QNAP NAS ohne QNAP

Digitale Souveränität selbstgemacht ;-)

Whoami

- Heiko Stübner <heiko@sntech.de>
- Linux-Kernel / U-Boot / Yocto + Debian
- Also in den unteren Etagen unterwegs
- Hobby-Projekt: Maintainer Rockchip-CPUs im Linux-Kernel

Digitale Souveränität

- vorhin allgemeiner Überblick
- Vortrag: Digiale Souveränität ohne Bullshit Bingo
- jetzt mal am Beispiel und selbstgemacht

Was ist ein NAS?

- Network Attached Storage
- Computer mit Platten und Netzwerkanschluss
- Gehäuse meist mit Hot-Swap-Laufwerksschächten
- GNU/Linux- oder BSD-basiertes Betriebssystem

Das kann man doch selber bauen?

- Single-Board-Computer (Pi o.ä.) + ein paar Platten
- vs. Für den Zweck gebautes Gerät
- “echtes NAS” sieht “wohnlicher” aus
- SBC mit 4 SATA Ports selten
- schon ein ITX Gehäuse ist größer
- Gehäuse mit Hot-Swap Schächten?

Nicht so

Schöne Hardware haben Sie da

- Geräte in ganz verschiedenen Formen und Größen
- Vom Mini-1-Platten-Ding (TS133)
- bis zum xHE 19 Zoll Rackgehäuse (TS433eU)
- nett anzusehende Weboberfläche
- aber bloß nicht zu genau hinschauen

Profis am Werk

FESTPLATTEN

WD-My-Cloud-NAS-Geräte kommen mit Backdoor-Account

Wer ein privates NAS aufsetzt, verspricht sich meist mehr Kontrolle über persönliche Daten als in der Cloud. Kritische Sicherheitslücken in Western-Digital-Geräten ermöglichen jedoch Angreifern, sich mit einem Backdoor-Account einzuloggen oder beliebig Dateien hochzuladen.

(15.01.2018)

Qnap blendet Bannerwerbung auf eigenen NAS-Systemen ein

Ob 500 oder 2.500 Euro: Immer mehr Nutzer ärgern sich über Pop-ups, die seit dem aktuellen Patch des Betriebssystems auftauchen.

(18.05.2020)

Sicherheitslücken betreffen praktisch alle Qnap-NAS-Systeme

Gleich drei Security-Probleme sind von Qnap gemeldet worden. Das Unternehmen rät zu einem sofortigen Update des Betriebssystems.

(05.06.2020)

Quelle: https://www.golem.de/ticker/

DEADBOLT

Ransomware verschlüsselt Daten auf Asustor-NAS

Die Ransomware Deadbolt wurde bereits auf Qnap-Geräten gefunden. Nun ist der Hersteller Asustor betroffen. Oft hilft nur der Kundendienst.

(23.02.2022)

RANSOMWARE

Qnap-NAS werden mit Zero Day angegriffen

Die Ransomware-Gruppe Deadbolt verschlüsselt NAS-Geräte von Qnap nach eigenen Angaben über eine bis dato unbekannte Sicherheitslücke.

(27.01.2022)

Qnap verschläft Patches und gelobt Besserung

Nach der Entdeckung teils schwerwiegender Sicherheitslücken in QTS und QuTS Hero liefert Qnap Patches und entschuldigt sich für die Verspätung.

(24.05.2024)

NACH PWN2OWN

QNAP und Synology patchen ausgenutzte NAS-Lücken

Neben Produkten von QNAP und Synology wurden bei dem Wettbewerb auch TrueNAS-Systeme attackiert. Erste Patches und Empfehlungen sind verfügbar.

(30.10.2024)

Aber Warum? - Zusammengewürfelt

- Marvel, Annapurna Labs, Realtek, Intel, Rockchip, …
- Auswahl des System-on-Chip offenbar hauptsächlich nach Preis
- Peripherie in SoCs funktioniert unterschiedlich
- Vendor-Kernel / BSP mit Millionen Zeilen out-of-tree-Code
- Jeder Soc mit eigenem ganz speziellen Kernel
- Noch schnell NAS-Kram rein gehackt
- Da gibt es fähige Entwickler, aber
- Technical Debt aufräumen bringt keinen Umsatz

#ifdef CONFIG_MACH_QNAPTS

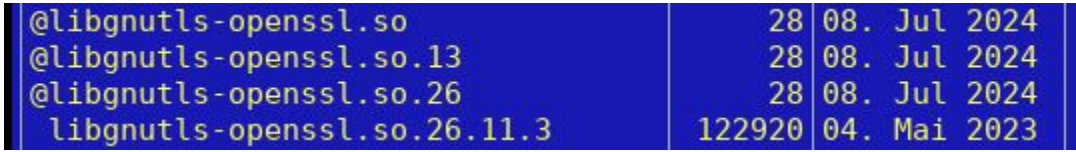
- Vendor-Kernel schon schlecht
- Code is voll mit #ifdef CONFIG_MACH_QNAPTS Blöcken
- Auch in sehr zentralen Kernelteilen
- Auch Memory-Manangement mm/memory.c, mm/oom_kill.c
- Auch SCSI drivers/scsi/scsi.c, drivers/scsi/scsi_lib.c, …
- “Trust Me Bro”, YOLO, oder so
- CRA hat noch keiner gehört
- kernel_booting_buzzer() in drivers/tty/serial/8250/8250_core.c

init/main.c:

```
static void __init
do_basic_setup(void)
{
        cpuset_init_smp();
        driver_init();
        init_irq_proc();
        do_ctors();
        usermodehelper_enable();
        do_initcalls();
#if defined(CONFIG_MACH_QNAPTS)
#if … ||
defined(CONFIG_ARCH_ROCKCHIP)
    // Bug#71780 [TS-231+] Need a …
    kernel_booting_buzzer();
#endif
#endif
}
```

Aber Warum? - Antike Mumien

- QNAP “Rettungssystem”
- libgnutls-openssl.so.26.11.3 findet Google nur in 2008er Blogbeitrag
- libc 2.23, selbst für Debian oldoldstable (2.28) zu alt
- jüngstes Debian Paket libc-2.23 von August 2016



Das wird garantiert nicht nur bei QNAP so sein!

Ist also nur die Firmware die ich mir angesehen habe.

Eigene Software laufen lassen (1)

- Synology / QNAP / etc haben eigene Paketsysteme / SSH-Zugang
- Entwickler-Programme wo man sich anmelden kann
- Gerätefirmware bleibt aber bestehen
- Abhängig vom Wohlwollen der Hersteller (Abo-Modell oder sowas)
- Noch ein anderes System lernen
- angelerntes Wissen auch in 6 Monaten noch haben
- Dead-End

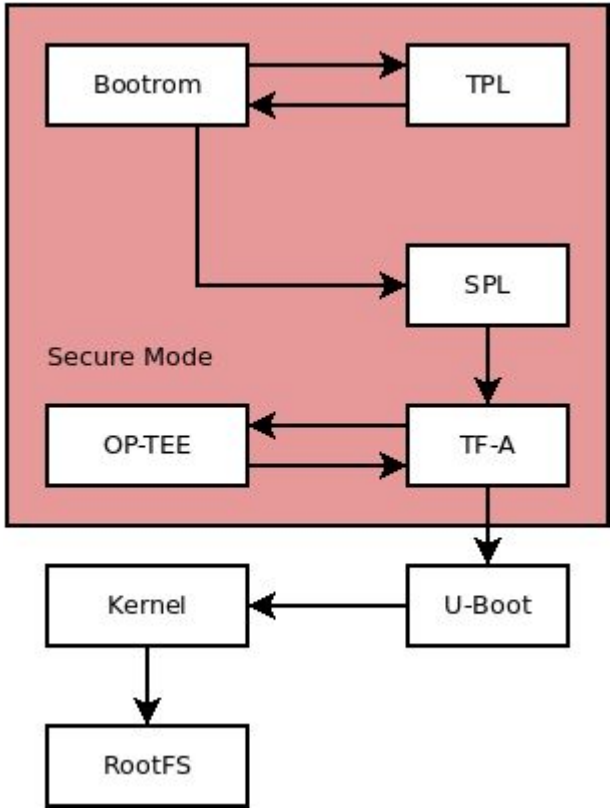
Eigene Software laufen lassen (2)

- Das Internet fragen (“Debian on $Gerätename”, etc)
- Findet machmal Foren mit Kernel-Images und lustigen Anleitungen
- Oft nur bis “läuft so” entwickelt, Feinschliff + Upstreaming fehlt
- Teile der Firmware / Bootloader bleiben
- Recovery wenn was schief geht?

- Tolle Arbeite keine Frage, aber >26 Seiten Foren-Threads?
- Was ist wenn ich in 12 Monaten mal den Kernel update will?

Wunschtraum

- Frische Software ab Bootrom
- Updates möglich
- Wissen auch in 6 Monaten noch da
- Keine Super-Spezial-Lösungen
- Alles “von der Stange” und Upstream



(M)ein spezielles Gerät

- Möglichst keine Hardware kaufen, die ich nicht verstehe
- QNAP TS-433 (133, 233, …)
- Rockchip RK3568 SoC
- Mainline Linux + U-Boot ziemlich gut
- 4 Cortex-A55
- Mali-G52 (Bifrost)
- NPU
- diverse Video-Encoder
- SATA + PCIe

git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/tree/MAINTAINERS

```
3028  ARM/Rockchip SoC support
3029  M:      Heiko Stuebner <heiko@sntech.de>
3030  L:      linux-arm-kernel@lists.infradead.org (moderated for non-subscribers)
3031  L:      linux-rockchip@lists.infradead.org
3032  S:      Maintained
3033  T:      git git://git.kernel.org/pub/scm/linux/kernel/git/mmind/linux-rockchip.git
3034  F:      Documentation/devicetree/bindings/i2c/i2c-rk3x.yaml
3035  F:      Documentation/devicetree/bindings/mmc/rockchip-dw-mshc.yaml
3036  F:      Documentation/devicetree/bindings/spi/spi-rockchip.yaml
3037  F:      arch/arm/boot/dts/rockchip/
3038  F:      arch/arm/mach-rockchip/
3039  F:      drivers/*/*/*rockchip*
3040  F:      drivers/*/*rockchip*
3041  F:      drivers/clk/rockchip/
3042  F:      drivers/i2c/busses/i2c-rk3x.c
3043  F:      sound/soc/rockchip/
3044  N:      rockchip
```

Deswegen ganz Wichtig

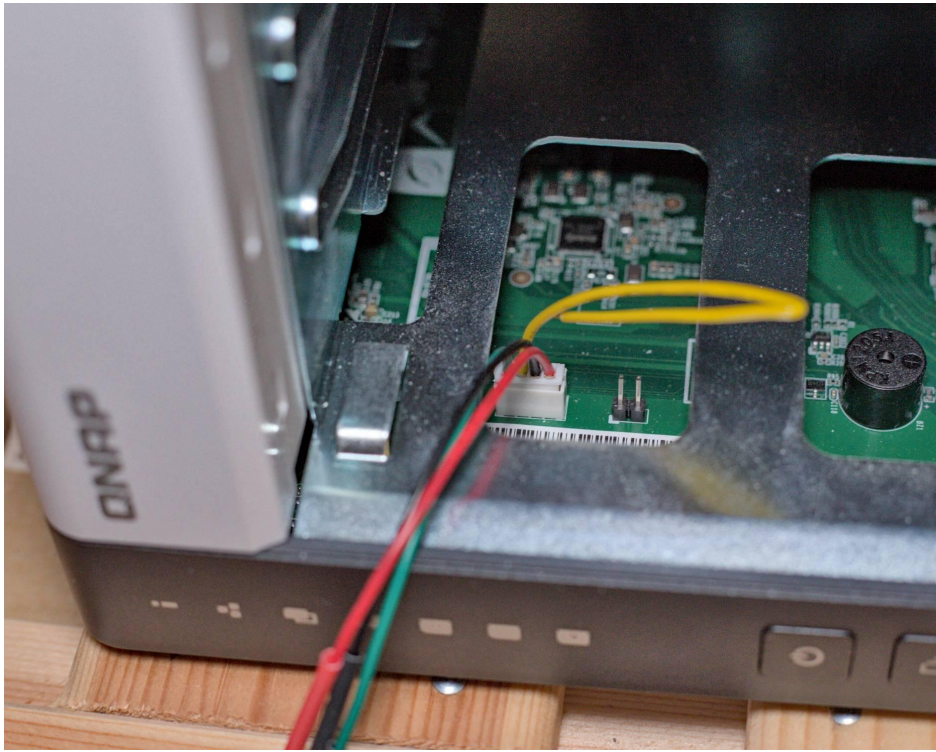
- Wir reden hier über eine spezielle Familie von Geräten
- TS-133, TS-216G, TS-233, TS-433, TS-433eU (19”), …
- Geht auch mit anderen SoCs
- Aber beginnt eben von vorn
- Die Geräte sehen von außen auch wirklich alle gleich aus
- Domain-Knowledge hilft

Jetzt steht das Teil da - Was jetzt

- Konsolen-Zugang
- Schauen was läuft
- Hardwarekomponenten ermitteln
- Aussperren verhindern
- Kernel
- Bootloader
- Root-Dateisystem

Zugang verschaffen

- embedded Geräte - serielle Konsole
- RX, TX, GND - oft nur Lötpunkte
- QNAP-Geräte mit echter Buchse
- JST-PH-Buchse auf der Platine
-
```
 ,_  _.
 |1234|  1=TX 2=VCC
 `----'  3=RX 4=GND
```
- `pyserial-miniterm /dev/ttyUSB 115200`



Neugier - das Rescue-System

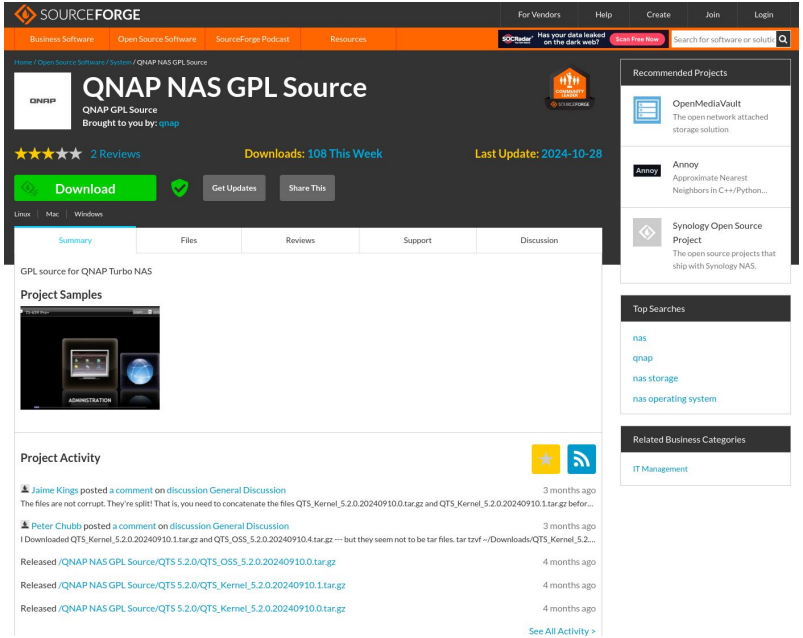
- Auf emmc ist einfaches (uraltes) Rescue-System
- kann auf HW zugreifen und soll eigentlich ein QNAP-Release installieren
- “hal_app” … Monsterapplikation, die alles mögliche tut
- sieht aus als wäre es über viele Jahre gewachsen
- Vendor-Kernel so verborgen, dass er zum historischen Userspace passt

Hardwarekomponenten finden

- SoC ist bekannt
- Hersteller folgen meist Board-Layout-Vorschlägen
- Alle Boards sehen also meist ähnlich aus
- I2C-Geräte (PMIC, RTC)
- LEDs
- SATA Ports
- USB
- Serieller Port
- was die anderen GPIOs so tun
- Devicetree zur Rettung

Quellen (der Informationen)

- Qnap veröffentlicht Sourcen
- Sourceforge als Plattform (Retro)
- tar Archive für Kernel und “sonstiges”
- Gigabytes ohne Revisionsinformationen
- dafür mit Compile-Rückständen
- ~~U-Boot-Quellen, nicht die vom Gerät~~
  U-Boot-Quellen seit Anfang März
- Wir sind lange genug im
  Devicetree-Zeitalter für Geräte



Devicetree

- Hardwarebeschreibung
- die meisten Informationen an einem Platz
- DT soll Spezifikation folgen
- der DT vom Vendor-Kernel passt nicht zum Rest der Welt
- aber eine nette Informationsquelle
- aber oft unvollständig (formal korrekt <-> geht doch)

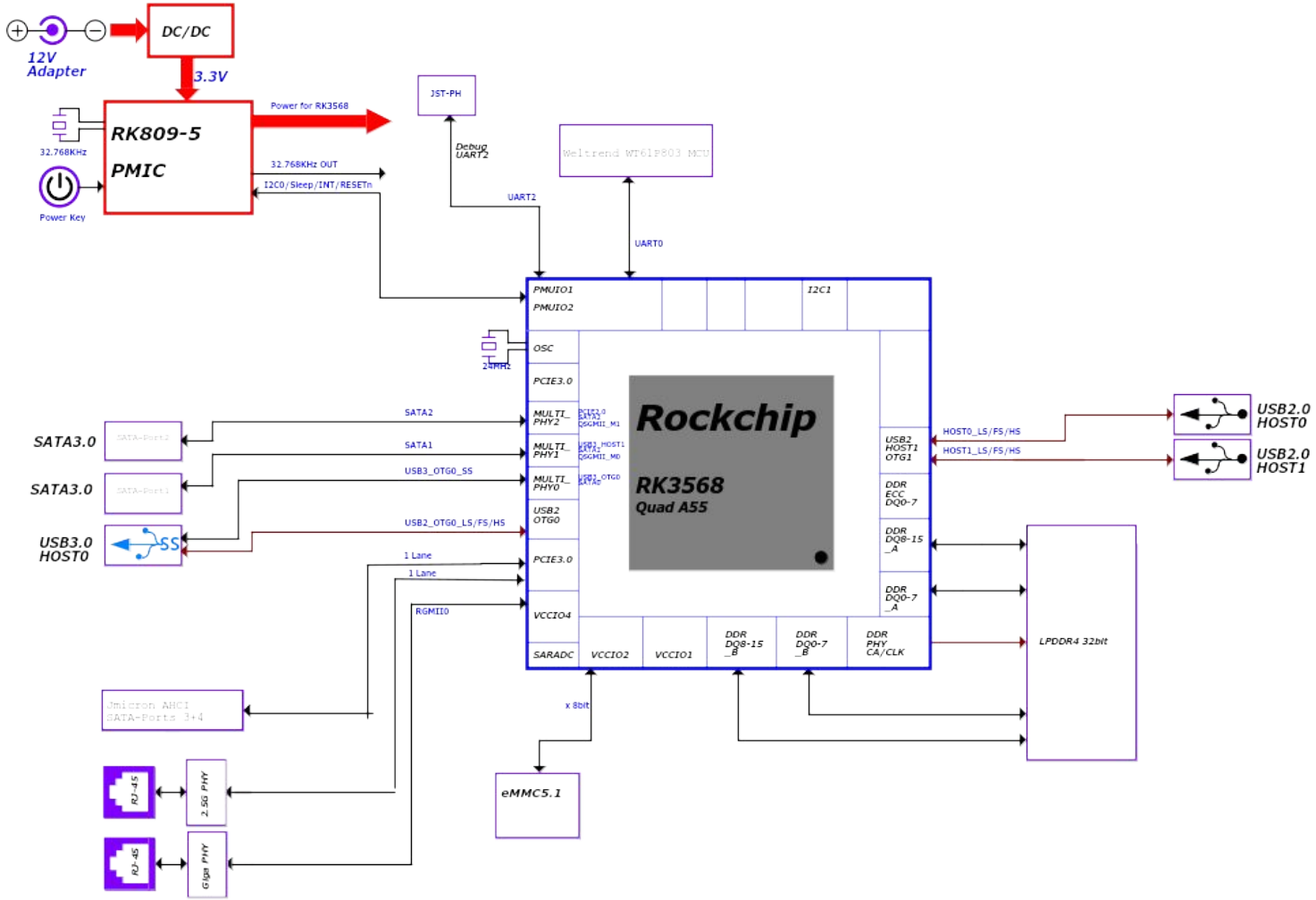
```
/dts-v1/;

#include <dt-bindings/gpio/gpio.h>
#include <dt-bindings/pinctrl/rockchip.h>
#include "rk3568.dtsi"

/ {
	model = "QNAP TS-433";
	compatible = "rockchip,rk3568-evb1-v10", "rockchip,rk3568";

	chosen: chosen {
		stdout-path = "serial2:115200n8";
	};

	dc_12v: dc-12v {
		compatible = "regulator-fixed";
		regulator-name = "dc_12v";
		regulator-always-on;
		regulator-boot-on;
		regulator-min-microvolt = <12000000>;
		regulator-max-microvolt = <12000000>;
	};
…
```

Gruselkabinet

- Immer Volle Pulle?
- HDD-LEDs nicht im DT, GPIOs via sysfs
- Informations-EEPROMs via manuellen I2C Kommandos aus Userspace

```
&cpu0_opp_table {
    /delete-node/ opp-408000000;
    /delete-node/ opp-600000000;
    /delete-node/ opp-816000000;
    /delete-node/ opp-1104000000;
    /delete-node/ opp-1416000000;
    /delete-node/ opp-1608000000;
    /delete-node/ opp-1800000000;
//    /delete-node/ opp-1992000000;
};
```

Qnap TS433 Block Diagram



Weitere Quellen

```
[/sys/kernel/debug] # cat gpio
gpiochip0: GPIOs 0-31, parent: platform/fdd60000.gpio, gpio0:
 gpio-5   (                    |vcc5v0-otg-regulator) out hi
 gpio-6   (                    |vcc5v0-host-regulato) out hi
 gpio-13  (                    |sysfs                ) in  hi .... reset-button
 gpio-14  (                    |sysfs                ) in  hi .... copy-button
 gpio-23  (                    |reset                ) out hi
 gpio-28  (                    |gpio-regulator       ) out hi

gpiochip1: GPIOs 32-63, parent: platform/fe740000.gpio, gpio1:
 gpio-61  (                    |sata led gpio        ) out hi
 gpio-62  (                    |sata led gpio        ) out hi
 gpio-63  (                    |sata led gpio        ) out hi

gpiochip2: GPIOs 64-95, parent: platform/fe750000.gpio, gpio2:
 gpio-64  (                    |sata led gpio        ) out hi
 gpio-94  (                    |reset                ) out hi

gpiochip3: GPIOs 96-127, parent: platform/fe760000.gpio, gpio3:

gpiochip4: GPIOs 128-159, parent: platform/fe770000.gpio, gpio4:
```

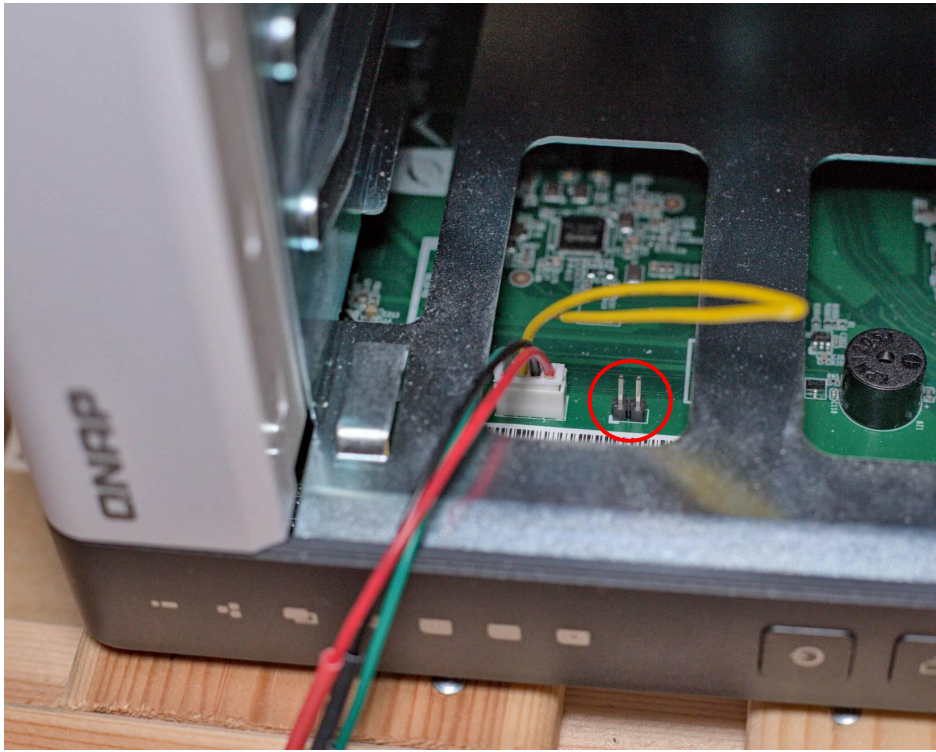
Weitere Quellen

[/sys/kernel/debug/pinctrl/pinctrl-rockchip-pinctrl] # cat pinconf-pins
Pin config settings per pin
Format: pin (name): configs
pin 0 (gpio0-0): input bias pull down, output drive strength (4 mA), input schmitt enabled, pin output (0 level), slew rate (1)
pin 1 (gpio0-1): input bias disabled, output drive strength (4 mA), input schmitt enabled, pin output (1 level), slew rate (1)
pin 2 (gpio0-2): input bias pull down, output drive strength (4 mA), input schmitt enabled, pin output (0 level), slew rate (1)
pin 3 (gpio0-3): input bias pull up, output drive strength (4 mA), input schmitt enabled, pin output (1 level), slew rate (1)
pin 4 (gpio0-4): input bias pull up, output drive strength (4 mA), input schmitt enabled, slew rate (1)
pin 5 (gpio0-5): input bias disabled, output drive strength (4 mA), input schmitt enabled, pin output (1 level), slew rate (1)

Mit anderem RK3568-Board vergleichen -> zeigt versteckte Änderungen

Failsafe - wenn was schief geht

- Ähnliche Systeme auf allen SoCs
- On-Chip Bootrom
- Firmware Download über USB
- Auf allen Rockchip SoCs ähnlich
- Kennste Eins, kennste Alle
- Muss nur die Elemente finden
- Maskrom-Override als Jumper :-)
- Vorderer USB3-A Port auch Gadget



Failsafe - Rollback

- In Rescue-System booten + USB Stick anschliessen
- dd if=/dev/mmcblk0 of=/mnt/usb/emmc.img

- `git clone https://github.com/rockchip-linux/rkbin.git`
- `tools/boot_merger RKBOOT/RK3568MINIALL.ini`
- `apt-get install rkdeveloptool`
- `rkdeveloptool db rk356x_spl_loader_v1.21.113.bin`
- `rkdeveloptool wl 0 emmc.img`

Kernel-Image

- RK3568 im Mainline-Kernel ist gut unterstützt
- die meisten (alle?) Komponenten haben bereits Treiber
- ein NAS ist ja keine exotische Hardware
- also einfach ein 6.12-Image mit der generischen defconfig bauen

```
apt-get install build-essential gcc-14-aarch64-linux-gnu
make ARCH=arm64 CROSS_COMPILE=aarch64-linux-gnu- defconfig
make ARCH=arm64 CROSS_COMPILE=aarch64-linux-gnu- Image
```

Kernel-Image und dann?

- Kernel-Image und Devicetree
- DT ähnlich zu allen anderen RK3568-Boards
- Ein spannendes Hardware-Detail PCIe-Bifurcation
- 2 PCIe Controller im RK3568
- 1 Controller mit 2 Lanes, 2 Controller mit je 1 Lane
- SATA-Controller und 2.5G Ethernet
- Vorher noch nicht auf RK3568 boards gesehen

Booten

- Aber wie booten?
- Vendor-U-Boot macht schlechte Laune
- RK3568 ist in Mainline U-Boot super unterstützt
- Existierendes Board kopieren
- Alten Namen durch TS433 ersetzen
- Kernel-DT dazu
- dann geht das schon ;-)

Erster Start

- `rkdeveloptool db rk356x_spl_loader_v1.21.113.bin`
- `rkdeveloptool wl 64 u-boot-rockchip.bin`
- `rkdeveloptool rd`

- `U-Boot 2025.01 (Mar 11 2025 - 11:03:51 +0100)`

```
Model: Qnap TS-433-4G NAS System 4-Bay
DRAM:  4 GiB
PMIC:  RK809 (on=0x40, off=0x00)
Core:  318 devices, 31 uclasses, devicetree: separate
```

Bootmedien

- Bootreihenfolge emmc, usb
- U-Boot sucht extlinux/extlinux.conf jeweils auf 1. Partition
- kann also einfach ein System vom USB-Stick booten

```
label kernel
   kernel /boot/Image
   fdt /boot/rk3568-qnap-ts433.dtb
   initrd /boot/initrd.img-6.13.0-00016-g5758ade23ad0-dirty
   append earlycon=uart8250,mmio32,0xfe660000 console=tty1 console=ttyS2,115200n8 rw
                          root=/dev/sda1 rootwait rootfstype=ext4 init=/sbin/init
```

- Im Vendor-U-Boot wäre alles viel aufwändiger
- (kein distroboot)

RootFS

- qemu-user-static - Binfmt-support, arm64 binaries transparent auf x86
- debootstrap

```
debootstrap –arch=arm64 –foreign testing /mnt/stick
chroot /mnt/stick

/debootstrap/debootstrap –second-stage
passwd
apt-get install initramfs-tools
update-initramfs -c -k $kernver
```

MCU - die große Unbekannte

- externer Mikrocontroller für zusätzliche Funktionen
- Weltrend WT61P803 - zuerst in TV-Geräten gesichtet
- LEDs, Power-Knopf, Lüfter
- Anbindung per UART
- Nicht dokumentiert
- Qnap hal_app und Bibliotheken übernehmen serielle Kommunikation
- MCUs per UART offenbar beliebt bei NAS Herstellern
- z.B. synology,power-off

Kleiner Lauschangrif

- interceptty (https://github.com/geoffmeyers/interceptty) zum mitschneiden
- mv /dev/ttyS0 /dev/ttyS0hw
  ./interceptty /dev/ttyS0hw /dev/ttyS0 &
- hal_app --se_buzzer enc_id=0,mode=0
  < 0x40 (@)
  < 0x43 (C)
  < 0x32 (2)
  < 0x31 (1)
  > 0x40 (@)
  > 0x30 (0)
  > 0x70 (p)
- macht Beep :-)
- hal_app hat sogar einen Hilfetext, manche Kommandos tun nichts

Echter Treiber

- Userspace-Programm für HW-Dinge ein No-Go
- Bsp. Lüftersteuerung basierend auf CPU-Temperatur
- Kernel hat seit ein paar Jahren “serdev”
- 4 Subsysteme involviert (mfd, led, hwmon, inpu)
- der MCU-Treiber hat von allem am längsten gedauert

Aktueller Zustand

- Devicetree im Mainline-Kernel (v6.9)
- Alle Hardwarekomponenten (außer MCU) unterstützt (v6.12)
- MCU-Treiber im Mainline-Kernel (vollständig in v6.14)
- Gelernt wie Serdev funktioniert
- U-Boot-Bootloader unterstützt den TS433 (v2025.01)
- Barebox-Bootloader-Support ist gerade in Arbeit
- Komfortable Debian-Installation ist in Arbeit

Nutzungsvarianten

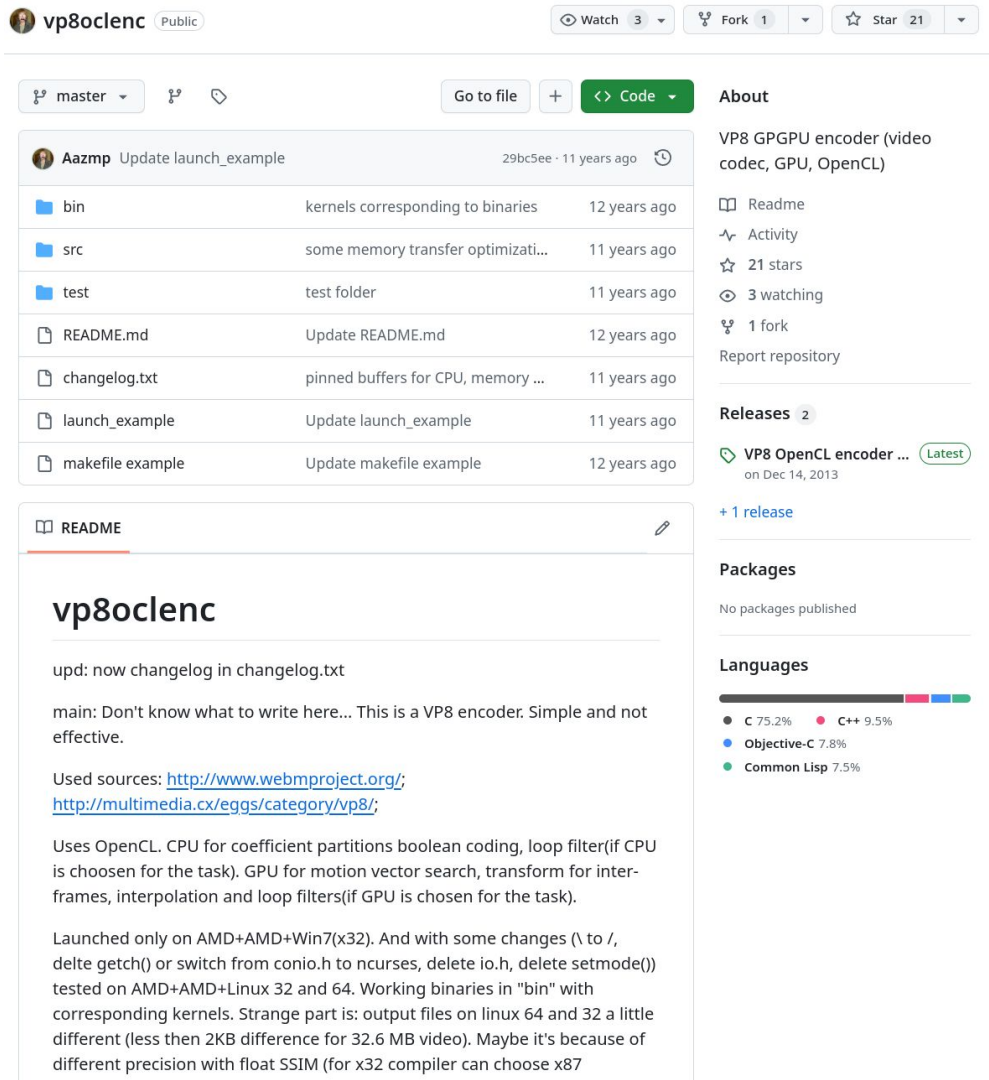
- Einfach nur NFS - also selber machen
- Nextcloud
- OpenMediaVault
- Alle großen Lösungen nisten sich sehr tief im System ein

Was lernen wir daraus

- Mit Dokumentation, Linux + U-Boot wäre ein Nachmittag gewesen
- So aber deutlich mehr Aufwand
- Gerät erforschen macht Spaß
- Aber so schnell brauch ich das nicht nochmal
- QNAP könnte ruhig etwas freigiebiger mit Informationen sein
- Frage nach MCU Doku … “ja gern, aber nur mit Knebel-NDA”

Ausblick - GPU

- Mesa unterstützt die Mali GPU im RK3568
- OpenGL nützt hier wenig
- OpenCL Unterstüztung ist vorhanden
- z.B. VP8 OpenCL encoder https://github.com/Aazmp/vp8oclenc



vp8oclenc Public

Watch 3 · Fork 1 · Star 21

master · Go to file · Code

Aazmp Update launch_example 29bc5ee · 11 years ago

bin	kernels corresponding to binaries	12 years ago
src	some memory transfer optimizati...	11 years ago
test	test folder	11 years ago
README.md	Update README.md	12 years ago
changelog.txt	pinned buffers for CPU, memory ...	11 years ago
launch_example	Update launch_example	11 years ago
makefile example	Update makefile example	12 years ago

README

vp8oclenc

upd: now changelog in changelog.txt

main: Don't know what to write here... This is a VP8 encoder. Simple and not effective.

Used sources: http://www.webmproject.org/; http://multimedia.cx/eggs/category/vp8/;

Uses OpenCL. CPU for coefficient partitions boolean coding, loop filter(if CPU is choosen for the task). GPU for motion vector search, transform for inter-frames, interpolation and loop filters(if GPU is chosen for the task).

Launched only on AMD+AMD+Win7(x32). And with some changes (\ to /, delte getch() or switch from conio.h to ncurses, delete io.h, delete setmode()) tested on AMD+AMD+Linux 32 and 64. Working binaries in "bin" with corresponding kernels. Strange part is: output files on linux 64 and 32 a little different (less then 2KB difference for 32.6 MB video). Maybe it's because of different precision with float SSIM (for x32 compiler can choose x87

About

VP8 GPGPU encoder (video codec, GPU, OpenCL)

Readme
Activity
21 stars
3 watching
1 fork
Report repository

Releases 2

VP8 OpenCL encoder ... Latest
on Dec 14, 2013

+ 1 release

Packages

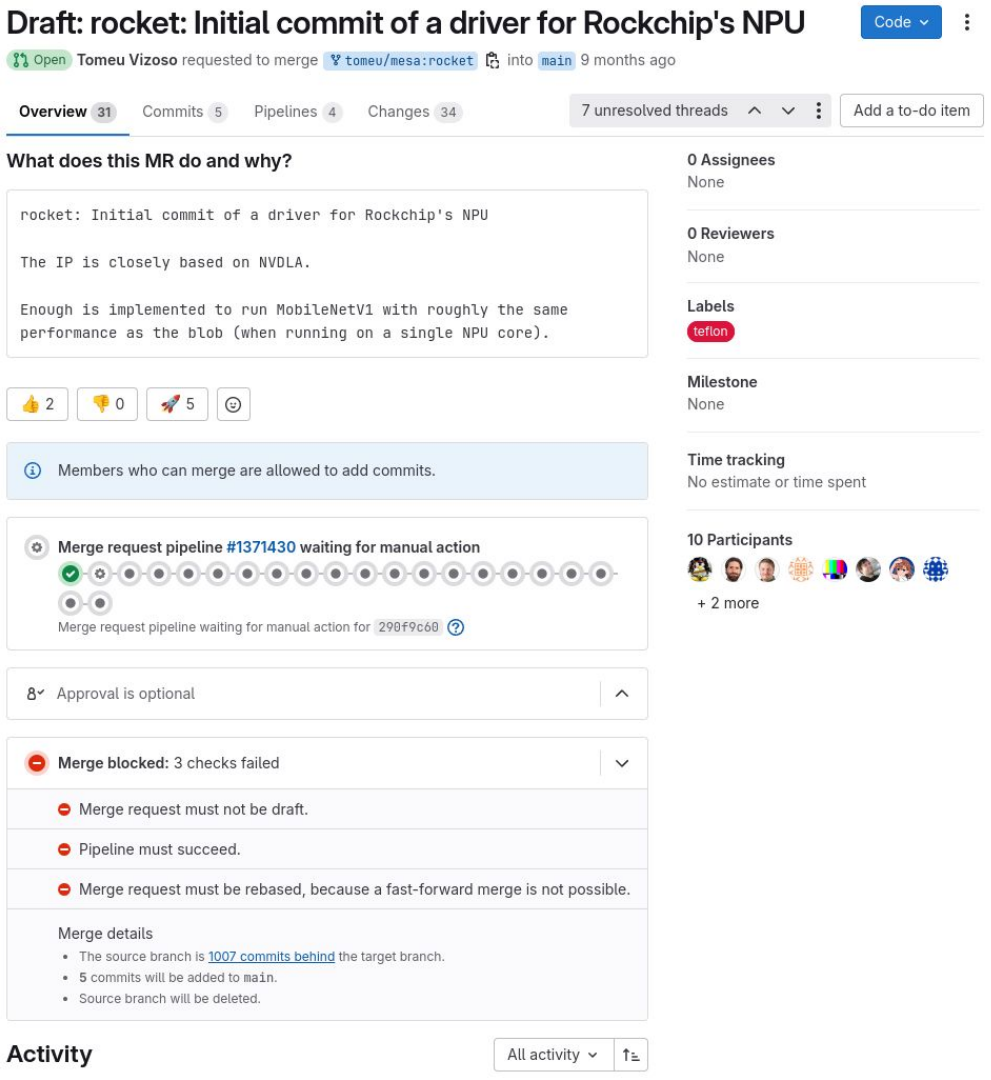
No packages published

Languages

C 75.2% · C++ 9.5% · Objective-C 7.8% · Common Lisp 7.5%

Ausblick - NPU

- proprietäre Bibliotheken
- proprietäre Formate für KI-Modelle
- erfordern Konvertierung
- geht oft schief / aufwändig
- WIP Mesa-Treiber für die NPU existiert bereits
- Integration als Tensorflow-Delegate



Fragen?

E-Mail: heiko@sntech.de