Developer's Diary: The Device Tree
(Experiences from the last 2 years)

Wolfram Sang

16.10.2009, ELC Europe 2009

Overview

Overview

Prerequisites

The device tree and me

- encountered device tree and platform_data at the same time
- like everything, there are things I like and dislike
- overall, mostly neutral: It's there, make the best out of it.

This talk and you

- not an introduction to the topic
- no pleading for or against device tree
- result from practical experience
- sum up core problems I see, so we can tackle them

Reminders

About the device tree

- just a hardware description (no driver specific details!)
- hardware independent
- OS independent (no linux specific details!)(!!!)

About properties

- is simply a key-value pair
- can easily be defined and parsed
- needs to be discussed (devicetree-discuss@lists.ozlabs.org)
- *compatible* property should be specific and cover most details

Overview

AT24: Configuration

The generic eeprom driver in the Linux Kernel.

Excerpt from platform_data:

- page_size (max bytes per write)
- ...
- AT24_IRUGO (make data world readable)
- AT24_READ_ONLY (make data read-only)
- ...

Mappable to properties (OS independent)?

AT24: Configuration

The generic eeprom driver in the Linux Kernel.

Excerpt from platform_data:

- page_size (max bytes per write)
- ...
- AT24_IRUGO (make data world readable)
- AT24_READ_ONLY (make data read-only)
- ...

Mappable to properties (OS independent)?

Problem:

Creating proper properties is not trivial (docs would help)

AT24: How to implement?

Task: get data into a generic (= non of) driver

First try (Mid 2008):

- of_i2c does the general device-driver matching
- no hooks available

Second try (Mid 2009):

- A. Vorontsov introduced dev_archdata meanwhile
- query properties in at24_probe (using #ifdef CONFIG_OF)
- not favoured upstream

Third try (somewhere in the future):

- separate call creating platform_data before probe?

AT24: How to implement?

Task: get data into a generic (= non of) driver

First try (Mid 2008):

- of_i2c does the general device-driver matching
- no hooks available

Second try (Mid 2009):

- A. Vorontsov introduced dev_archdata meanwhile
- query properties in at24_probe (using #ifdef CONFIG_OF)
- not favoured upstream

Third try (somewhere in the future):

- separate call creating platform_data before probe?

Problem:

Some generic functionality still missing

PCA953X: Properties through the backdoor

Very similar problem was addressed, just...(no offence)

- no discussion about the properties (devicetree-discuss)
- picked up by Andrew Morton
- *linux,base*?
- *polarity*?
- need to be replaced; might create potential confusion

PCA953X: Properties through the backdoor

Very similar problem was addressed, just...(no offence)

- no discussion about the properties (devicetree-discuss)
- picked up by Andrew Morton
- *linux,base*?
- *polarity*?
- need to be replaced; might create potential confusion

Problem:

Creating proper properties is not trivial (docs would help)
We need more awareness about device tree matters

Overview

Wrapping uio_pdrv_genirq I

Task: provide an of-version of the driver

First try (Late 2008):

- put everything into one source file as proposed on linuxppc-dev and seen in xilinxfb.c
- parse properties and create uioinfo-structure directly
- rejected on lkml („never combine such stuff into one source file!“)

Second try (Mid 2009):

- split of-specific parts into separate file
- parse properties and fill platform_data (which will become uioinfo)
- better, but using „uio-generic“ is a Linux-specific value for compatible

Wrapping uio_pdrv_genirq II

Task: provide an of-version of the driver

Third try (somewhere in the future):

- don't use generic binding
- don't add every possible user to the static compatible list
- add mechanism to add and force bindings at runtime
- adapt and resubmit second try

Wrapping uio_pdrv_genirq II

Task: provide an of-version of the driver

Third try (somewhere in the future):

- don't use generic binding
- don't add every possible user to the static compatible list
- add mechanism to add and force bindings at runtime
- adapt and resubmit second try

Problem:

Docs would help (bindings, preferred way to adapt drivers)
Device tree is specific, some drivers are generic
Some generic functionality still missing

Overview

Generic watchdog using GPIO I

Device tree can bind to GPIOs quite elegant.

Task: Ask how to do it

First try, just RFC (Mid 2008):

```
watchdog@gpio {
        compatible = "gpio-watchdog";
        gpios = <&gpio_simple 19 0>;
};
```

- naive „platform_data“-oriented approach
- rightfully rejected
- such a driver would be useful for platform_data, too
  same problem as AT24

Generic watchdog using GPIO II

Second try (somewhere in the future):

Ask again how to do it

```
gpio@c000 { ...
        gpio-controller
        watchdog@5 {
                compatible = "ptx,super-sexy-board"
                gpio = <5 0>;
        }; ...
```

- just a sketch, surely needs further discussion...

Generic watchdog using GPIO II

Second try (somewhere in the future):

Ask again how to do it

```
gpio@c000 { ...
        gpio-controller
        watchdog@5 {
                compatible = "ptx,super-sexy-board"
                gpio = <5 0>;
        }; ...
```

- just a sketch, surely needs further discussion...

Problem:

Creating proper properties is not trivial (docs would help)
Device tree is specific, some drivers are generic
Some generic functionality still missing

Overview

Core problems I

We need more documentation

- How to create properties (to prevent a mess)
- How to adapt drivers
- ...

Core problems I

We need more documentation

- How to create properties (to prevent a mess)
- How to adapt drivers
- ...

We need more awareness

- (drop the hate, be technical)
- What are device tree matters
- Who to contact

Core problems II

Device tree is specific, some driver are very generic

- don't use linux specific generic property
- don't bloat the static compatible table

Core problems II

Device tree is specific, some driver are very generic

- don't use linux specific generic property
- don't bloat the static compatible table

We still need some generic functionality

- mechanism to force a binding at runtime
- mechanism to fill platform_data before calling probe
- ...

Core problems II

Device tree is specific, some driver are very generic

- don't use linux specific generic property
- don't bloat the static compatible table

We still need some generic functionality

- mechanism to force a binding at runtime
- mechanism to fill platform_data before calling probe
- ...

You might say: „Stop whining, send patches!“

Summary

Well...

Help!

Summary

Well...

Help!

More manpower is needed; it would help to

- get active; don't just wait for a solution
- collaborate; don't hack around (especially with properties)
- support sustainable solutions (be it with time or money)
  These could be useful for other descriptions, too (EFI?)

The End

Thank you for your attention!

Questions?