The complete story of the in-kernel sloppy GPIO logic analyzer

Wolfram Sang, Consultant / Renesas

03.06.2022, KernelRecipes

The elephant in the room

In-kernel sloppy GPIO logic analyzer

- samples data by polling GPIOs
- on an “isolated” CPU core
- does so with irqs + preemption disabled

The elephant in the room

In-kernel sloppy GPIO logic analyzer

- samples data by polling GPIOs
- on an “isolated” CPU core
- does so with irqs + preemption disabled

Why is it called “sloppy”?

The elephant in the room

In-kernel sloppy GPIO logic analyzer

- samples data by polling GPIOs
- on an “isolated” CPU core
- does so with irqs + preemption disabled

Why is it called “sloppy”?

- Not really a question, or??

Questions?

Wait until the end, please[1]

[1] It's a bit of a ride...

The Task

enable IP cores on new board

- cores were known
- add dt-bindings
- enable clocks
- set pinmuxing

We only sent tested patches upstream

The Problem

Wolfram at home (.de)

[2] Easy for I2C, MMC, ...; not so easy for PWM, ...

The Problem

Wolfram at home (.de)

board in lab (.jp)[2]

[2] Easy for I2C, MMC, ...; not so easy for PWM, ...

The idea

What did I not have

- multiple logic analyzers in the lab
- a person constantly in the lab

The idea

What did I not have

- multiple logic analyzers in the lab
- a person constantly in the lab

What did I have

- wires to be set up once
- lots of idle CPU cores

Was I an expert in CPU isolation?

[3] Really high risk of brown paper bag situation here

Was I an expert in CPU isolation?

No[3]

[3] Really high risk of brown paper bag situation here

Did I develop a thick skin after all these years?

Did I develop a thick skin after all these years?

Maybe

Do I want to share what I developed?

Do I want to share what I developed?

YES!

SGLA: the kernel part

relatively easy

- config files in debugfs
  - out: freq, size, triggers, start
  - in: data, meta data, timing info
- when run:
  - lock CPU
  - wait for triggers
  - sample
  - unlock CPU

SGLA: the script

relatively complex

- provide easy syntax for the debugfs files
- “isolate” CPU
- run polling task on that CPU
- convert data to sigrok format

Should run everywhere, so any shell

script: avoid bashism

- only ash, zip, and taskset needed
- verified by check_bashisms

script: avoid bashism

- only ash, zip, and taskset needed
- verified by check_bashisms
- verified by shellcheck

script: avoid bashism

- only ash, zip, and taskset needed
- verified by check_bashisms
- verified by shellcheck
- verified by Andy Shevchenko

script: isolate CPU

I thought there was a helper for all this?[4]

- set up a cpuset
- set smp_affinity for all irqs
- set cpumask for all workqueues
- use taskset to move tasks away from “isolated” CPU
- tell RCU that stalled CPUs are okay
- (set cpufreq governor to performance)

[4] `isolcpus` is nice but deprecated

DT example

```
i2c2-analyzer {
        compatible = "gpio-sloppy-logic-analyzer";

        probe-gpios = <&gpio6 21 GPIO_OPEN_DRAIN>,
                      <&gpio6 4 GPIO_OPEN_DRAIN>;

        probe-names = "SCL", "SDA";
};
```

Live demo #1

Snoop I2C traffic on a local machine

Result from Live Demo #1

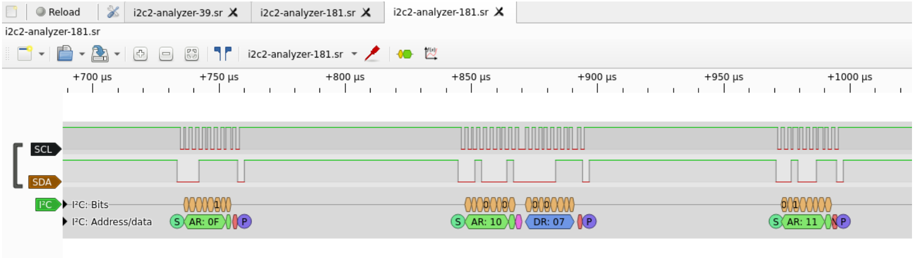


Figure 1: Not so live

Commercial break: sigrok

Great project...

Commercial break: sigrok

Great project...
having serious man-power
problems :(

Live demo #2

Snoop PWM on a remote machine

Murphy’s law, take #1

```
# gpio-sloppy-logic-analyzer -s 1500000 \\
                             -d 12000 \\
                             -t "1H+2H,1H+2F"
Could not isolate CPU1. Does it exist?
```

Murphy's law, take #1

```
# gpio-sloppy-logic-analyzer -s 1500000 \\
                             -d 12000 \\
                             -t "1H+2H,1H+2F"
Could not isolate CPU1. Does it exist?
```

Oh, no SMP yet?

Murphy’s law, take #1

```
# gpio-sloppy-logic-analyzer -s 1500000 \\
                             -d 12000 \\
                             -t "1H+2H,1H+2F"
Could not isolate CPU1. Does it exist?
```

Oh, no SMP yet?

What to do?

Result from Live Demo #2

It worked, really, trust me!

Let's skip the wiring!

Ulrich's idea

- he read the GPIOIN reg and could skip the wiring

Let's skip the wiring!

Ulrich's idea

- he read the GPIOIN reg and could skip the wiring
- I hacked the GPIO subsystem to reuse GPIOs

Let's skip the wiring!

Ulrich's idea

- he read the GPIOIN reg and could skip the wiring
- I hacked the GPIO subsystem to reuse GPIOs
- Linus mentioned that this was a known and supported feature in the GPIO subsystem called "non-strict"

Let's skip the wiring!

Ulrich's idea

- he read the GPIOIN reg and could skip the wiring
- I hacked the GPIO subsystem to reuse GPIOs
- Linus mentioned that this was a known and supported feature in the GPIO subsystem called “non-strict”
- we can measure pins which are not even exposed on the board \o/

Let’s skip the wiring!

Ulrich’s idea

- he read the GPIOIN reg and could skip the wiring
- I hacked the GPIO subsystem to reuse GPIOs
- Linus mentioned that this was a known and supported feature in the GPIO subsystem called “non-strict”
- we can measure pins which are not even exposed on the board \o/

Our HW supports that, but not our pinctrl driver :(

One line fix to the rescue!

```
-        if (!pfc->gpio) {
+        if (!pfc->gpio && !cfg->mux_mark) {
```

DT example 2

```
i2c2-analyzer {
        compatible = "gpio-sloppy-logic-analyzer";

        probe-gpios = <&gpio5 4 0>,
                      <&gpio5 0 0>;

        probe-names = "SCL", "SDA";
};
```

Live demo #3

Snoop I2C traffic on a local machine without wires

Result from Live Demo #3

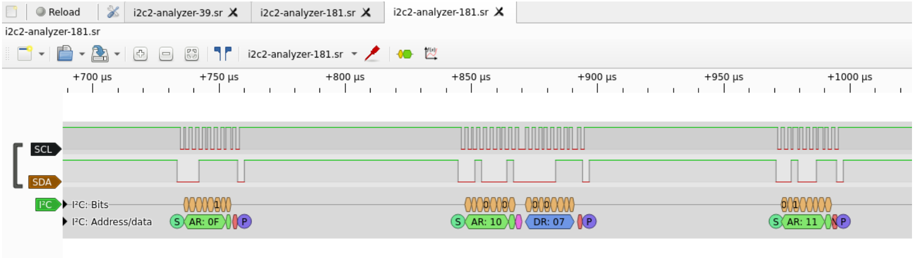


Figure 2: Not so live

Live demo #4

Snoop PWM traffic on a remote machine without wires

Result from Live Demo #4

Result from Live Demo #4

NULL

Murphy's law, take #2

HW no longer non-strict with GPIOs[5]

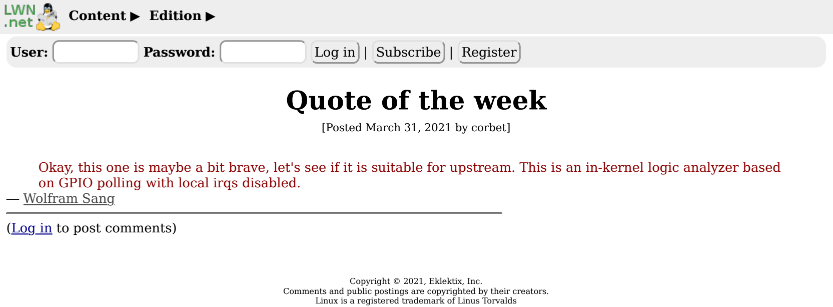
[5] most likely

current status

- works reasonably well for my mostly idle targets
- sadly our HW lost the non-strict GPIOs
- was already useful when debugging other issues
- we can go even more wild
- lots of fun while creating

  (unless other people reported back ;))
- still not upstream
- still not a logic analyzer

But I got a “Quote of the Week” \o/



Quote of the week

[Posted March 31, 2021 by corbet]

> Okay, this one is maybe a bit brave, let's see if it is suitable for upstream. This is an in-kernel logic analyzer based on GPIO polling with local irqs disabled.

— Wolfram Sang

(Log in to post comments)

The End

Questions? Comments?

(except from Marek ;))

Questions?

- Right here, right now...
- At the conference
- wsa@kernel.org

And thanks go to Renesas for partly funding this work!