Running FOSS on a Thermal Camera

Sebastian Reichel

Collabora

Who am I?

- Kernel Engineer at Collabora
- Maintainer of kernel's power-supply subsystem
- Debian Developer
- Living in Oldenburg, Germany
  - Co-Founder of the local hackerspace
  - Deputy Lead of the Fire Brigade Diver Squad

What’s a thermal camera?

- Camera for infrared (8-14µm) instead of visible spectrum (380nm - 750nm)
- Usually low resolution (Kilopixel instead of Megapixel)
- Usually low speed
- Best known vendor for sensors is FLIR Systems
- US export restrictions
  - 9 FPS, 17 um pixel pitch (~640x480)

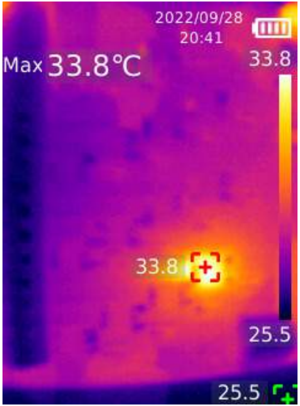
What are they used for?

- target search (hence the export restrictions)
- search and rescue
  - finding hot spots after a fire
  - finding persons, especially in winter
- night vision & surveillance
- building inspection
- electronic fault search
- . . .

UNI-T UTi260B

- Thermal Cameras used to be quite expensive or super low resolution
- Found a 256x192 camera with 25 FPS for ∼300€
- For reference, the model we use at the fire brigade costs 2000€ and has roughly the same specs

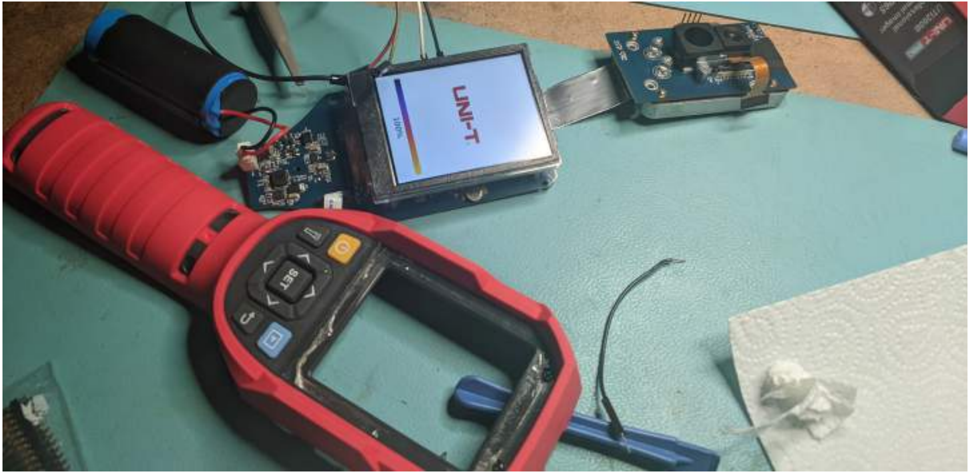
PCB fault



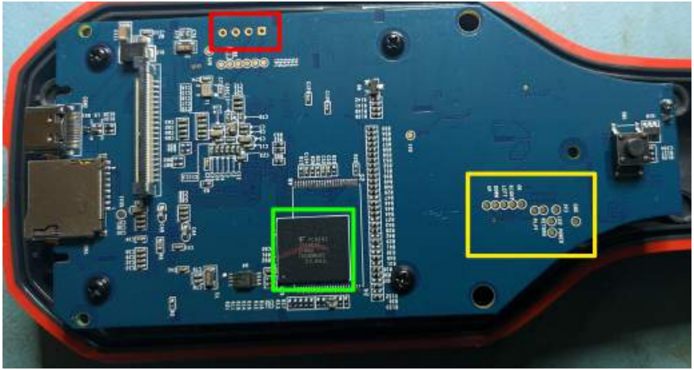
Software is quite bad

- Needs roughly 30 seconds to boot
- Taking an image keeps the overlay (like a screenshot)
- System is a bit laggy
- Colormap automatically rescales and cannot be locked

Void warranty - Let’s look inside

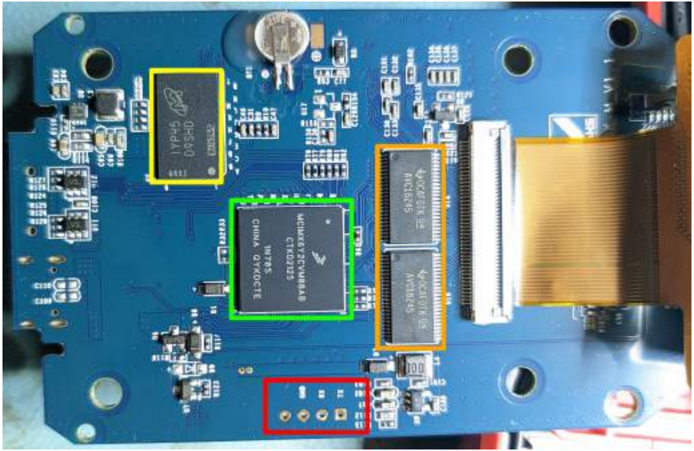


Hardware



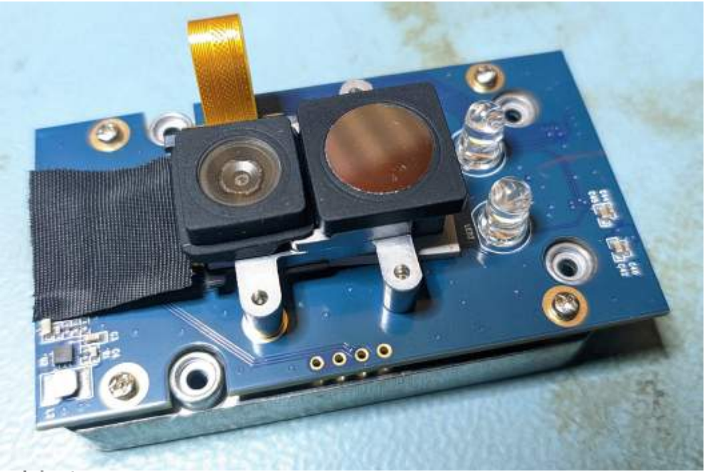
- Green
  - THGBMNG5 D1LBAIL
  - 4GB eMMC
- Yellow
  - Debug Pads
  - Plaintext Labels :)
- Red
  - unpopulated connector
  - UART?

Hardware



- Yellow
  - D9SHD
  - Micron 4Gb DDR3-1866
- Orange
  - AVC16245
  - TI bus transceiver
- Green
  - MCIMX6Y2CVM08AB
  - NXP i.MX6ULL
- Red
  - GND / RX / TX
  - UART ?

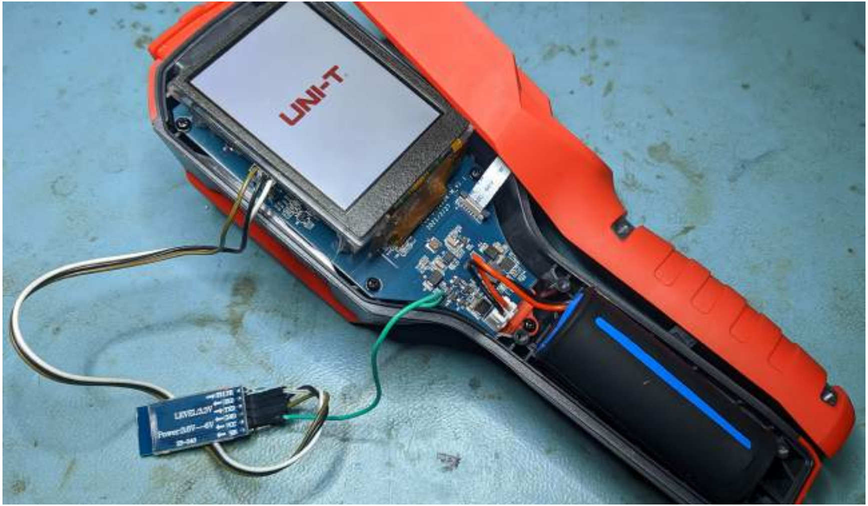
Camera Module Hardware



Hardware Hacking

- Solder in Connector
- Power on the system
- Measure Voltages
  - usually one of 1.8V, 3.3V or 5V
- Use matching USB serial adapter
  - 5V on 1.8V pin breaks the pads and or device!

Everything is better with Bluetooth



UART output

```
U-Boot 2015.04 (Mar 25 2020 - 13:40:37)

CPU:   Freescale i.MX6ULL rev1.1 at 396 MHz
CPU:   Temperature 36 C
Reset cause: POR
Board: MX6UL 14x14 EVK
I2C:   ready
DRAM:  512 MiB
MMC:   FSL_SDHC: 0, FSL_SDHC: 1
[...]
Starting kernel ...
[    0.000000] Booting Linux on physical CPU 0x0
[    0.000000] Linux version 3.14.38-6UL_ga (ubuntu@ubuntu) (gcc version 4.6.2 20110630 (prerelease)
[    0.000000] CPU: ARMv7 Processor [410fc075] revision 5 (ARMv7), cr=10c53c7d
[    0.000000] CPU: PIPT / VIPT nonaliasing data cache, VIPT aliasing instruction cache
[    0.000000] Machine model: Freescale i.MX6 UltraLite 14x14 EVK Board
[...]
UNIT login:
```

Investigating hardware

- booted into existing system with `init=/bin/sh`
- change root password
- reboot
- investigate running system
  - there are two V4L2 devices
    - one is i.MX6ULL CSI
    - one is USB Video Class
  - there is one binary running
    - unstripped, linked against Qt and OpenCV
  - there is basically no optimization
    - system has ALSA, Can, Network, Bluetooth, . . .
- let's switch to our own setup

U-Boot

```
=> printenv bootcmd
bootcmd=
    if mmc rescan; then
        if run loadbootscript; then
            run bootscript;
        else
            if test ${bootdev} = sd1; then
                echo update firmware.........;
                run update_from_sd;
            else
                echo mmc boot..........;
                if run loadimage;
                    then run mmcboot;
                    else run netboot;
                fi;
            fi;
        fi;
    else
        run netboot;
    fi;
```

U-Boot

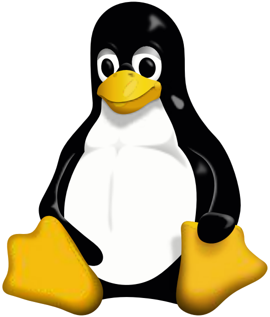
- loadbootscript tries to load boot.scr from eMMC. . .
- . . . but that's not used by UNI-T
- modify that to check for boot.scr on SD card
- TODO: find an exploit in original FW that can do this

```
=> printenv loadbootscript
loadbootscript=fatload mmc ${mmcdev}:${mmcpart} ${loadaddr} ${script};
=> printenv mmcdev
mmcdev=1
=> printenv mmcpart
mmcpart=1
=> setenv loadbootscript 'fatload mmc 0:1 ${loadaddr} ${script};'
=> saveenv
```

Prepare SD card

- Create FAT partition for U-Boot with boot.scr
-
  ```
  echo "Executing boot.scr from MicroSD card..."
  setenv mmcroot '/dev/mmcblk0p2 rootwait rw console=ttymxc0,115200n8'
  setenv fdt_file imx6ull-uti260b.dtb
  setenv mmcdev 0
  run loadimage
  run loadfdt
  run mmcargs
  bootz ${loadaddr} - ${fdt_addr}
  ```
- debootstrap –arch=armhf testing /mnt/sdcard

Kernel & DT



- i.MX6ULL is supported mainline
  - `make imx_v6_v7_defconfig`
  - `sed -i "s/=m/=y/g" .config`
  - (optionally) go through config and remove unnecessary stuff
    - e.g. Bluetooth and WLAN
    - my zImage went down from 40MB to 3.6MB
    - faster boot, faster copy-to-device => faster test cycles
- start with very small device DT
  - Model
  - UART
  - Memory
  - SD card

Watchdog

- device has no hard reset button
  - option 1: wait for battery to be empty
  - option 2: open device and disconnect battery
  - option 3: make sure device never hangs
  - option 4: modify hardware
    - (e.g. add normally closed reed switch)

Improving hardware support

- Original system leaks information
  - DT blob can be decompiled
    - `dtc -I dtb -O dts imx6ul-14x14-evk.dtb > dump.dts`
  - GPIOs can be investigated in sysfs
    - some of them might not be GPIOs in mainline
    - e.g. `<&gpio2 2>` controls the flashlight LED (leds-gpio)
    - e.g. `<&gpio2 3>` is the power button (gpio-keys)
- Bootloader also leaks information
  - Original Linux just configures LCDIF
  - But U-Boot states `LCD st7789v init successfully!`
  - That's an SPI controller
  - bootloader pinmux reveals the right SPI port

USB

- The device has a USB-C port
  - Used for charging, but also supports USB gadget mode
  - Original FW offers to screencast via USB UVC
- Add bootscript to enable USB gadget mode with ECM
  - Device will provide itself as ethernet device
  - One can SSH to it

Hardware that did not work OOTB

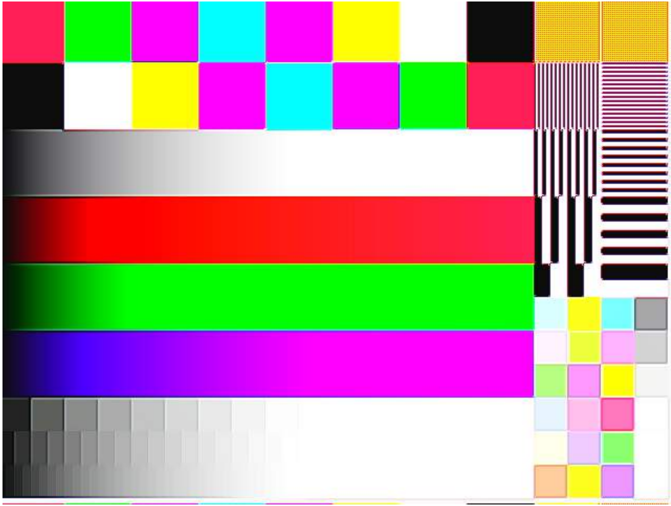
- Battery Handling (6.5)
  - Charger is TP5000 (found on PCB)
  - Has a GPIO to report that a charger is connected
  - There's a second GPIO to report that charging is done
  - For Battery only Voltage is available via ADC
  - there are existing `gpio-charger` and `adc-battery` drivers
  - https://lore.kernel.org/all/20230317225707.1552512-1-sre@kernel.org/
- Display Driver (6.6)
  - Labeled "Inanbo T28CP45TN89 v17"
  - Tried to use existing ST7789V driver
  - flipped some bits and got it working
  - https://lore.kernel.org/all/20230714013756.1546769-1-sre@kernel.org/
- Cameras

Optical Camera

- i.MX6ULL CSI driver recently moved from staging
- Optical sensor is Galaxycore GC0308 (640x480 / 0.3MP)
  - No mainline driver :(
  - datasheet is public, but hard to read
  - sensor has some ISP functionality (like auto gain)
  - there's a bunch of low quality out of tree drivers
  - many configurations break the i.MX6ULL CSI driver
  - still WIP

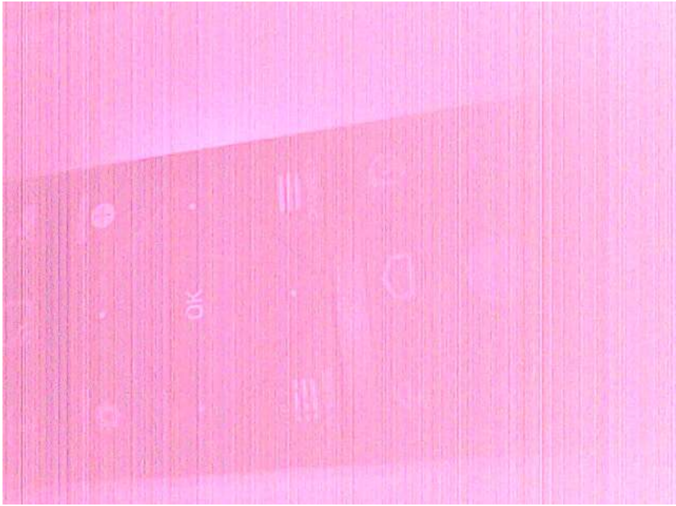
COLLABORA

Optical Camera



Open First

Optical Camera



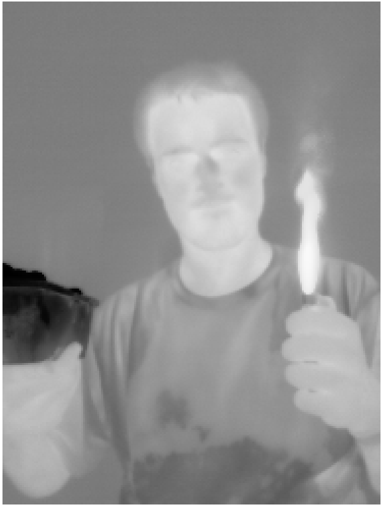
Thermal Camera

- Exposes UVC
- It lies about data format
- In addition to UVC also takes vendor USB control commands
    - e.g. high gain (up to 100°C) VS low gain (up to 550 °C)

```
gst-launch-1.0 v4l2src device=/dev/video1 !
video/x-raw,format=YUY2,width=256,height=384,framerate=25/1 !
videocrop top=192 ! videoconvert ! videoflip method=clockwise !
videoconvert ! video/x-raw,format=GRAY8 ! videoconvert ! videoscale !
waylandsink
```

Thermal Camera (without crop & gray8 convert)

Fire and Ice



Thermal Camera UART (57600 baud)

```
U-Boot 2016.11 (Mar 16 2021 - 02:20:19 +0000)

Sheipa Platform -- Taroko CPU: 500M :rx5281 prid=0xdc02
DRAM:  64 MiB @ 1066 MHz
...
[    0.000000] Linux version 4.9.51 (root@a239637c8718) (gcc version 6.4.1 20180425 (Realtek RSDK-6.4.1 Build 3029) ) #1 Tue Mar 16 02
[    0.000000] MIPS: machine is Sheipa Platform
[    0.000000] bootconsole [early0] enabled
[    0.000000] CPU0 revision is: 0000dc02 (Taroko)
[    0.000000] FPU revision is: 01730001
[    0.000000] MIPS: machine is RTS3903N EVB
...

 .oooooo..o                     oooo
d8P'    `Y8                     `888
Y88bo.       .ooooo.   .oooo.    888
 `"Y8888o.  d88' `88b `P  )88b   888
     `"Y88b 888ooo888  .oP"888   888
oo     .d8P 888    .o d8(  888   888
8""88888P'  `Y8bod8P' `Y888""8o o888o

Please press Enter to activate this console. Build Time: Mar 16 2021 06:37:24
```

Thermal Camera

- Module is labeled “Infiray Tiny 1B”
- FW is way more optimized than the i.MX6ULL one
- Realtek RTS3903N SoC support is fully out of tree
  - Seems to be mainly used for IP/WLAN cameras
  - see also https://drmnsamoliu.github.io/hardware.html
- Sensor module is quite fragile
  - I accidently broke one when doing tests with an oscilloscope
- I’m focusing on the i.MX6ULL side at least for now

Thermal Camera: Open Issues

- Figuring out the Vendor Controls
  - I could extract a bunch of them from the UNI-T binary
  - “InfiRay P2 Pro” seems to be similar
    - 0x0bda:0x5830 (P2 Pro) vs 0x0bda:0x3901 (Tiny 1B)
    - There’s a reverse engineered project for that sensor
    - https://github.com/LeoDJ/P2Pro-Viewer/tree/main
    - Unfortunatley protocol is different (USB request 0x44/0x45 vs 0x20/0x19)

Reverse Engineering

```
thermalcam# mount /dev/mmcblk1p2 /mnt
thermalcam# ls /mnt/root
CalTempConfig.ini    UTi160E_config.ini  gpio_keys_test      power_off
DCIM_100             UTi260B_Thermal     led_ctrl_test       usb_charge_status
ImageCal_config.ini  adc_test            live555MediaServer  uvc-gadget
ImageConfig.ini      cam_test            loop.sh             v4l2tester
SystemConfig.ini     gpio_adc_test       play.png
thermalcam# file /mnt/root/UTi260B_Thermal
/mnt/root/UTi260B_Thermal: ELF 32-bit LSB executable, ARM, EABI5 version 1 (SYSV),
dynamically linked, interpreter /lib/ld-linux.so.3, for GNU/Linux 2.6.31, not strippe
```

- not easily possible: run binary from mainline
- binary can be analyzed with radare2 and/or Ghidra
- static strace binary from original system

Upstream Thermal Camera Handling

- Figuring out a good way to handle this in Linux
  - Handle custom vendor control messages via quirk in UVC kernel driver?
    - How to expose the controls?
  - Handle everything in userspace?

Questions?

- Kernel Tree: git.kernel.org: sre/linux-misc.git (branch: uti260b)

Bonus: Flat Connector to Camera Module

- 2x20 flat connector to sensor board, pin 1 is marked, top view

```
GND              1  2           3V3
3V3              3  4           3V3
GND              5  6           5V0
LED_EN           7  8           GND
GC0308.SCL       9 10           GC0308.SDA
GND             11 12           -- LOW --
 -- LOW --      13 14           THERM ~RST
GND             15 16           THERM USB
THERM USB       17 18           GND
 -- LOW --      19 20           GND
GC0308.DATA     21 22           GC0308.DATA
GC0308.DATA     23 24           GC0308.DATA
GC0308.DATA     25 26           GC0308.DATA
GC0308.DATA     27 28           GC0308.DATA
GND             29 30           GC0308.HSYNC (?) (7.5 KHz)
 -- HIGH --     31 32           GND
GC0308.PCLK     33 34           GC0308.INCLK
GND             35 36           GC0308.~RST
GC0308.PWDN     37 38           GND
 -- LOW --      39 40           GND
```

Bonus: Enable USB Gadget Mode

```
#!/bin/sh
mkdir -p /sys/kernel/config/usb_gadget/g1
cd /sys/kernel/config/usb_gadget/g1

echo "0x1d6b" > idVendor   # The Linux Foundation
echo "0x0104" > idProduct  # Multifunction Composite Gadget

mkdir -p strings/0x409     # 0x409 = en-US
echo "0000" > strings/0x409/serialnumber
echo "UNI-T" > strings/0x409/manufacturer
echo "UTi260B" > strings/0x409/product

mkdir -p functions/ecm.usb0

# MAC seen by host system
echo "00:00:00:00:00:42" > functions/ecm.usb0/host_addr

mkdir -p configs/c.1
mkdir -p configs/c.1/strings/0x409
echo "UTi260B ECM" > configs/c.1/strings/0x409/configuration

ln -s functions/ecm.usb0 configs/c.1

echo ci_hdrc.0 > /sys/kernel/config/usb_gadget/g1/UDC
```