LAS16-TR03: Upstreaming 201

Daniel Thompson

ENGINEERS
AND DEVICES
WORKING
TOGETHER

Overview

- Building on Upstreaming 101
  - Treat this session like a workshop
  - Ask questions immediately
- Example driven
  - Subjects/Commit Messages
  - Responding to Comments
  - Upstreaming a new driver
- Review some potentially useful tools and techniques
  - Mailing lists and IRC
  - Source navigation
  - Static checkers
  - Handling regressions
- Free bonus extra (unlikely to be covered today)
  - Upstreaming a new architecture

SubmittingPatches - Subjects

For these reasons, **the "summary" must be no more than 70-75 characters, and it must describe both what the patch changes, as well as why the patch might be necessary.** It is challenging to be both succinct and descriptive, but that is what a well-written summary should do.

The "summary phrase" may be prefixed by tags enclosed in square brackets: **"Subject: [PATCH tag] <summary phrase>".** The tags are not considered part of the summary phrase, but describe how the patch should be treated. **Common tags might include a version descriptor if the multiple versions of the patch have been sent out in response to comments (i.e., "v1, v2, v3"), or "RFC" to indicate a request for comments. If there are four patches in a patch series the individual patches may be numbered like this: 1/4, 2/4, 3/4, 4/4.** This assures that developers understand the order in which the patches should be applied and that they have reviewed or applied all of the patches in the patch series.

SubmittingPatches - Commits

The explanation body will be committed to the permanent source changelog, so should make sense to a competent reader who has long since forgotten the immediate details of the discussion that might have led to this patch. Including symptoms of the failure which the patch addresses (kernel log messages, oops messages, etc.) is especially useful for people who might be searching the commit logs looking for the applicable patch. If a patch fixes a compile failure, it may not be necessary to include _all_ of the compile failures; just enough that it is likely that someone searching for the patch can find it. **As in the "summary phrase", it is important to be both succinct as well as descriptive.**

Commit Messages - Advice on style

```
Subject: Re: [PATCH 2/3] panic: improve panic_timeout calculation
From: Ingo Molnar <mingo@xxxxxxxxxx>
Date: Mon, 11 Nov 2013 12:32:18 +0100

Felipe Contreras <felipe.contreras@xxxxxxxxx> wrote:
> We want to calculate the blinks per second, and instead of making it 5
> (1000 / (3600 / 18)), let's make it 4, so the user can see two blinks
> per second.

Please use the customary changelog style we use in the kernel:

  " Current code does (A), this has a problem when (B).
    We can improve this doing (C), because (D)."
```

Summary/Commit Messages - Good

Subject: **[PATCH] ARM: dts: imx6qdl-sabresd: SDHC ports are 8 bit-wide**
From: Fabio Estevam fabio.estevam at freescale.com
Date: Tue Sep 17 12:46:23 EDT 2013

On imx6qdl-sabresd the SDHC2 and SDHC3 are 8 bit-wide, so pass the bus-width property to reflect that.

Otherwise the mmc driver will operate with the default bus-width value of 4.

Signed-off-by: Fabio Estevam <fabio.estevam at freescale.com>
---
 arch/arm/boot/dts/imx6qdl-sabresd.dtsi | 2 ++
 1 file changed, 2 insertions(+)

Don’t ignore comments

```
From: Mark Brown <>
Subject: Re: [PATCH v2 3/8] regulator: MT6397: Add support for MT6397 regulator
Date: Fri, 28 Nov 2014 15:22:47 +0000
…
> +  np = of_node_get(pdev->dev.parent->of_node);
> +  if (!np)
> +       return -EINVAL;
> +
> +  regulators = of_get_child_by_name(np, "regulators");
```

To further repeat my previous review comments:

| Define regulators_node and of_match in the regulator desc and you can
| remove both this table and all your DT matching code in the driver, the
| core will handle it for you.

Please don't ignore review comments.

Clarifying comments

Subject: Re: [PATCH v9 2/4] Documentation: Add documentation for APM X-Gene SoC SATA host controller DTS binding
From: Loc Ho <lho@xxxxxxx>
Date: Wed, 15 Jan 2014 12:04:02 -0800

>> +- clocks : Reference to the clock entry.
>> +- phys : PHY reference with parameter 0.
>
> The specific value of the phy-specifier shouldn't matter to this
> binding. What should matter is what it logically corresponds to.

I not quite following this. Are you suggest that I drop the value 0. In the binding, one needs to specify the mode of operation - 0 is for SATA. Can you explain more?

-Loc

Incorporating comments

```
Subject: Re: [PATCH 4/5] ARM: tegra: Add host1x, dc and hdmi to Tegra114
device tree
From: Mikko Perttunen
Date: Wed, 28 Aug 2013 07:54:47 -0700

On 08/28/2013 03:25 PM, Thierry Reding wrote:
> * PGP Signed by an unknown key
>
> On Wed, Aug 28, 2013 at 01:40:58PM +0300, Mikko Perttunen wrote:
>> Add host1x, dc (display controller) and hdmi devices to Tegra114
>> device tree.
>
> "DC" and "HDMI".
```

Will fix.

Responding to comments - Don’t do this

Subject Re: [Arm-netbook] getting allwinner SoC support upstream (was Re: Uploading linux (3.9.4-1))
From"luke.leighton" <>

On Thu, Jun 6, 2013 at 1:01 AM, Tomasz Figa <tomasz.figa@gmail.com> wrote:

> I don't see any other solution here than moving all the Allwinner code to
> DT (as it has been suggested in this thread several times already), as
> this is the only hardware description method supported by ARM Linux.

i repeat again: please state, explicitly and unequivocably **that you - linux kernel developers - are happy that the reach of linux and gnu/linux OSes is dramatically reduced due to this intransigent position.**

or, tomasz, please state that you, tomasz, represent each and every one of the linux kernel developers so that i do not need to keep asking.

Upstreaming a driver - watchdog

- Logically split the work
  - Device Tree binding (DT team / Wim Van Sebroeck)
  - Any watchdog framework changes (Wim Van Sebroeck)
  - Driver and build plumbing (Wim Van Sebroeck)
  - defconfig update, if applicable (mach-* maintainer or arm-soc team)
  - DTS additions to enable driver (mach-* maintainer)
- Split to divide amongst maintainer trees
- Split to divide framework updates, specific driver features, and/or individual bug fixes

Creating a driver series - commits

```
$ git log --oneline
1234567 watchdog: bcm281xx: Watchdog Driver
7654321 ARM: bcm281xx: watchdog configuration
```

[If we had a DT binding and associated DTS for this series]

```
$ git log --oneline
0000000 watchdog: bcm281xx: add DT binding
1234567 watchdog: bcm281xx: Watchdog Driver
7654321 ARM: bcm281xx: watchdog configuration
8888888 ARM: dts: bcm281xx: add watchdog DT support
```

Posting a driver series - create patches

What we want

```
Subject: [PATCH 0/2] watchdog: bcm281xx: Watchdog Driver
Subject: [PATCH 1/2] watchdog: bcm281xx: Watchdog Driver
Subject: [PATCH 2/2] ARM: bcm281xx: watchdog configuration
```

Command

```
$ git format-patch --cover-letter -o /tmp/bcm281xx_wd \
    --to … --to … --cc … --cc … --cc … v4.3-rc1..
```

Results

```
$ ls /tmp/bcm281xx_wd
0000-cover-letter.patch
0001-watchdog-bcm281xx-watchdog-driver.patch
0002-ARM-bcm281xx-watchdog-configuration.patch
```

Posting a driver series - cover letter

```
From: Markus Mayer <markus.mayer@linaro.org>
Subject: [PATCH 0/2] watchdog: bcm281xx: Watchdog Driver

This series introduces the watchdog driver for the BCM281xx family of
mobile SoCs.

Markus Mayer (2):
  watchdog: bcm281xx: Watchdog Driver
  ARM: bcm281xx: watchdog configuration

 arch/arm/configs/bcm_defconfig  |    3 +
 drivers/watchdog/Kconfig        |   21 +++
 drivers/watchdog/Makefile       |    1 +
 drivers/watchdog/bcm_kona_wdt.c |  399
++++++++++++++++++++++++++++++++++++++++
 4 files changed, 424 insertions(+)
 create mode 100644 drivers/watchdog/bcm_kona_wdt.c
```

Posting a driver series - watchdog driver

```
From: Markus Mayer <markus.mayer@linaro.org>
Subject: [PATCH 1/2] watchdog: bcm281xx: Watchdog Driver

This commit adds support for the watchdog timer used on the BCM281xx
family of SoCs.

Signed-off-by: Markus Mayer <markus.mayer@linaro.org>
Reviewed-by: Matt Porter <matt.porter@linaro.org>
---
 drivers/watchdog/Kconfig         |   21 +++
 drivers/watchdog/Makefile        |    1 +
 drivers/watchdog/bcm_kona_wdt.c  |  399
++++++++++++++++++++++++++++++++++++++++
 3 files changed, 421 insertions(+)
 create mode 100644 drivers/watchdog/bcm_kona_wdt.c
diff --git a/drivers/watchdog/Kconfig b/drivers/watchdog/Kconfig
index d1d53f3..59013f6 100644
--- a/drivers/watchdog/Kconfig
+++ b/drivers/watchdog/Kconfig
...
```

Posting a driver - Reposting

From: Markus Mayer <markus.mayer@linaro.org>
Subject: [PATCH v2 0/2] watchdog: bcm281xx: Watchdog Driver

This is version 2 of the watchdog driver for the BCM281xx family of mobile SoCs.

Changes since version 1:

- Added module name to "help" section in Kconfig
- A few cosmetic code simplifications and fixes
- Removed most dev_info() calls and changed the remaining ones to dev_dbg()
- Renamed SECWDOG_WD_LOAD_FLAG_MASK to SECWDOG_WD_LOAD_FLAG
- Added some comments to secure_register_read() and struct bcm_kona_wdt
- Added delay to secure_register_read()
- Reduced maximum retry loop from 10000 to 1000
- Introduced "busy_count" variable to count how often secure_register_read()
** gets stalled; this is available through debugfs**
- Simplified secure_register_read() to return -ETIMEDOUT rather than using
** a variable parameter to indicate a timeout error**

...

Posting a driver - Comments

```
From: Markus Mayer <markus.mayer@linaro.org>
Subject: Re: [PATCH 1/2] watchdog: bcm281xx: Watchdog Driver

On 12 November 2013 15:39, One Thousand Gnomes <gnomes@lxorguk.ukuu.org.uk> wrote:
...
>> +     val = secure_register_read(wdt->base + SECWDOG_CTRL_REG, &timeout);
>> +     if (!timeout) {
>> +             val &= ~SECWDOG_RES_MASK;
>> +             val |= wdt->resolution << SECWDOG_CLKS_SHIFT;
>> +             writel_relaxed(val, wdt->base + SECWDOG_CTRL_REG);
>> +     } else {
>> +             ret = -EAGAIN;
>
```

> This is I think the wrong choice of return. If the register fails to read
> then presumably the device is b*ggered ? In which case return something
> like -EIO and log something nasty.
>
> EAGAIN has fairly specific semantics around signals and/or specific
> requests for an I/O operation not to wait.

I will change that based on Guenter's and your comments.

Posting a driver series - acceptance

From: Wim Van Sebroeck <>
Subject: Re: [PATCH v5 0/2] watchdog: bcm281xx: Watchdog Driver

Hi Markus,

> This is version 5 of the watchdog driver for the BCM281xx family of
> mobile SoCs.

I applied the patches but without the BCM_KONA_WDT_DEBUG part, because I still have some questions about it. So what is left is the following patch:

...

Posting a driver: get_maintainer.pl

```
$ scripts/get_maintainer.pl -f drivers/mmc/host/sdhci-foo.c
Ulf Hansson <ulf.hansson@linaro.org> (maintainer:MULTIMEDIA
 CARD (MMC), SECURE DIGITAL (SD) AND...)
"GitAuthor: Daniel Thompson" <daniel.thompson@linaro.org>
 (authored:1/1=100%,added_lines:1/1=100%)
linux-mmc@vger.kernel.org (open list:MULTIMEDIA CARD (MMC),
 SECURE DIGITAL (SD) AND...)
linux-kernel@vger.kernel.org (open list)
```

Posting a driver: get_maintainer.pl

```
$ scripts/get_maintainer.pl 0001-mmc-sdhci-Dummy-driver-for-foo.patch
Ulf Hansson <ulf.hansson@linaro.org> (maintainer:MULTIMEDIA CARD (MMC),
 SECURE DIGITAL (SD) AND...,commit_signer:18/19=95%,commit_signer:7/8=88%)
Jean Delvare <jdelvare@suse.de> (commit_signer:4/19=21%,authored:4/19=21%)
Alim Akhtar <alim.akhtar@samsung.com> (commit_signer:3/19=16%,removed_lines:1/15=7%)
Andrew Bresticker <abrestic@chromium.org> (commit_signer:2/19=11%,authored:2/19=11%,removed_lines:2/15=13%)
Stephen Boyd <sboyd@codeaurora.org>
 (commit_signer:2/19=11%,removed_line:8/15=53%,commit_signer:2/8=25%,removed_lines:1/1=100%)
Ondrej Zary <linux@rainbow-software.org> (authored:1/19=5%,authored:1/8=12%,added_lines:1/7=14%)
Chaotian Jing <chaotian.jing@mediatek.com> (authored:1/19=5%,authored:1/8=12%,added_lines:1/7=14%)
Scott Branden <sbranden@broadcom.com> (authored:1/19=5%,added_lines:14/81=17%,authored:1/8=12%,added_lines:1/7=14%)
Vincent Yang <vincent.yang.fujitsu@gmail.com> (added_lines:11/81=14%,commit_signer:1/8=12%)
Srinivas Kandagatla <srinivas.kandagatla@linaro.org> (added_lines:11/81=14%,authored:1/8=12%,added_lines:1/7=14%)
addy ke <addy.ke@rock-chips.com> (added_lines:9/81=11%,commit_signer:1/8=12%)
Daniel Thompson <daniel.thompson@linaro.org> (added_lines:9/81=11%,removed_lines:1/15=7%,authored:1/8=12%,added_lines:1/7=14%,authored:1/1=100%,added_lines:1/1=100%)
Kevin Hao <haokexin@gmail.com> (removed_lines:2/15=13%)
Ray Jui <rjui@broadcom.com> (commit_signer:1/8=12%)
linux-kernel@vger.kernel.org (open list)
linux-mmc@vger.kernel.org (open list:MULTIMEDIA CARD (MMC), SECURE DIGITAL (SD) AND...)
```

ENGINEERS
AND DEVICES
WORKING
TOGETHER

Sharp tools and smart techniques

- Filtering mailing lists
- Source navigation
- Static checkers
- Handling regressions
- Bisectability testing

Sharp tools and smart techniques - Overview

Nothing in this section is mandatory

Maintainers do not care whether you use special tools… they only care that your patches meet the kernel's standards

So here we will introduce tools and techniques that could help you work better with the community and meet the upstream standards

It's up to you to decide which works for **you**

Filtering mailing lists

Why?

It is impossible to read **everything** posted to LKML - “like drinking from a fire hose”...

... but if you don’t read **anything** on the mailing lists you will never absorb enough kernel culture

How?

Subscribe anyway and employ **really** aggressive filters to focus on things that interest you

Filtering mailing lists - Good habits

- Have a whitelist - delete **everything** else
  - Don't hoard old messages... search engines can find anything you miss
- Create a filter to keep an eye on code you have written
  - It's good for your reputation to keep an eye on things you write
  - You can use a highly focused filter here... only care about your driver/code
- Create a filter to keep an eye on what you are working on now
  - Review comments from similar drivers will now cross your desk automatically
  - Use `git log drivers/of/interest.c` to get some idea of useful subject lines to filter
  - Create a generic filter here... you want to see what other people are doing right/wrong
- Still fairly high volume
  - Keep the mail part of your normal mailbox flow
  - Colourize/mark mail from the list so your normal mail stands out compared to ML traffic
  - Make sure **only** ML traffic is tagged (i.e. if you **and** ML are in To/Cc: do **not** tag like ML)
  - For tagged mail, press Delete straight away if the subject line doesn't interest you

Filtering mailing lists - Special gmail account

- LKML and most other Linux lists do **not** require you to be a member in order to post them
  - You don't need to send and receive messages from same e-mail account
- Create a special purpose gmail account to pre-filter mail
  - Register your work e-mail as a forwarding address in gmail
  - Write filters to forward targeted messages to you work e-mail
  - After forwarding, filter everything else to the Trash (where it will be deleted after 30 days)
- Advantages
  - Easy to colourize ML messages: if "X-Forwarded-For" does not contain "mygmailname"
  - Never conflict with messages which include you in To:/Cc: (should be more obvious that ML)
  - Even with auto-delete set you can still search the last 30 days of traffic to quickly dig out and save patchsets. Easier than copy 'n paste from archival web sites
- Disadvantages
  - Initial setup is more complex

Socialising on IRC

IRC channels on the freenode network is a good place to get informal contact with developers and maintainers (during periods when timezones overlap).

Start with `#armlinux` .

Finding channels of interest can be hard (IRC is not listed in `MAINTAINERS`). Try:

```
/msg alis list #linux-m*
```

It's OK to sit and "watch" until you feel confident with the culture.

Don't ask permission to ask a question. If you have a question, just ask it.

Great place to follow up on code reviews. For example: If reviewers idea won't work but you can think of something similar and want to discuss that.

Source navigation

Why?

If you can browse the kernel source **quickly** then you can participate more effectively in e-mail and IRC discussion

Maintainers sometimes ask “are you sure this is safe when ...?”. That makes it your job to find out. **Use the source**.

How?

Be sure to have one (or more) indexing tools integrated into your workflow

Source navigation - tags/cscope/gtags

- Problem: Indexing only the current architecture
  - The auto-discovery tools of your normal indexer will end up indexing every architecture
  - Great to check for portability problems but the rest of the time it just slows you down
- Solution: Use a supported indexer (uses ARCH to restrict code indexed)
  - ctags: make tags
  - cscope: make cscope
  - GNU GLOBAL: make gtags
- You can use more than one indexer!
  - Use one that is well integrated into your editor but maybe you also need more power...
  - Search for Calling/Called by is good to check the consequences of the “tiny” change you need to make to a framework to better support your driver
- Can’t decide? You want a sane default with trivial editor integration? Try:

  ARCH=arm64 `make cscope; cbrowser`

Source navigation - On the web

- Great for looking at really big picture
  - Many web indexers allow you to step back/forward in time (check 3.10 versus mainline)
  - Links are durable and can be shared with others via IRC and in email
  - Great if you are away from your workstation and working on a weak machine (or chromebook)
- LXR
  - Linux Cross Referencer
  - Based on (exuberant) ctags
  - No canonical server but http://lxr.free-electrons.com/ is popular and well maintained

Source navigation - Finding structure definitions

From `Documentation/CodingStyle`:

> It's a **mistake** to use typedef for structures and pointers.

Some source navigators work really badly for this coding style making it hard to find structures definitions when headers use incomplete types:

```
/* no definition of struct virtual_container seen so far
 */
void do_vc_func(struct virtual_container *p)
```

Solution... fall back to grep:

```
git grep "struct virtual_container {"
```

Static checkers

Why?

Mostly it's about catching issues early. The fewer issues that come up in code review that faster you can deliver code upstream.

Maintainers will sometimes tell you that "your patch introduces a sparse warning, please fix it". It's important to know how to run the same checks maintainers do.

How?

Need to find ways to integrate static checkers into your workflow. This is a very personal decision so today we'll just look at the tools.

Static checkers - General

Kernel build system can automatically run `$(CHECK) $(CFLAGS) $<`

- `make C=1` calls `$(CHECK)` before each file is compiled
- `make C=2` calls `$(CHECK)` for all files unconditionally
- By default `CHECK=sparse`

Note: CFLAGS contains lots of GCC specific arguments

```
sparse -D__linux__ … -Wbitwise -Wno-return-void … -nostdinc -isystem
…/lib/gcc/arm-linux-gnueabihf/4.8.3/include
-Wp,-MD,arch/arm/mm/.cache-l2x0.o.d  -nostdinc … -I./arch/arm/include …
-mlittle-endian -Wall … -fno-strict-aliasing -fno-common … -std=gnu89
-fno-dwarf2-cfi-asm -mabi=aapcs-linux -mno-thumb-interwork  -Uarm -O2
--param=allow-store-data-races=0 … arch/arm/mm/cache-l2x0.c
```

Static checkers - sparse

- Semantic PARSEr - generates a parse tree and runs checkers over the results
  - https://www.kernel.org/doc/Documentation/sparse.txt
- Primary checking feature is the enriched type system
  - Track user pointers vs. mmio pointers vs. kernel pointers
  - Warn when per-cpu pointers are directly dereferenced
  - Mixing bitwise types with other types
  - Lock acquire/release/hold

```
warning: incorrect type in argument 1 (different address spaces)
warning: cast removes address space of expression
warning: implicit cast to nocast type
error: symbol 'do_exit' redeclared with different type - different modifiers
warning: context imbalance in '__lock_task_sighand' - different lock contexts
for basic block
```

Static checkers - Further reading

`make help` - Describes some of the available static checks

`checkstack`	Generate a list of stack hogs
`coccicheck`	Semantic (pattern matching) patches
`namespacecheck`	Name space analysis on compiled kernel
`versioncheck`	Sanity check on version.h usage
`includecheck`	Check for duplicate included header files
`export_report`	List the usages of all exported symbols
`headers_check`	Sanity check on exported headers
`headerdep`	Detect inclusion cycles in headers

`make W=1` (or `W=123` if you **like** false positives)

`make C=1 CHECK="smatch -p=kernel"`

Static checkers - Bringing it all together

- False positives are a big problem for static checkers
- aiaiai
  - Builds multiple kernel configurations without and without your patches
  - Integrates many static checkers and will reports **changes** in compiler warnings and static checker output for each patch
  - http://git.infradead.org/users/dedekind/aiaiai.git

This runs with extra compiler warnings and lots of static checkers. Quick checks could just run with fewer checkers (maybe just W=1 and sparse)

```
BASELINE=v4.8-rc2; KERNTEST=v4.8-rc5; \
git format-patch --stdout $BASELINE | \
aiaiai-test-patchset -v -j 4 -c $KERNTEST \
-M W=1 --sparse --cppcheck --coccinelle --smatch \
$PWD defconfig,x86 defconfig,arm64,aarch64-linux-gnu- \
multi_v7_defconfig,arm,arm-linux-gnueabihf-
```

Handling regressions

Automate everything...

... so when you find a regression ...

... you can automatically find how it was introduced

Maintainers try to ensure **every** commit is compilable (and mostly they succeed). This makes it possible to perform binary search for the commit that introduced a regression.

`git bisect` can automate this search and uses an algorithm that can cope with any compilation failures the maintainers missed

An aside: Automate everything

- It's **your** platform… in the early days no-one will be more motivated than you to keep your platform running
- **Automatic testing** of boot and basic kernel features is valuable
  - Many options for this... both public labs and internal labs... jenkins/LAVA/...
  - Cannot be tackled in depth during this presentation (ask your TLE to discuss options offered by Linaro)
- Some characteristics of good automation
  - Accessible: can be deployed, by **developers** (not just testers) on user selectable git trees
  - Scriptable: automated suites must be controllable by high level automation including ad-hoc shell scripts running on developer workstations
  - Understandable: test failure reports must be rich enough to allow basic failure analysis without further reproduction

An aside: Automate everything - No excuses

"we should use a board farm for that"

"it's too much work"

"we need equipment to do an automatic reboot"

"I don't understand CI"

Computers are **good** at accurately repeating stuff…

… and you are **not** …

… so **look** for things **you** can automate.

Handling regressions - git bisect

```
git bisect start <bad_sha1> <good_sha1>
git bisect run compile_kernel_and_run_test.sh
```

`compile_kernel_and_test.sh` must return one of four return values:

0:	kernel compiled and test passed
1-124,126,127:	kernel compiled, test failed
125:	kernel failed to compile, test not run
Other:	Abort bisection (catastrophic fail)

Even if it looks difficult to write `compile_kernel_and_test.sh` try anyway!

Manual bisection using git good/bad/skip is error prone

If a manual step is required (power cycle, press button, etc) then get the shell script to prompt you when you need to to perform it but automate everything else.

Bisectability testing

- Routinely test large, long-lived patchsets for bisectability
  - Integration trees for upstream board/SoC enablement are good examples for this
  - Good for vendor trees too (to ensure you can use bisect to find your own regressions)
- aiaiai again
  - In addition to reporting deltas, aiaiai can check that a patch set is bisectable
  - Builds multiple kernel configurations **for every patch in a patch set**
  - Remove the static checkers for large patchsets (static check before and after whole patchset only can reduce build time)

```
BASELINE=v4.1-rc2; KERNTEST=v4.1-rc5; \
git format-patch --stdout $BASELINE | \
aiaiai-test-patchset -v -j 4 -c $KERNTEST --bisectability \
--sparse --cppcheck --coccinelle --smatch \
$PWD defconfig,x86 defconfig,arm64,aarch64-linux-gnu- \
multi_v7_defconfig,arm,arm-linux-gnueabihf-
```

Linaro connect
Las Vegas 2016

ENGINEERS
AND DEVICES
WORKING
TOGETHER

Graphic by Jost, CC0-PD

Linaro Limited Lifetime Warranty

This training presentation comes with a **lifetime warranty**.

Everyone here today can send **any** questions about today's session, at **any** point in the future, to support@linaro.org .

Members can also use this address to get support on any other Linaro output. Engineers from **club** and **core** members can also contact support to discuss how premium services can help you with additional services and training.

Thanks to Andrew Hennigan for introducing me to the idea of placing a guarantee on training.

Thank You

#LAS16

For further information: www.linaro.org or support@linaro.org

LAS16 keynotes and videos on: connect.linaro.org

Bonus - Upstreaming a new architecture

- Logically split the work
  - mach-foo/ family-specific ops (arm-soc team)
  - clocksource driver (Daniel Lezcano/Thomas Gleixner)
  - irq controller driver (Thomas Gleixner)
  - Device Tree binding (DT team / arm-soc team)
  - multi_v7_defconfig or add new defconfig (arm-soc team)
  - DTS to enable platform foo (arm-soc team)
- Split to divide amongst maintainer trees
- Split to divide according to arm-soc categories

ENGINEERS
AND DEVICES
WORKING
TOGETHER

Less work for ARMv8 architectures

- Logically split the work
  - ~~mach-foo/ family-specific ops (arm-soc team)~~
  - clocksource driver (Daniel Lezcano/Thomas Gleixner)
  - ~~irq controller driver (Thomas Gleixner)~~
  - Device Tree binding (DT team / arm-soc team)
  - ~~multi_v7_~~defconfig ~~or add new defconfig (arm-soc team)~~
  - DTS to enable platform foo (arm-soc team)
- Split up to divide cleanly amongst maintainer trees
- Split up to divide according to arm-soc categories

Understanding arm-soc

- Divide patches going to arm-soc:
  - next/fixes
  - next/cleanup
  - next/soc
  - next/drivers
  - next/boards
  - next/dt
- Usually the platform maintainer worries the most about this
- Individual contributors should know this to break up their commits to fit these categories
- More details:
  - http://elinux.org/images/4/48/Elc2013_Johansson.pdf

Allwinner A1X Upstreaming - Cover Letter

Subject: [PATCH 0/6] Add basic support for Allwinner A1X SoCs
From: Maxime Ripard <maxime.ripard@xxxxxxxxxxxxxxxxx>
Date: Thu, 15 Nov 2012 23:46:19 +0100

Hi,

You'll find in this patchset the initial support for Allwinner A10 and A13 SoCs from Allwinner. Since the internal name of these SoCs are sun4i and sun5i, the mach- directory is named sunxi.

You can find these SoCs in the Cubieboard, the A13-olinuxino or the Melee A1000.

Both SoCs should work fine, as the A13 is a trimmed down version of the A10, but it has only been tested on a A13-OlinuXino from Olimex.

Support is quite minimal for now, since it only includes timer and IRQ controller drivers, so we can only boot to userspace through initramfs. Support for the other peripherals on these SoCs will come eventually.

Allwinner A1X Upstream - Comments

Subject: Re: [PATCH 4/6] ARM: sunxi: Add earlyprintk support
From: Thomas Petazzoni <thomas.petazzoni@xxxxxxxxxxxxxxxxxx>
Date: Fri, 16 Nov 2012 11:41:36 +0100

On Thu, 15 Nov 2012 23:46:23 +0100, Maxime Ripard wrote:
...

> +#define SUNXI_UART1_PHYS_BASE 0x01c28400
> +#define SUNXI_UART1_VIRT_BASE 0xf1c28400

Maybe:

```
#ifdef CONFIG_DEBUG_SUNXI_UART1
#define SUNXI_UART_DEBUG_PHYS_BASE 0x01c28400
#define SUNXI_UART_DEBUG_VIRT_BASE 0xf1c28400
#endif
```

Allwinner A1X Upstreaming - Changelog

Subject: [PATCH v2 0/7] Add basic support for Allwinner A1X SoCs
From: Maxime Ripard <maxime.ripard@xxxxxxxxxxxxxxxxx>

You'll find in this patchset the initial support for Allwinner A10 and A13 SoCs from Allwinner. Since the internal name of these SoCs are sun4i and sun5i, the mach- directory is named sunxi.

...

Changes from v1:

- **Changed the earlyprintk support to add a more generic mechanism, since boards can have both the debug UART on UART0 or UART1**
- **Small fixes in the dt: moved the memory node to the dtsi, fixed the memory size on A13**
- **Simplified the irq controller driver as suggested by Stefan Roese**
- **Removed the hardcoded clock frequency in the timer to a fixed rate clock using clk framework**
- **Added a README file to the documentation to mention the supported SoCs and the related datasheet**

Allwinner A1X Upstreaming - Acceptance

Subject: Re: [PATCH v2 0/7] Add basic support for Allwinner A1X SoCs
From: Arnd Bergman <arnd@xxxxxxxxxxxxxxxxx>

On Friday 16 November 2012, Maxime Ripard wrote:
> Le 16/11/2012 22:40, Arnd Bergmann a écrit :
> > Looks all great!
> >
> > Are you planning to send a pull request, or should we apply the
> > port from patches?
>
> Whatever you prefer, just tell me.
>

Generally, git pull requests are preferred, ideally using a signed tag that has a changeset description for the git history.

We can handle patches as well, but they cause slightly more work.

Allwinner A1X Upstreaming - request-pull

```
Subject: [GIT PULL] ARM: sunxi: Add basic support for Allwinner SoCs
From: Maxime Ripard <maxime.ripard@xxxxxxxxxxxxxxxxx>

Arnd, Olof,

Here is a pull request to add the basic support for Allwinner A1X SoCs.

Thanks,
Maxime

The following changes since commit 77b67063bb6bce6d475e910d3b886a606d0d91f7:

  Linux 3.7-rc5 (2012-11-11 13:44:33 +0100)

are available in the git repository at:

  git://github.com/mripard/linux.git tags/tags/sunxi-support-for-3.8

for you to fetch changes up to 1b106699647b56313bac707e12e7ad67180cb147:

  ARM: sunxi: Add entry to MAINTAINERS (2012-11-16 21:56:53 +0100)
...
```

Thank You

#LAS16

For further information: www.linaro.org or support@linaro.org

LAS16 keynotes and videos on: connect.linaro.org