table>

### Official Images

- **Android 9**
  - [Android Install Wiki](https://android.com)

- **Ubuntu Focal**
  - [Ubuntu Product. Install Wiki](https://ubuntu.com)
  - Ubuntu Focal (20.04).

- **Debian Buster**
  - [Debian Product. Install Wiki](https://deb.com)
  - Debian Buster(10).

### Third Party Images

- **openSUSE Tumbleweed JeOS**
  - [Build 2022-07-04](https://opensuse.com)
  - [Install Wiki](https://www.opensuse.com)
  - user:root password:linux
Peripherals

Drivers and special component firmware can be very nasty.

▶ graphics/GPU
▶ audio
▶ wireless/Wi-Fi + Bluetooth

Fun story: We had to install bluez-firmware to get Wi-Fi working on a Radxa Zero.
Drivers and special component firmware can be very nasty.

- graphics / GPU
- audio
- wireless / Wi-Fi + Bluetooth

Fun story:
We had to install bluez-firmware to get Wi-Fi working on a Radxa Zero.
Peripherals

Drivers and special component firmware can be very nasty.

- graphics / GPU
- audio
- wireless / Wi-Fi + Bluetooth

Fun story:

We had to install bluez-firmware to get Wi-Fi working on a Radxa Zero.
Prediction

The Future of Consumer SBCs: Has the Pi bubble burst?

Five Future SBC Predictions

- There will be an increasing diversity of low-cost, consumer, small form-factor computers.
- SBCs in general will be more industrially focused.
- There will be a strong market for consumer SBCs costing up to about $75 (c. £65 / €70).
- SBCs costing $100+ will increasingly struggle in the consumer market, unless they offer key maker features, such as GPUs/NPUs for machine learning.
- A greater use of microcontrollers in the maker space.

https://www.youtube.com/watch?v=Hjb3bx6vxnc
Understanding hardware
Why does my LCD not work?

Look very, very closely at the interface…

Aha, Pin 31 enablesthe display.

But it's not connected on my board.

Let's fix it! :‑)
Why does my LCD not work?

Look very, very closely at the interface…

Aha, Pin 31 enables the display.

But it's not connected on my board.

Let's fix it! :‑)
Why does my LCD not work?

Look very, very closely at the interface…

<table>
<thead>
<tr>
<th>No.</th>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>VBL-</td>
<td>Backlight LED Cathode</td>
</tr>
<tr>
<td>2</td>
<td>VBL+</td>
<td>Backlight LED Anode.</td>
</tr>
<tr>
<td>3</td>
<td>GND</td>
<td>System Ground</td>
</tr>
<tr>
<td>4</td>
<td>VCC</td>
<td>Power supply for logic operation</td>
</tr>
<tr>
<td>5~12</td>
<td>R0~R7</td>
<td>Data bus</td>
</tr>
<tr>
<td>13~20</td>
<td>G0~G7</td>
<td>Data bus</td>
</tr>
<tr>
<td>21~28</td>
<td>B0~B7</td>
<td>Data bus</td>
</tr>
<tr>
<td>29</td>
<td>GND</td>
<td>System Ground</td>
</tr>
<tr>
<td>30</td>
<td>CLK</td>
<td>Pixel clock signal</td>
</tr>
<tr>
<td>31</td>
<td>DISP</td>
<td>Display on/off control</td>
</tr>
<tr>
<td>32</td>
<td>HSYNC</td>
<td>Horizontal Sync signal</td>
</tr>
<tr>
<td>33</td>
<td>VSYNC</td>
<td>Vertical Sync signal</td>
</tr>
<tr>
<td>34</td>
<td>DEN</td>
<td>Data Enable</td>
</tr>
<tr>
<td>35</td>
<td>NC</td>
<td>No connect</td>
</tr>
<tr>
<td>36</td>
<td>GND</td>
<td>System Ground</td>
</tr>
<tr>
<td>37</td>
<td>NC/XR</td>
<td>TP pin XR</td>
</tr>
<tr>
<td>38</td>
<td>NC/YD</td>
<td>TP pin YD</td>
</tr>
<tr>
<td>39</td>
<td>NC/XL</td>
<td>TP pin XL</td>
</tr>
<tr>
<td>40</td>
<td>NC/YU</td>
<td>TP pin YU</td>
</tr>
</tbody>
</table>
Why does my LCD not work?

Look very, very closely at the interface…

<table>
<thead>
<tr>
<th>No.</th>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>VBL-</td>
<td>Backlight LED Cathode</td>
</tr>
<tr>
<td>2</td>
<td>VBL+</td>
<td>Backlight LED Anode.</td>
</tr>
<tr>
<td>3</td>
<td>GND</td>
<td>System Ground</td>
</tr>
<tr>
<td>4</td>
<td>VCC</td>
<td>Power supply for logic operation</td>
</tr>
<tr>
<td>5~12</td>
<td>R0~R7</td>
<td>Data bus</td>
</tr>
<tr>
<td>13~20</td>
<td>G0~G7</td>
<td>Data bus</td>
</tr>
<tr>
<td>21~28</td>
<td>B0~B7</td>
<td>Data bus</td>
</tr>
<tr>
<td>29</td>
<td>GND</td>
<td>System Ground</td>
</tr>
<tr>
<td>30</td>
<td>CLK</td>
<td>Pixel clock signal</td>
</tr>
<tr>
<td>31</td>
<td>DISP</td>
<td>Display on/off control</td>
</tr>
<tr>
<td>32</td>
<td>HSYNC</td>
<td>Horizontal Sync signal</td>
</tr>
<tr>
<td>33</td>
<td>VSYNC</td>
<td>Vertical Sync signal</td>
</tr>
<tr>
<td>34</td>
<td>DEN</td>
<td>Data Enable</td>
</tr>
<tr>
<td>35</td>
<td>NC</td>
<td>No connect</td>
</tr>
<tr>
<td>36</td>
<td>GND</td>
<td>System Ground</td>
</tr>
<tr>
<td>37</td>
<td>NC/XR</td>
<td>TP pin XR</td>
</tr>
<tr>
<td>38</td>
<td>NC/YD</td>
<td>TP pin YD</td>
</tr>
<tr>
<td>39</td>
<td>NC/XL</td>
<td>TP pin XL</td>
</tr>
<tr>
<td>40</td>
<td>NC/YU</td>
<td>TP pin YU</td>
</tr>
</tbody>
</table>

Aha, Pin 31 enables the display. But it’s not connected on my board. Let’s fix it! :-(
Hardware Hacks

https://github.com/adamgreig/d1rgb
To understand a chip, you need its *Technical Reference Manual (TRM)*. It may also be called *SoC (System on Chip) manual* or just *datasheet*.
To understand a chip, you need its Technical Reference Manual (TRM).
It may also be called SoC (System on Chip) manual or just datasheet.
Note that some datasheets may be only brief summaries.
To understand a chip, you need its *Technical Reference Manual (TRM)*.

It may also be called *SoC (System on Chip) manual* or just *datasheet*.

Note that some datasheets may be only brief summaries.

Manuals are rarely public, often contain “confidentiality” notes.

They may have errata and/or be incomplete.
So you want to build an embedded Linux system?

The first step is to architect your system. This is hard to do unless what you’re building is trivial or you have a lot of experience, so you’ll probably start by buying some reference hardware, trying it out to see if it can do what you’re trying to do (both in terms of hardware and software), and then using that as a jumping-off point for your own designs.

https://jaycarlson.net/embedded-linux/
Looking at the SoC
Capabilities

Not every SoC is general purpose.
Many SoCs are designed for narrow tasks, yet seem generic.
Common FruitPi SoC Vendors

Note: These compete on the multimedia device market, e.g., TV boxes.

https://linux-sunxi.org/Comparison_of_chip_maker_openness
Common FruitPi SoC Vendors

OrangePi, BananaPi, CherryPi, MangoPi, … you get the idea.

Note: These compete on the multimedia device market, e.g., TV boxes.

https://linux-sunxi.org/Comparison_of_chip_maker_openness
Common FruitPi SoC Vendors

OrangePi, BananaPi, CherryPi, MangoPi, … you get the idea.
Common FruitPi SoC Vendors

OrangePi, BananaPi, CherryPi, MangoPi, … you get the idea.

Loader Tools
sunxi-fel/xfel  rkflashtool  pyamlboot
Common FruitPi SoC Vendors

OrangePi, BananaPi, CherryPi, MangoPi, … you get the idea.

Loader Tools
sunxi-fel/xfel
rkflashtool
pyamlboot

Note: These compete on the multimedia device market, e.g., TV boxes.
Common FruitPi SoC Vendors

OrangePi, BananaPi, CherryPi, MangoPi, … you get the idea.

Loader Tools

sunxi-fel/xfel  rkflashtool  pyamlboot

Note: These compete on the multimedia device market, e.g., TV boxes.

https://linux-sunxi.org/Comparison_of_chip_maker_openness
More SoC vendors

Texas Instruments
AM{3,4,5,6}xx series used in Beaglebone Black

NXP
i.MX application processors used in MNT Reform laptop
Even more SoC vendors
Tracking upstream
Firmware

abootloader for Embedded boards

https://u-boot.readthedocs.io/
a fork of coreboot, with C removed, written in Rust.

https://github.com/oreboot/oreboot

U-Boot logo under CC BY 4.0 by Heinrich Schuchardt

Rust logo under CC BY 4.0,

https://github.com/rust-lang/rust-artwork

Ferris the crab from https://rustacean.net/
Firmware

a boot loader for
Embedded boards
https://u-boot.readthedocs.io/
Firmware

a boot loader for Embedded boards
https://u-boot.readthedocs.io/

a fork of coreboot, with C removed, written in Rust.
https://github.com/oreboot/oreboot
Firmware

a boot loader for Embedded boards
https://u-boot.readthedocs.io/

a fork of coreboot, with C removed, written in Rust.
https://github.com/oreboot/oreboot

U-Boot logo under CC BY 4.0 by Heinrich Schuchardt
Rust logo under CC BY 4.0, https://github.com/rust-lang/rust-artwork
Ferris the crab from https://rustacean.net/
Linux
## Linux

**Protocol** | **Location** |
--- | --- |
HTTP | https://www.kernel.org/pub/ |
GIT | https://git.kernel.org/ |
RSYNC | rsync://rsync.kernel.org/pub/ |

<table>
<thead>
<tr>
<th>Version</th>
<th>Release Date</th>
<th>Build Options</th>
<th>Links</th>
</tr>
</thead>
<tbody>
<tr>
<td>mainline</td>
<td>6.4</td>
<td>2023-06-25</td>
<td>[tarball] [pgp] [patch] [view diff] [browse]</td>
</tr>
<tr>
<td>stable</td>
<td>6.3.10</td>
<td>2023-06-28</td>
<td>[tarball] [pgp] [patch] [inc. patch] [view diff] [browse] [changelog]</td>
</tr>
<tr>
<td>longterm</td>
<td>6.1.36</td>
<td>2023-06-28</td>
<td>[tarball] [pgp] [patch] [inc. patch] [view diff] [browse] [changelog]</td>
</tr>
<tr>
<td>longterm</td>
<td>5.15.119</td>
<td>2023-06-28</td>
<td>[tarball] [pgp] [patch] [inc. patch] [view diff] [browse] [changelog]</td>
</tr>
<tr>
<td>longterm</td>
<td>5.10.186</td>
<td>2023-06-28</td>
<td>[tarball] [pgp] [patch] [inc. patch] [view diff] [browse] [changelog]</td>
</tr>
<tr>
<td>longterm</td>
<td>5.4.249</td>
<td>2023-06-28</td>
<td>[tarball] [pgp] [patch] [inc. patch] [view diff] [browse] [changelog]</td>
</tr>
<tr>
<td>longterm</td>
<td>4.19.288</td>
<td>2023-06-28</td>
<td>[tarball] [pgp] [patch] [inc. patch] [view diff] [browse] [changelog]</td>
</tr>
<tr>
<td>longterm</td>
<td>4.14.320</td>
<td>2023-06-28</td>
<td>[tarball] [pgp] [patch] [inc. patch] [view diff] [browse] [changelog]</td>
</tr>
<tr>
<td>linux-next</td>
<td><strong>next-20230629</strong></td>
<td>2023-06-29</td>
<td>[browse]</td>
</tr>
</tbody>
</table>

https://kernel.org
Specialized distros for Arm

- ArchLinuxArm https://archlinuxarm.org/
- openSUSE https://en.opensuse.org/Portal:Arm
- Fedora https://fedoraproject.org/wiki/Architectures/ARM
- Armbian https://www.armbian.com/
- Ubuntu https://ubuntu.com/download/server/arm

Problem: Many of these have specific images per board. Why? (many reasons)
Specialized distros for Arm

Arch Linux Arm
https://archlinuxarm.org/

openSUSE
https://en.opensuse.org/Portal:Arm

Fedora
https://fedoraproject.org/wiki/Architectures/ARM

Armbian
https://www.armbian.com/

Ubuntu
https://ubuntu.com/download/server/arm
Specialized distros for Arm

Arch Linux Arm
https://archlinuxarm.org/

openSUSE
https://en.opensuse.org/Portal:Arm

Fedora
https://fedoraproject.org/wiki/Architectures/ARM

Armbian
https://www.armbian.com/

Ubuntu
https://ubuntu.com/download/server/arm

Problem: Many of these have specific images per board. Why? (many reasons)
Cool projects
Does it have to be…

There are many microcontrollers, too. In general, they are more open. You can get one for free: Wettersonde

https://github.com/arnobert/rs41_rust
Does it have to be…

… a general purpose distro?
Does it have to be…

… a general purpose distro?
Make your own system!
Does it have to be…

… a general purpose distro?
Make your own system!

Frameworks

- Yocto/OpenEmbedded
- Buildroot
- OpenWrt
- AOSP (Android)
Does it have to be…

… a general purpose distro?  … an application processor?
Make your own system!

Frameworks
▶ Yocto/OpenEmbedded
▶ Buildroot
▶ OpenWrt
▶ AOSP (Android)

There are many microcontrollers, too. In general, they are more open.
You can get one for free: Wettersonde
https://github.com/arnobert/rs41_rust
Does it have to be…

… a general purpose distro? … an application processor?
Make your own system! There are many microcontrollers, too.

In general, they are more open.

Frameworks
▶ Yocto/OpenEmbedded
▶ Buildroot
▶ OpenWrt
▶ AOSP (Android)
Does it have to be…

… a general purpose distro?
Make your own system!

… an application processor?
There are many microcontrollers, too.
In general, they are more open.
You can get one for free: Wettersonde

Frameworks
▶ Yocto/OpenEmbedded
▶ Buildroot
▶ OpenWrt
▶ AOSP (Android)

https://github.com/arnobert/rs41_rust
u-root

an initramfs builder with Busybox-like tools written in Go

https://u-root.org
u-root

an initramfs builder with Busybox-like tools written in Go

https://u-root.org

cpu

   cpu command in Go, inspired by the Plan 9 cpu command

https://github.com/u-root/cpu
u-root

an initramfs builder with Busybox-like tools written in Go
https://u-root.org

cpu

cpu command in Go, inspired by the Plan 9 cpu command
https://github.com/u-root/cpu

How about USB CPU? Demo time!
what if we massively reduced the overall system complexity by
getting rid of all software we don’t strictly need, and instead built
up a minimal system from scratch entirely in Go

https://gokrazy.org/
what if we massively reduced the overall system complexity by getting rid of all software we don’t strictly need, and instead built up a minimal system from scratch entirely in Go

https://gokrazy.org/

Build Go appliances for the Raspberry Pi using gokrazy!
https://media.ccc.de/v/gpn21-78-build-go-appliances-for-the-raspberry-pi-using-gokrazy-
Racklet

*Racklet is a fully-integrated, miniature server rack.*

https://racklet.io/
Finding a community
Communication channels

There are wikis, forums, IRC, Matrix, Slack, Telegram groups...

Examples

https://linux-sunxi.org/
https://en.opensuse.org/openSUSE:IRC_list
https://slack.osfw.dev/

In person

Have you visited your local fablab, hackerspace or makerspace yet?
Communication channels

There are wikis, forums, IRC, Matrix, Slack, Telegram groups…
Communication channels

There are wikis, forums, IRC, Matrix, Slack, Telegram groups…

Examples
https://linux-sunxi.org/
https://en.opensuse.org/openSUSE:IRC_list
https://slack.osfw.dev/
Communication channels

There are wikis, forums, IRC, Matrix, Slack, Telegram groups…

Examples
https://linux-sunxi.org/
https://en.opensuse.org/openSUSE:IRC_list
https://slack.osfw.dev/

In person
Have you visited your local fablab, hackerspace or makerspace yet?
Thank you! :)
Related

Repurposing Gadgets (FOSSASIA Summit 2021)

Drivers from Outer Space (CLT 2022)

Speedy Distro Porting via the cpu Command
https://media.ccc.de/v/3802-speedy-distro-porting-via-the-cpu-command

Platform System Interface - Design und Evaluation holistischer Computerarchitektur (rC3 2022)
Follow Me

https://github.com/orangecms
https://twitter.com/orangecms
https://mastodon.social/@cyrevolt
https://youtube.com/@cyrevolt
https://twitch.tv/cyrevolt

Daniel Maslowski


License: CC BY 4.0 https://creativecommons.org/licenses/by/4.0/