MediaTek Upstreaming

From Bring-up to Test Coverage
Angelo Gioacchino Del Regno
Software engineer
Open First
MediaTek, really?! A little background
MediaTek, really?!  

- OpenWRT people started upstreaming Ralink  
  - MediaTek acquired Ralink in 2011  
    - Router SoCs: initially MIPS, then ARM/ARM64  
- A wild laptop appears: Chromebooks!  
  - First MediaTek Chromebook: Acer Chromebook R13 (MT8173), year 2016  
    - Almost fully upstreamed in year 2020  
  - A Current MediaTek Chromebook: Acer Chromebook Spin 513, year 2022  
    - Upstreamed. Currently misses external display support, nothing else!
MediaTek, Yes!

• MediaTek devices are everywhere!
  - You probably have a MediaTek smartphone somewhere
  - Chinese el-cheapo’s: different name, same board

• MediaTek and upstream
  - Relatively small community around it
    • Vast majority on Qualcomm
  - More smartphones, please
    • Mythbusting: MediaTek SoCs are not slow!
Bring-up: Hardware Tools
Chromebooks and Smartphones
Basic tools: power supply

Powering laptops/Chromebooks:

*Embedded Controller (EC) takes care of everything*
Basic tools: serial comms

Chromebooks make it easy (sometimes):

SuzyQable: Type-C → Type-A – triple UART, triple easy
Difficulties buying SuzyQable? No problem!

<table>
<thead>
<tr>
<th>Type C Male</th>
<th>Other host (Type A Male)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A8 (SBU 1)</td>
<td>D+</td>
</tr>
<tr>
<td>B8 (SBU 2)</td>
<td>D-</td>
</tr>
<tr>
<td>A4, A9, B4, B9 (VBUS)</td>
<td>VBUS, 5V</td>
</tr>
<tr>
<td>A5 (CC1)</td>
<td>22 kΩ resistor to VBUS</td>
</tr>
<tr>
<td>B5 (CC2)</td>
<td>56 kΩ resistor to VBUS</td>
</tr>
<tr>
<td>A1, A12, B1, B12 (GND)</td>
<td>GND</td>
</tr>
</tbody>
</table>

Even debugging tools can be open source. Thanks, Google!
Ref.: Chromium - hdctools “Making your own SuzyQ”.

Basic tools: Make your own!
Basic tools: Patience.

Smartphones don’t forgive:

Sony Xperia M5, MediaTek Helio X10 (MT6795) – UART Debug Port
Bring-up: Software Tools
Chromebooks and Smartphones
Chromebook Bring-up Software Tools

• Reference Downstream Kernel Source Code
  - Usually not too old
  - Includes GIT history, you can see what’s going on!
  - Publicly browsable at chromium.googlesource.com

• Recovery Images
  - Chromebook Recovery Utility: Windows and Mac only. (Yes, no Linux support...)
  - Chromium Dash: Manual download, *dd* on your own
  - Installation through USB Mass Storage (pendrives, HDDs, etc.)
Smartphone Bring-up Software Tools

• Reference Downstream Kernel Source Code
  - Usually very old (if you can find it, even)
  - Rarely includes GIT history, many times just a tarball drop
  - Rarely online browsable at <device manufacturer GIT>

• Recovery Images
  - Rarely, manufacturers make recovery tools available; if not...
  - Be prepared to browse potentially dangerous places to find a recovery image...
  - Installation through fastboot flashing
  - Disaster recovery: proprietary and confidential flash tool, not public
Development Tools
Time to put our hands on it
Development tools: Global

• Obvious Basics
  - Your hands
  - Your favorite text editor
  - (cross-)Toolchain
  - Patience, people!

• Scripts and commodities
  - Never underestimate scripting
  - Scripts from the Linux kernel
  - External sources
P.S.: Newer devices need even more parameters...!
Chromebooks: rootfs and kernel
Public and open source

- Everything in one folder, no system-wide install
- Automatic toolchain download and setup
- Manage kernel configuration fragments
- Verify code/bindings, build and flash!

Commands useful for development workflow:

```
deploy_kernel
  Compile the Linux kernel, its modules, the vboot image and deploy all
  on the storage device.
```

For example, to do everything on a SD card (format, repartition, flash the
rootfs, build and flash kernel) for the Acer Chromebook Spin 513 (ARM64):

```
./chromebook-setup.sh do_everything --architecture=arm64 --storage=/dev/sdX
```

or to do the same to use NFS for the root filesystem:

```
./chromebook-setup.sh do_everything --architecture=arm64 --storage=/srv/nfs/nfsroot
```

Available at: https://gitlab.collabora.com/google/chromebooks
Kernel scripts:
checkpatch

Automate it!

echo '#!/bin/bash
exec git diff --cached HEAD | \'

'scripts/checkpatch.pl --no-signoff --no-tree --strict -q -
exit 0' > .git/hooks/pre-commit

kholk@IcarusMOD ~/WORK/COLLABORA/mtk/linux-mtk $ git commit COPYING
WARNING: Possible repeated word: 'Kernel'
#7: FILE: COPYING:1:
+The Linux Kernel Kernel is provided under:

  total: 0 errors, 1 warnings, 0 checks, 5 lines checked

P.S.: Make sure your hooks are executable!
Kernel scripts: bloat-o-meter

Trying to optimize code size?

- Build the kernel
- Save the old `vmlinux` or `*.a` or `*.o`
- Make your changes
- Build again
- bloat-o-meter!

```bash
kholk@IcarusMOD ~/WORK/COLLABORA/mtk/linux-mtk $ scripts/bloat-o-meter ../*linux-mtk-holly/drivers/clk/mediatek/clk-mux.o drivers/clk/mediatek/clk-mux.o add/remove: 10/7 grow/shrink: 2/5 up/down: 458/-495 (-37)
Function                  old     new    delta
mtk_clk_mux_disable_setclr  116     56     -60
mtk_clk_mux_is_enabled      208     128     -80
mtk_clk_mux_set_parent_setclr_lock 428     288    -140
Total: Before=2359, After=2322, chg -1.57%
P.S.: Remember! Smaller is not always better!
```
Static analysis helps a lot!

- Pointer initialization
- Variables usage
- Off-by-one errors, even!
- More life-savers

```c
CHECK drivers/gpu/drm/mediatek/mtk_disp_color.c
CHECK drivers/gpu/drm/mediatek/mtk_disp_gamma.c
drivers/gpu/drm/mediatek/mtk_disp_gamma.c:192 mtk_gamma_set_common() error: we previously assumed 'gamma->data' could be null (see line 120)
CHECK drivers/gpu/drm/mediatek/mtk_disp_merge.c
CHECK drivers/gpu/drm/mediatek/mtk_disp_ovl.c

120 if (gamma && gamma->data) {
121   lut_diff = gamma->data->lut_diff;
192 if (gamma && !gamma->data->has_dither) {
193   /* Descending or Rising LUT */
```

P.S.: Remember: always interpret and evaluate!
Let’s bring it up!
The real deal.
Let’s bring it up: MediaTek SoC

- **Basics…**
  - Serial (UART)
  - Pinctrl
  - Clocks
  - Devicetree

- **…and more**
  - PMIC / Regulators
  - eMMC/SD
  - Display

```
chromium / chromiumos / third_party / kernel / refs/heads/chrromos-5.10 / arch / arm64 / boot / dts / mediatek
```

- 42f6f6c  FROMGIT: arm64: dts: mediatek: mt8195: Add mediatek broken-save-restore-fw to cherry by Douglas Anderson · 2 weeks ago
- b3d9896  FROMGIT: arm64: dts: mediatek: mt8183: Add mediatek broken-save-restore-fw to kukui by Douglas Anderson · 2 weeks ago
- ccc167e  CHROMIUM: Merge "v5.10.173" into chrromos-5.10 by chromeos-kernel-merge · 3 months ago
- d7cf3f6  arm64: dts: mediatek: mt7222: Add missing pwm-cells to pwm node by AngeloGioacchino Del Regno · 3 months ago
- 717aa39  arm64: dts: mediatek: mt8183: Fix systimer 13 MHz clock description by Chen-Yu Tsai · 3 months ago
- 5a3143b  UPSTREAM: arm64: dts: mt8192: Fix CPU map for single-cluster SoC by Linux Patches Robot · 4 months ago
- d7b165a  CHROMIUM: Merge "v5.10.163" into chrromos-5.10 by William K Lin · 5 months ago
Let’s bring it up: UART

Knowing what’s happening

Essential stuff

- Bootloaders usually leave UART pins configured
- Linux can use early console without HW setup
- No need for a full serial driver
- Slow, but effective raw prints

```
AP_UART0@0x11002000 {
    cell-index = <0>;
    compatible = "mediatek,AP_UART0";
    reg = <0x11002000 0x1000>;

    chosen {
        bootargs = "earlycon=mtk8250,0x11002000";
    }
};
```

P.S.: Only for early bring-up!
Let’s bring it up: Pinctrl

Copy-paste or real work?
Hope to find a “modern” driver

- MediaTek kernel <=3.10: pain and suffering
- Modern era: Linux >=3.18: 60 minutes of work
- Essential for SoC integrated devices as well

```c
if (pin == GPIO164) {  //ms8 DS
    return (((GPIO_RD32(reg->msdc0_ctrl4.val) & (iL << 2)) != 0)? 8: 1);
} else if (pin == GPIO165) {  //ms8 RST
    return (((GPIO_RD32(reg->msdc0_ctrl3.val) & (iL << 2)) != 0)? 8: 1);
} else if (pin == GPIO162) {  //ms8 cmd
    return (((GPIO_RD32(reg->msdc0_ctrl1.val) & (iL << 2)) != 0)? 8: 1);
} else if (pin == GPIO163) {  //ms8 clk

static const struct mtk_pinctrl_devdata mt6397_pinctrl_data = {
    .pins = mtk_pins_mt6397,
    .npins = ARRAY_SIZE(mtk_pins_mt6397),
    .dir_offset = (MT6397_PIN_REG_BASE + 0x000),
    .ies_offset = MTK_PINCTRL_NOT_SUPPORT,
    .smt_offset = MTK_PINCTRL_NOT_SUPPORT,
};
```
Copy-paste or real work?
Hope to find a “modern” driver

- MediaTek kernel <=3.10: pain and suffering
- Modern era: Linux >=3.18: 60 minutes of work
- Essential for SoC integrated devices as well

```c
static struct subsys syss[NR_SYSS] = {
    .name = __stringify(SYS_MD1),
    .type = SYS_TYPE_MODEM,
    .default_sta = PWR_DOWN,
    .sta_mask = 1U << 0,
    // .ctl_addr = SPM_MD_PWR_CON,
    .ops = &md1_sys_ops,
},

static const struct mtk_gate mm_clks[] __initconst = {
    /* MM0 */
    GATE_MM0(CLK_MM_SMI_COMMON, "mm_smi_common", "mm_sel", 0),
    GATE_MM0(CLK_MM_SMI_LARB0, "mm_smi_larb0", "mm_sel", 1),
    GATE_MM0(CLK_MM_CAM_MDP, "mm_cam_mdp", "mm_sel", 2),
    GATE_MM0(CLK_MM_MDP_RDMA0, "mm_mdp_rdma0", "mm_sel", 3),
```
Let’s bring it up: Devicetree

Copy-paste or real work?
Hope to find a “modern” driver

- MediaTek kernel <=3.10: pain and suffering
- Modern era: Linux >=3.18: 60 minutes of work
- Essential for SoC integrated devices as well
Copy-paste or real work?
Hope to find a “modern” driver

- MediaTek kernel <=3.10: pain and suffering
  - Definitions scattered all around
- Modern era: Linux >=3.18: not too bad
  - A bit cleaner, more centralized
- Regulators required to get out of early porting
  - Think about eMMC voltage signaling!
- No optimizations yet! Get it working first.
  - Better working than fast....failures!
- It’s never just a copy-paste job anyway.
Test coverage
More important than you think
Test coverage: how it helped me

• Kernel Test Robot (Intel 0-day)
  - Monitoring all (or most of) for-next branches
  - Promptly reporting build issues to you
  - Compiling your ARM64 code for other archs
  - Less work for you

• KernelCI
  - Monitoring Linux stable, linux-next and more
  - Collabora’s MediaTek Integration Kernel
  - Testing on the real hardware!
Test coverage: Intel 0-day

Don’t be scared: it’s here to help you
...you shouldn’t feel annoyed!

- Catches issues before patches reaching -stable
- Goes straight to your mail inbox
- No action required to trigger tests

Image credits: © Intel Corporation, 01.org

More info at: https://01.org/lkp/documentation/0-day-brief-introduction
Test coverage: KernelCI

Catching issues dynamically
Testing on the real hardware

- Static analysis can’t catch all issues
- Tests on multiple devices / SoCs
- Also auto-bisects

Details for «collabora-chromeos-kernel» (for-kernelci)

Showing at most the last 20 results from the available data.

<table>
<thead>
<tr>
<th>Category</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total unique builds</td>
<td>24</td>
</tr>
<tr>
<td>Total defconfigs</td>
<td>24</td>
</tr>
<tr>
<td>Total test results</td>
<td>110,382</td>
</tr>
</tbody>
</table>

Check it out at: https://linux.kernelci.org and https://chromeos.kernelci.org
Test coverage: MediaTek Integration

Collabora’s MediaTek Integration Kernel
Where everything goes together

- Mostly feature complete, always based on -next
- Testing all the pieces together on real hardware
- Our development kernel, KernelCI enabled
- Open to everyone!

Google > ChromeOS Kernel

ChromeOS Kernel
Project ID: 2348

~916,218 Commits 13 Branches 647 Tags 3.1GB Project Storage

Tracks patches on the way to upstream that benefits Chromebook devices

Check it out at: https://gitlab.collabora.com/google/chromeos-kernel
Thank you!
We are hiring
col.la/careers