



**Linaro  
connect**  
San Francisco 2017

# Upstreaming 101

Daniel Lezcano



[connect.linaro.org](http://connect.linaro.org)



**Linaro  
connect**

San Francisco 2017

ENGINEERS  
AND DEVICES  
WORKING  
TOGETHER

# Introduction

- This presentation is focused on upstreaming the Linux code
- What means upstreaming ?
- The Linux kernel development cycle
- The Linux kernel code organization
- Contributions
- Writing one patch
- Writing several patches
- Conclusion
- Introducing Upstreaming 201





**Linaro  
connect**

San Francisco 2017

ENGINEERS  
AND DEVICES  
WORKING  
TOGETHER

# What is upstreaming ?

- Bring your private changes to the mainstream kernel
- Be prepared to:
  - restart from scratch
  - change your approach
  - be part of the OSS community
- No deadline, no schedule, no obligation to take a patch
  - Linux is evolution, best proposed solution wins
  - Consensus is the key





**Linaro  
connect**

San Francisco 2017

ENGINEERS  
AND DEVICES  
WORKING  
TOGETHER

# Why upstream ?

- The Linux kernel has one rule: no regression
  - The community won't break your code
- The submitted code will be reviewed in detail
  - Better quality
- The community will give support
  - Better knowledge
- Stop porting out of tree code to newer kernel
  - Save money and effort





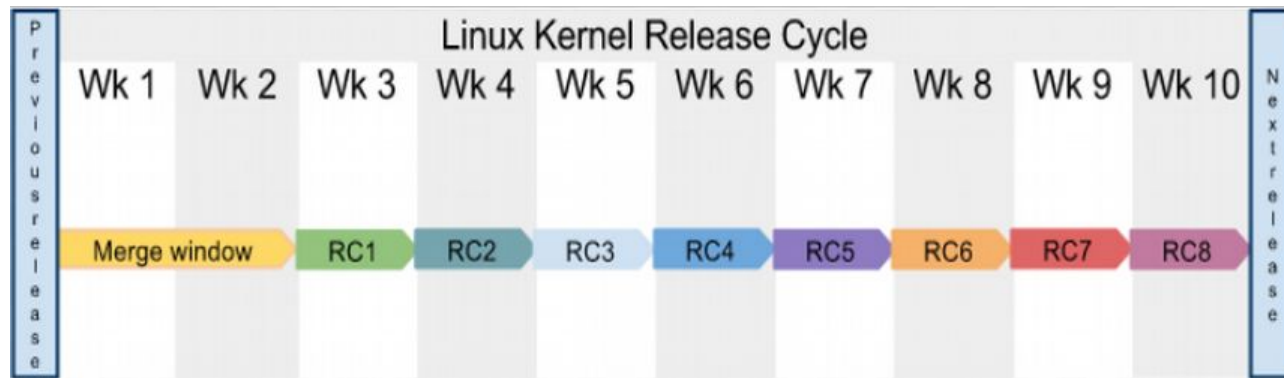
Linaro  
connect

San Francisco 2017

ENGINEERS  
AND DEVICES  
WORKING  
TOGETHER

# The Linux kernel development cycle

1. Iterative **release candidates**: v4.10-rcX, up to -rc8
  - Usually every Sunday
2. New release: **v4.10**
3. Merge window
4. New release candidate: v4.11-rc1
5. ... and so on





**Linaro  
connect**

San Francisco 2017

ENGINEERS  
AND DEVICES  
WORKING  
TOGETHER



# The Linux kernel development cycle

- A new release every 3 months
- When to send patches?
  - Fixes: anytime
  - New code and cleanup: depend on the maintainer
- How long is the merge window?
  - The merge window lasts two weeks
- How to know if the merge window is happening?
  - As soon as there is a new release
- How to know if the merge window is finished?
  - As soon as there is a -rc1



Linaro  
connect

San Francisco 2017

ENGINEERS  
AND DEVICES  
WORKING  
TOGETHER

# The Linux kernel development cycle

- Automatic with a RSS feed:
  - <https://www.kernel.org/feeds/kdist.xml>
- Manually by polling:
  - <https://www.kernel.org/>

Protocol	Location
HTTP	<a href="https://www.kernel.org/pub/">https://www.kernel.org/pub/</a>
GIT	<a href="https://git.kernel.org/">https://git.kernel.org/</a>
RSYNC	<a href="rsync://rsync.kernel.org/pub/">rsync://rsync.kernel.org/pub/</a>

Latest Stable Kernel:



4.9.10

mainline:	<b>4.10-rc8</b>	2017-02-12	[tar.xz]	[pgp]	[patch]		[view diff]	[browse]
stable:	4.9.10	2017-02-14	[tar.xz]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse] [changelog]
longterm:	4.4.49	2017-02-14	[tar.xz]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse] [changelog]
longterm:	4.1.38	2017-01-18	[tar.xz]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse] [changelog]
longterm:	3.18.48 [EOL]	2017-02-08	[tar.xz]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse] [changelog]
longterm:	3.16.39	2016-11-20	[tar.xz]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse] [changelog]
longterm:	3.12.70	2017-02-01	[tar.xz]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse] [changelog]
longterm:	3.10.105	2017-02-10	[tar.xz]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse] [changelog]
longterm:	3.4.113	2016-10-26	[tar.xz]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse] [changelog]
longterm:	3.2.84	2016-11-20	[tar.xz]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse] [changelog]
linux-next:	next-20170216	2017-02-16						[browse]



**Linaro  
connect**

San Francisco 2017

ENGINEERS  
AND DEVICES  
WORKING  
TOGETHER

# Linux-next : catching the issues early

- Linux-next integrates all the different trees
  - Kudos to Stephen Rothwell
- Image of the future Linux release ahead of the schedule
- Testing with this tree allows to catch the bugs before they hit mainline







Linaro  
connect

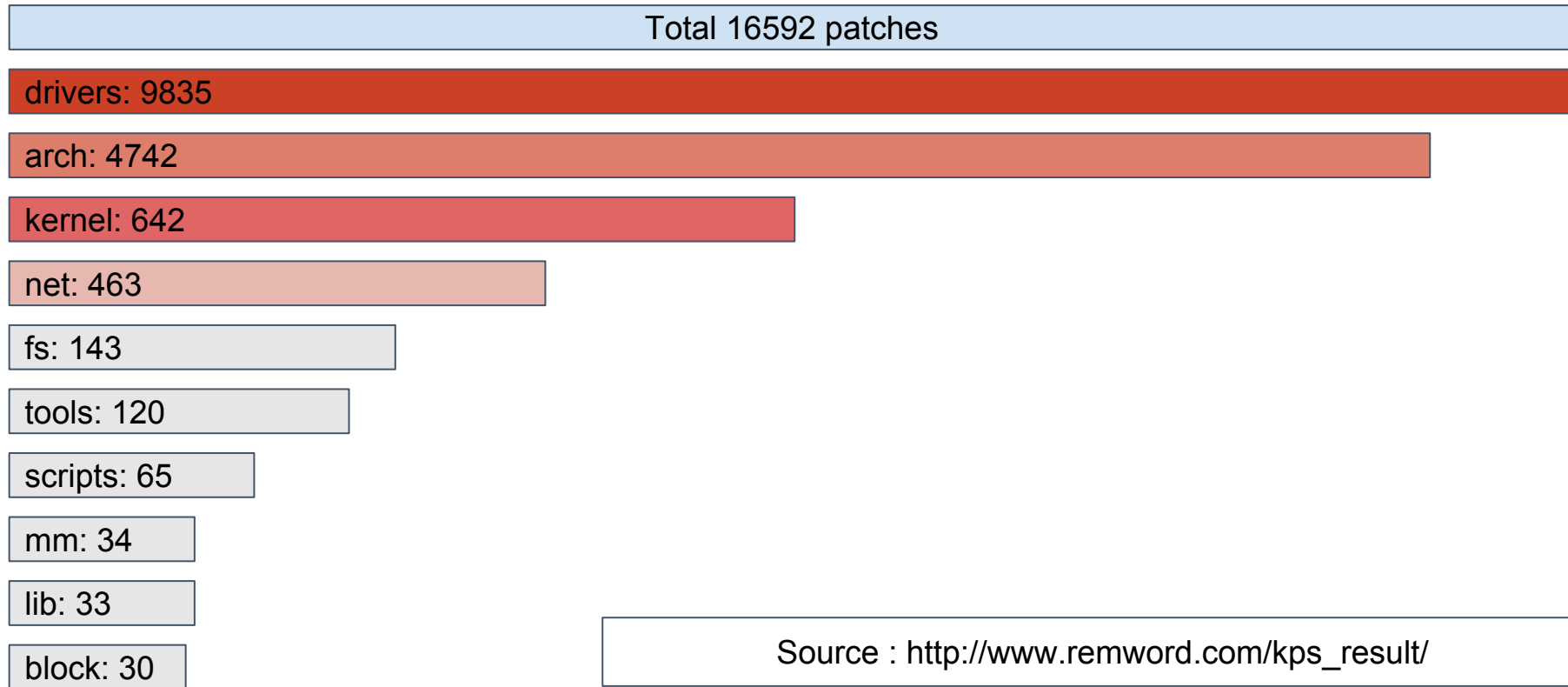
San Francisco 2017

ENGINEERS  
AND DEVICES  
WORKING  
TOGETHER

# How to start contributing ?

- Code review
  - Cleanups
  - Trivial fixes
  - Potential issue
- Compilation test coverage
  - Cross compile and fix errors / warnings
  - Use rare compilation option (eg. headers\_check, sparse, ...)
- Communicate
  - Dig into mailing lists to help people
  - Review patches
- Test linux-next
  - Boot and fix bugs
  - Report compilation warnings
- Debug
  - <https://bugzilla.kernel.org>

# Linaro contribution for 4.9





**Linaro  
connect**

San Francisco 2017

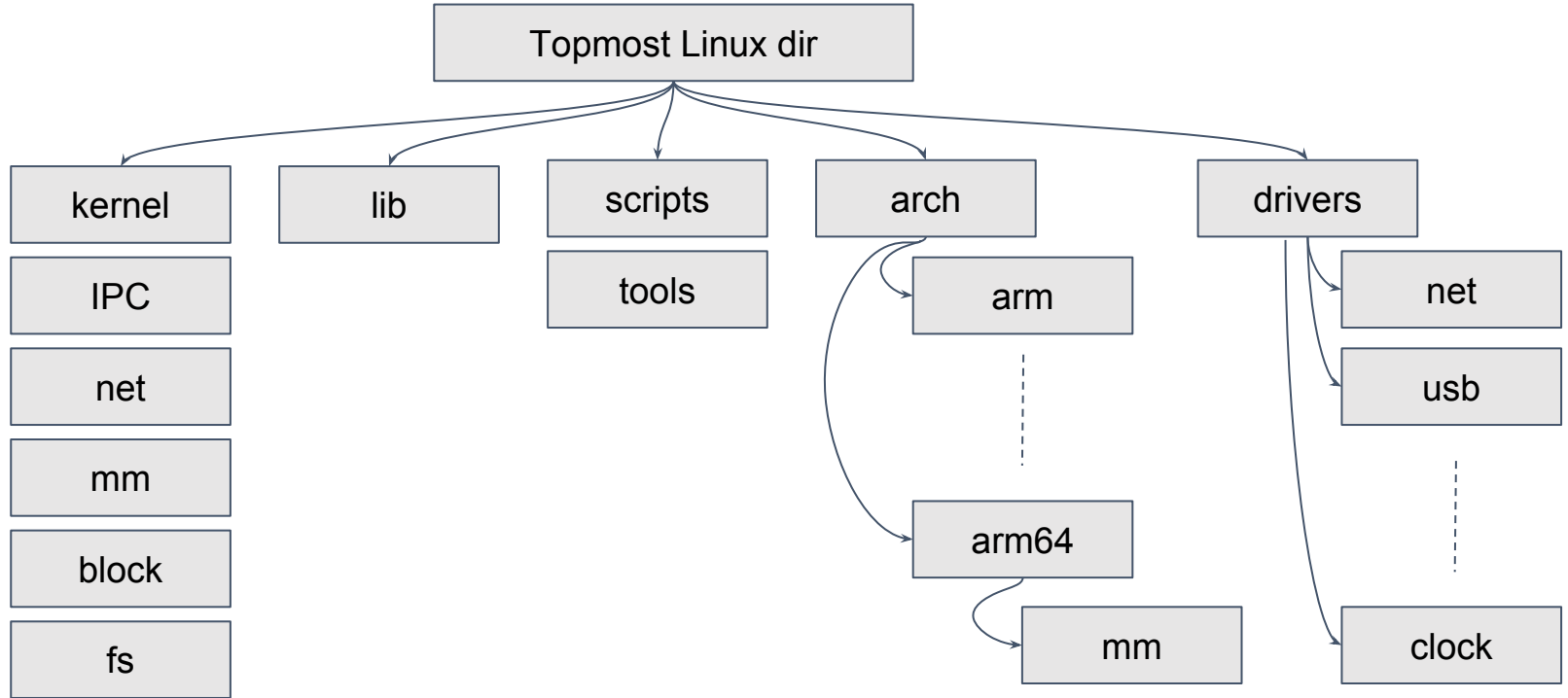
ENGINEERS  
AND DEVICES  
WORKING  
TOGETHER

# Linux code organization

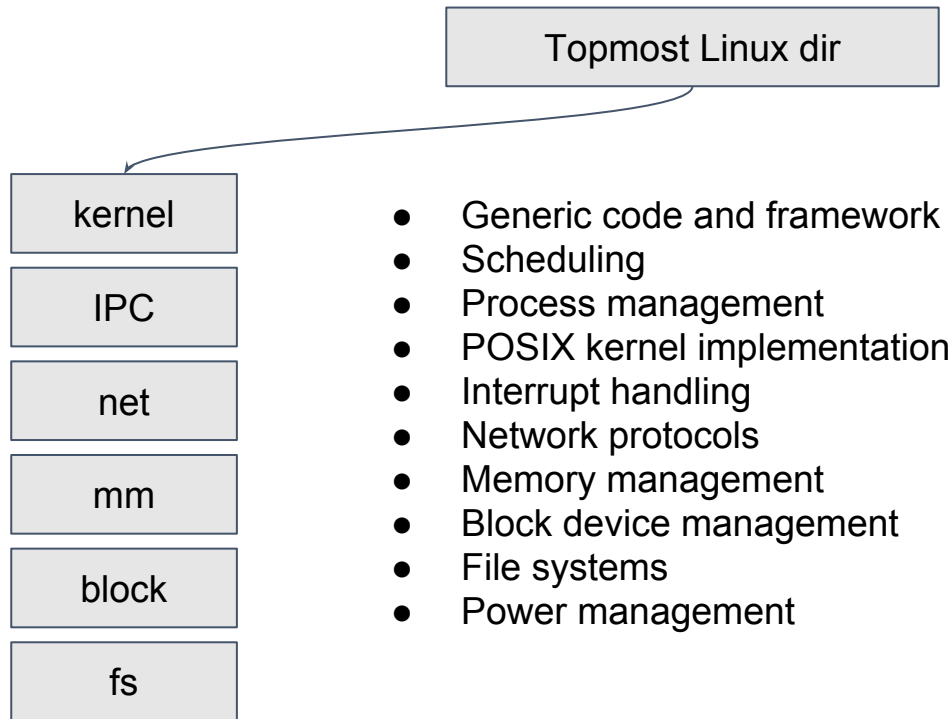
- Huge number of files, more than 40,000
- Organized in subsystems
- Posix implementation
- Architecture specific
- Frameworks
- Drivers
- Network stack



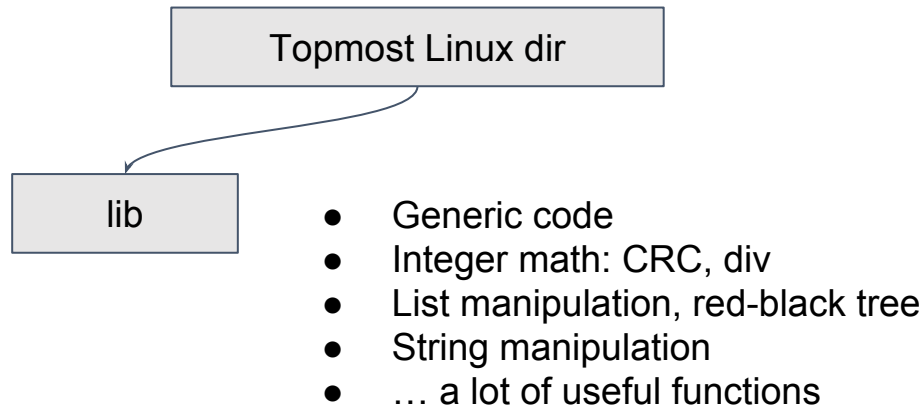
# Linux kernel code organization



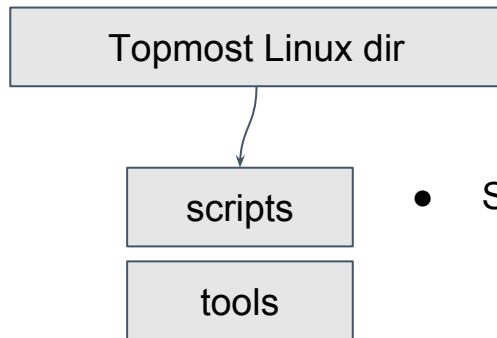
# Linux kernel code organization



# Linux kernel code organization



# Linux kernel code organization

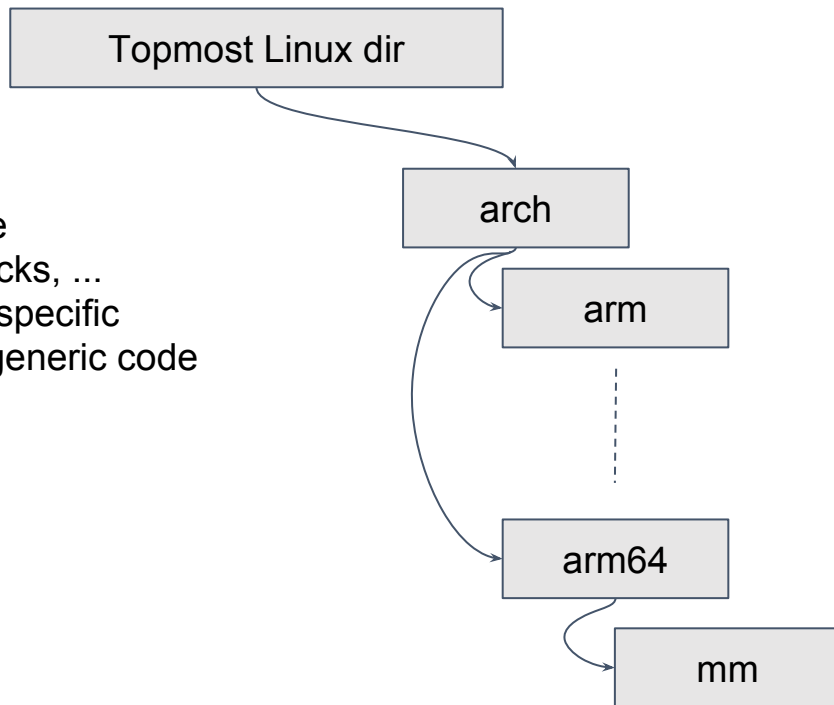


- Scripts to help making Linux better
  - Patch checking, header check
  - Kernel symbol size comparison
  - ...
- Userspace tools with a dependency on the kernel sources



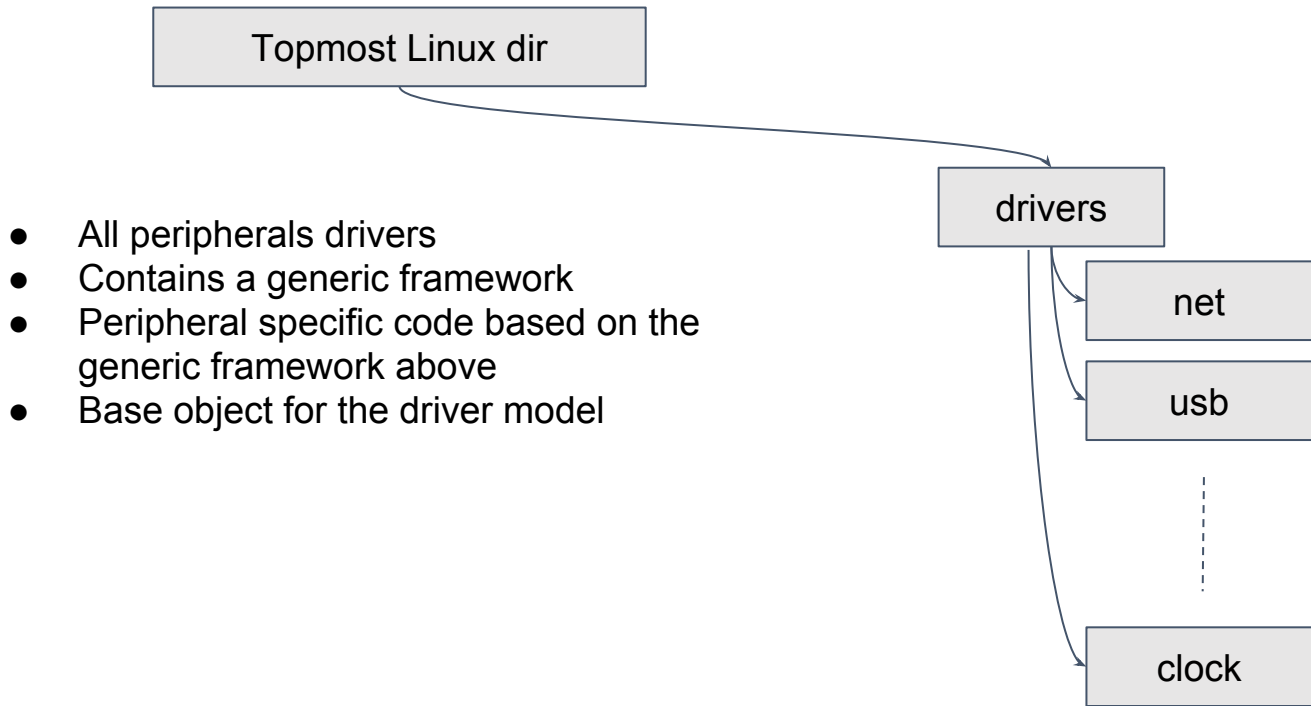
# Linux kernel code organization

- Architecture specific code
- Memory management, locks, ...
- Everything which is arch specific
- Fills the gaps left by the generic code





# Linux kernel code organization





**Linaro  
connect**

San Francisco 2017

ENGINEERS  
AND DEVICES  
WORKING  
TOGETHER

# What is a maintainer ?

- Responsible of a part of the Linux kernel code
- Ensure the code complies with some rules:
  - Coding style
  - Consistency
  - Consensus
  - Technically relevant
- A maintainer is a gatekeeper





**Linaro  
connect**

San Francisco 2017

ENGINEERS  
AND DEVICES  
WORKING  
TOGETHER

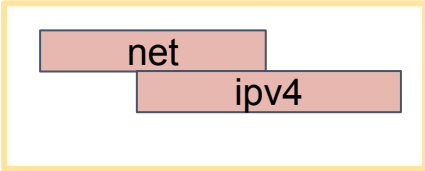
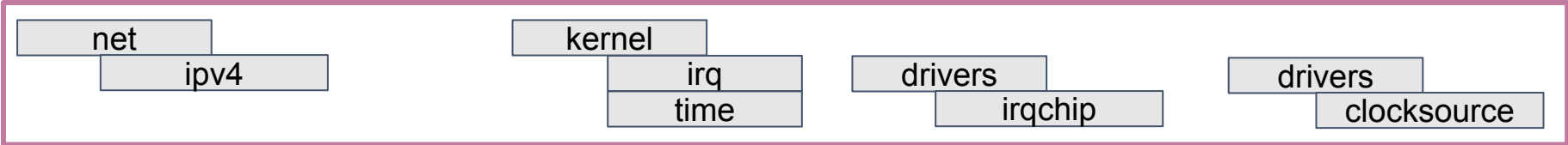
# Maintainership

- Each directory falls under the umbrella of a maintainer
- Each maintainer has its own tree
- Each proposed change must stick to the relevant tree
  
- After a kernel release, all the maintainers' tree are merged together : it is the merge window
- Topmost maintainer is Linus Torvalds
  
- All maintainers are listed in the MAINTAINERS file

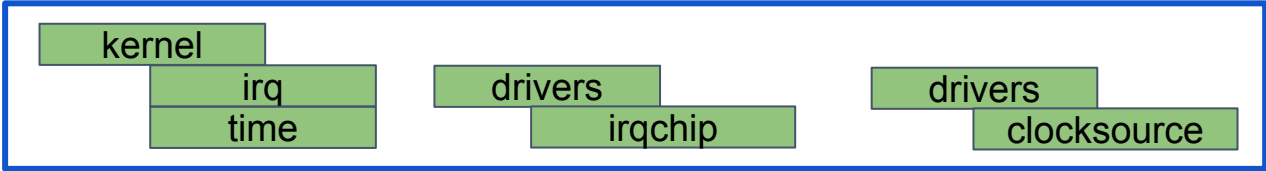


# Example of a merge process

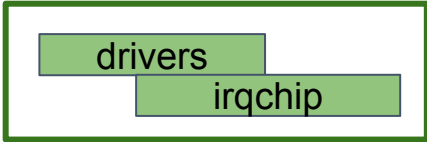
Linus Torvald's tree - v4.10



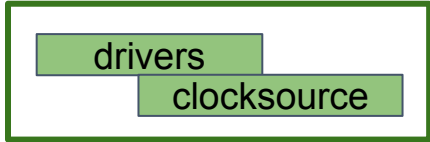
v4.10+net



v4.10+tip



v4.10+tip

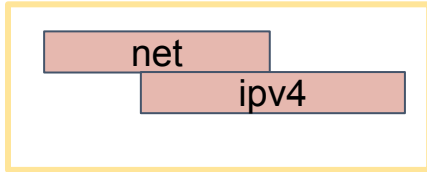
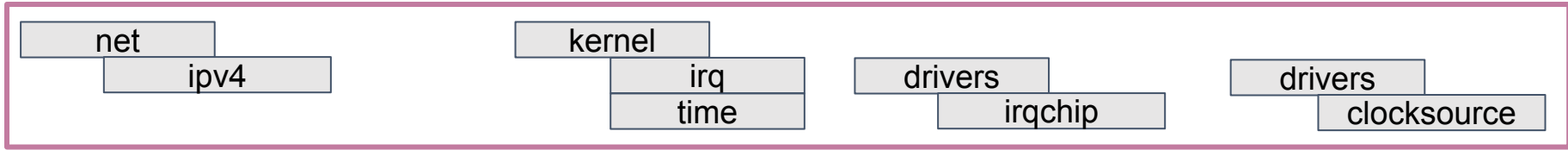


v4.10+tip

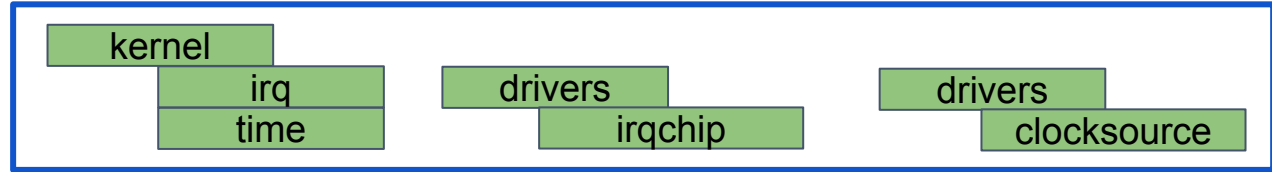


# Example of a merge process

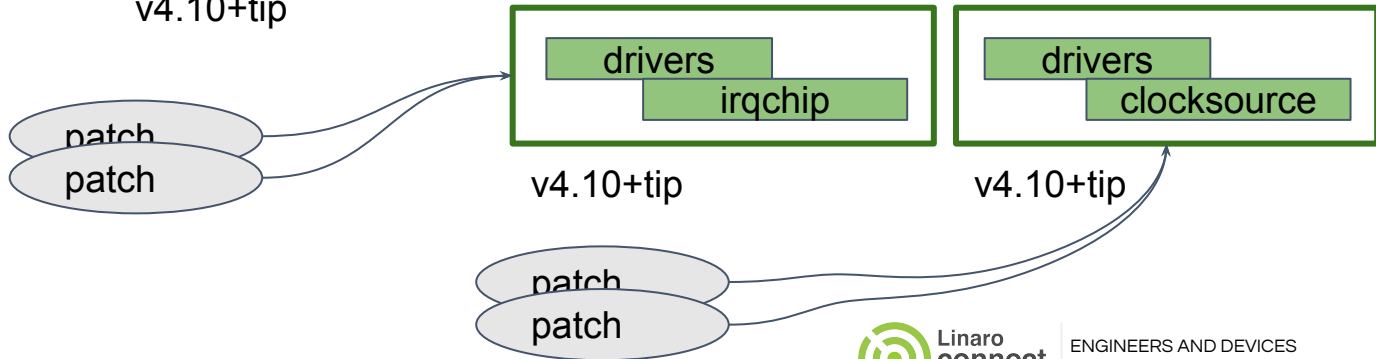
Linus Torvald's tree - v4.10



v4.10+net

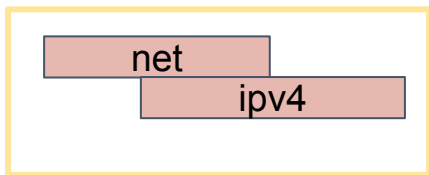
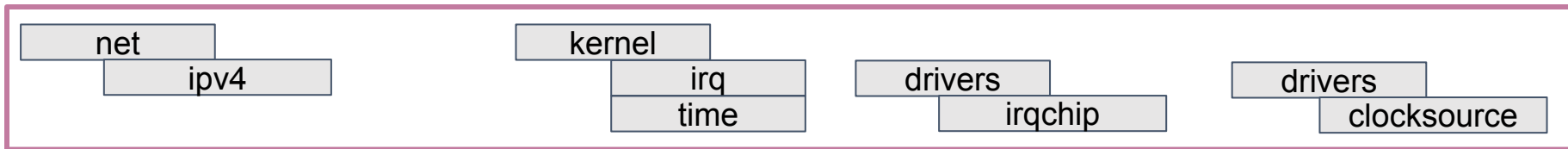


v4.10+tip

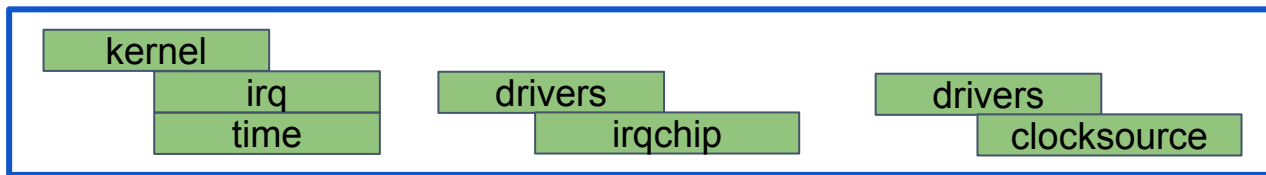


# Example of a merge process

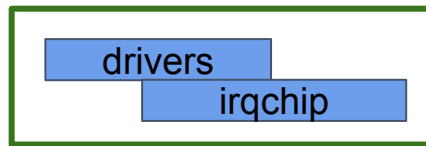
Linus Torvald's tree - v4.10



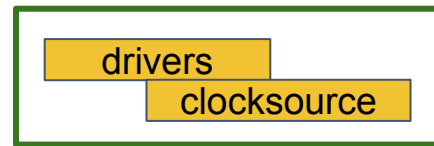
v4.10+net



v4.10+tip



v4.10+tip+irqchip

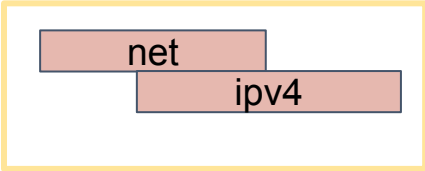
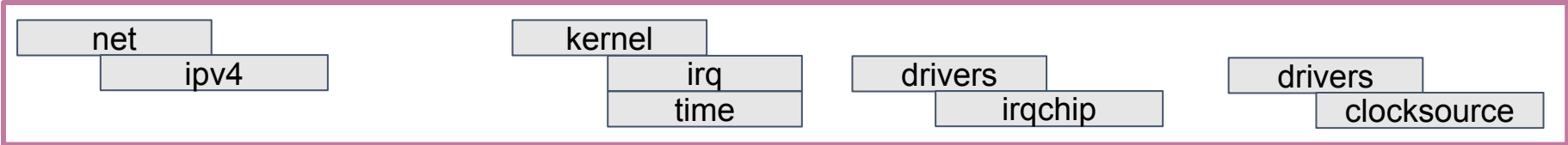


v4.10+tip+clocksource

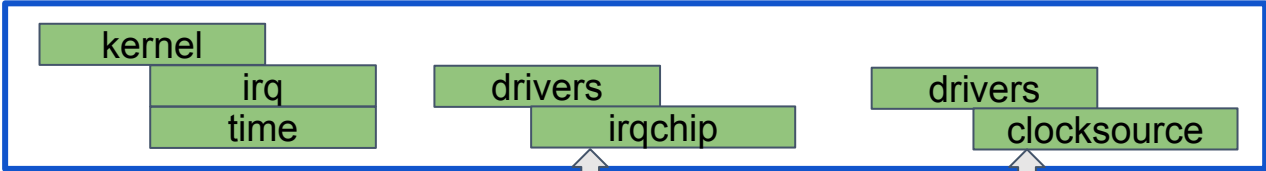


# Example of a merge process - step 1

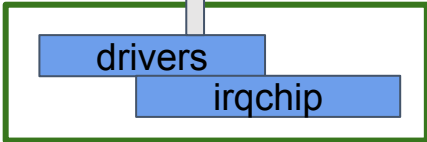
Linus Torvald's tree - v4.10



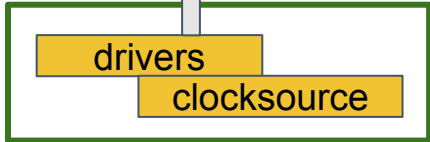
v4.10+net



v4.10+tip



v4.10+tip+irqchip

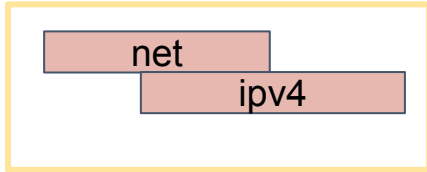
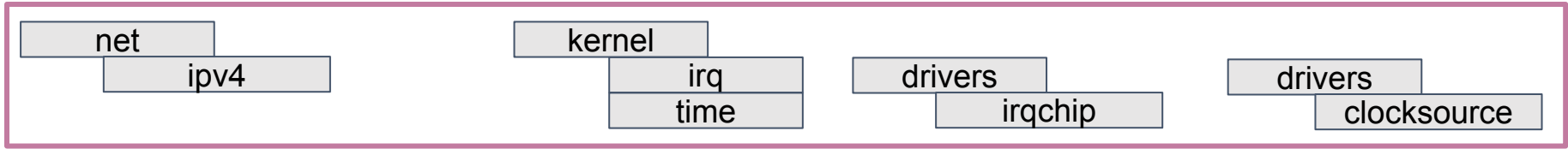


v4.10+tip+clocksource

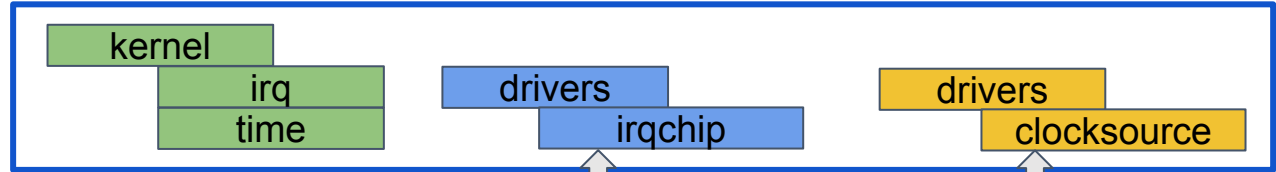


# Example of a merge process - step 1

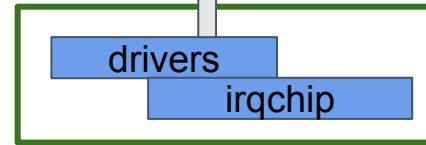
Linus Torvald's tree - v4.10



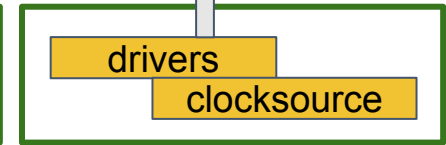
v4.10+net



v4.10+tip



v4.10+tip+irqchip



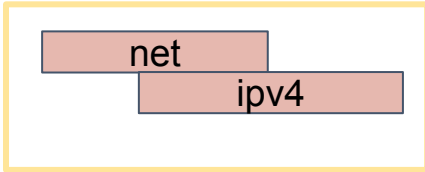
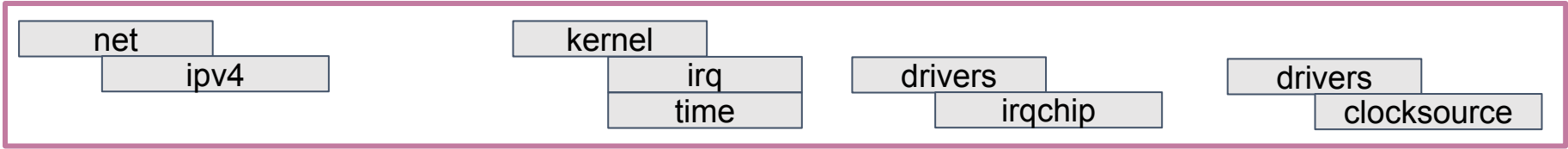
v4.10+tip+clocksource



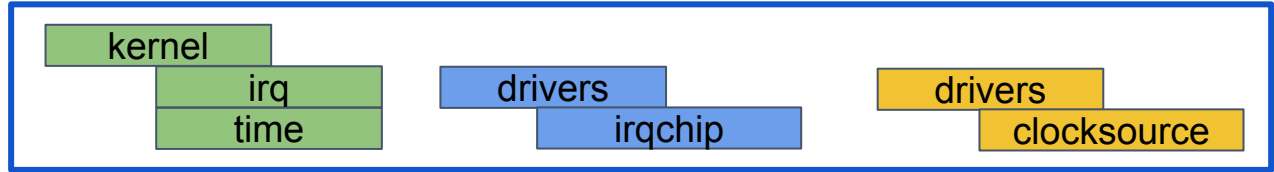


# Example of a merge process - step 1

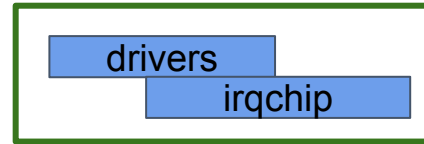
Linus Torvald's tree - v4.10



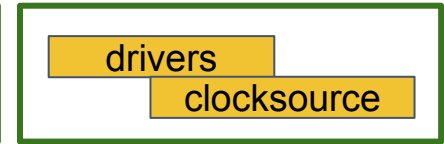
v4.10+net



v4.10+tip++



v4.10+tip+irqchip

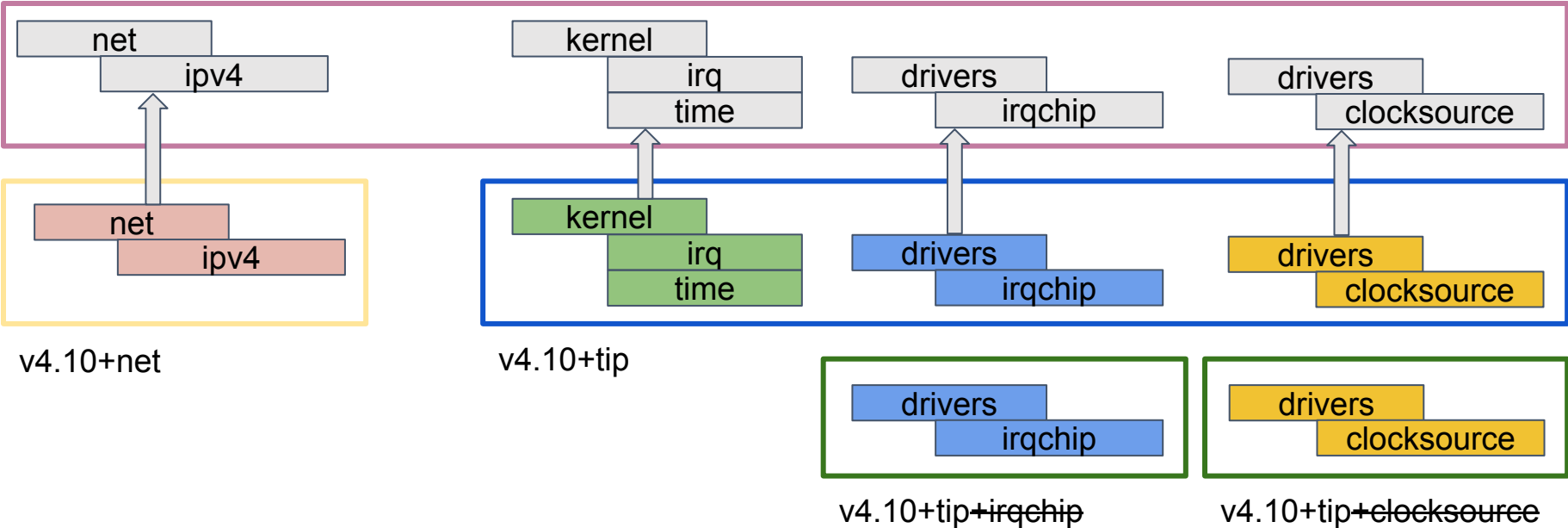


v4.10+tip+clocksource



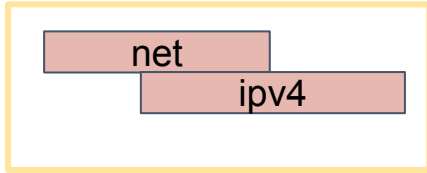
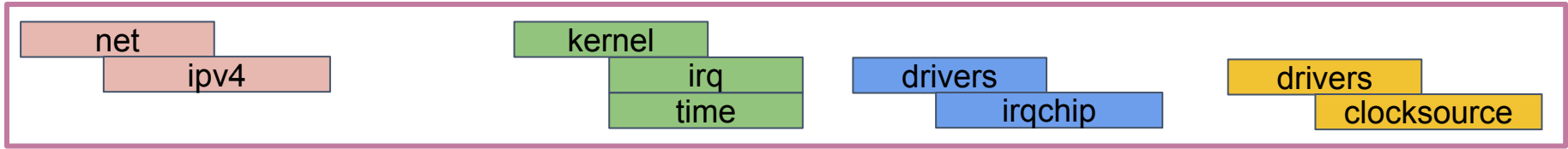
# Example of a merge process - step 2

Linus Torvald's tree - v4.10

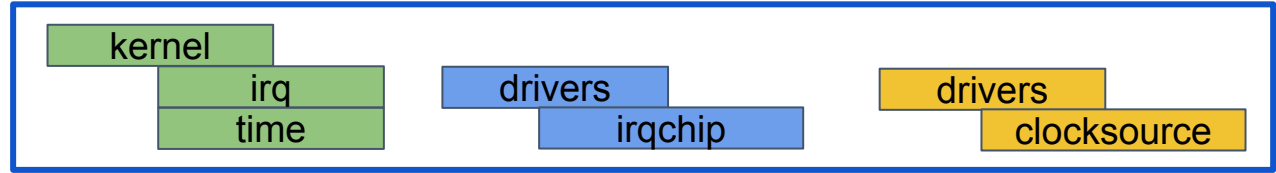


# Example of a merge process - step 2

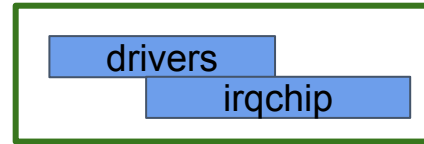
Linus Torvald's tree - v4.10++++



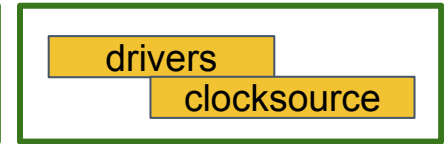
v4.10+net



v4.10+tip



v4.10+tip+irqchip

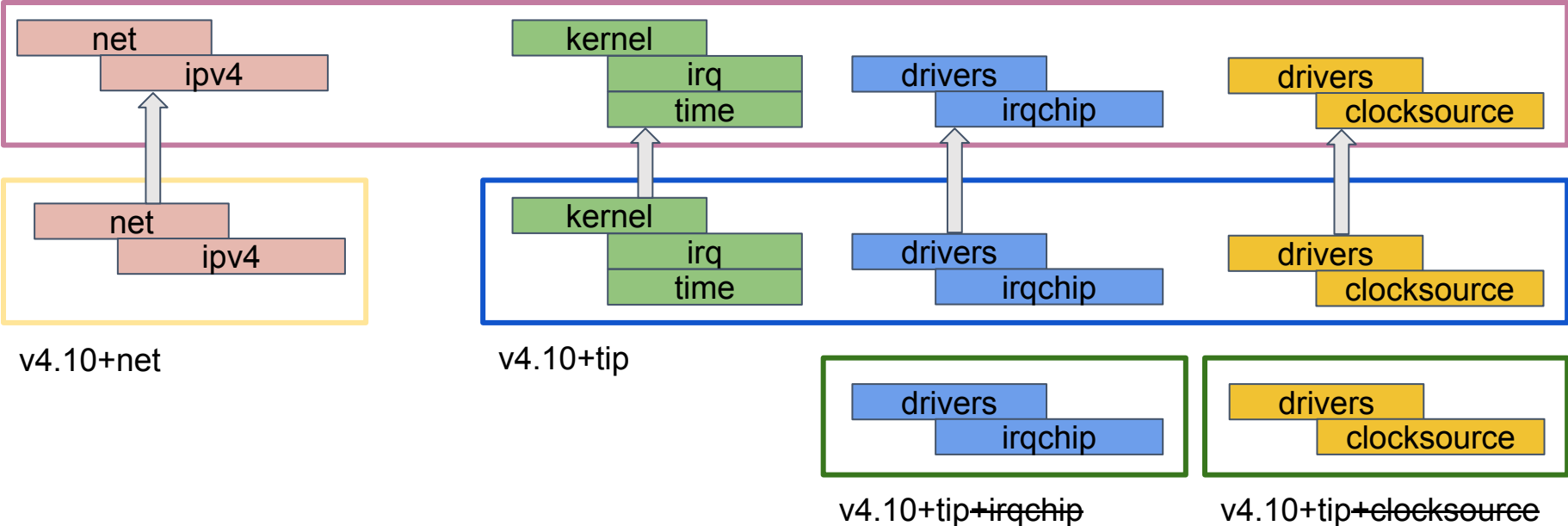


v4.10+tip+clocksource



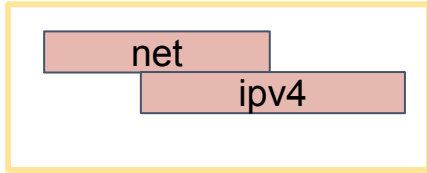
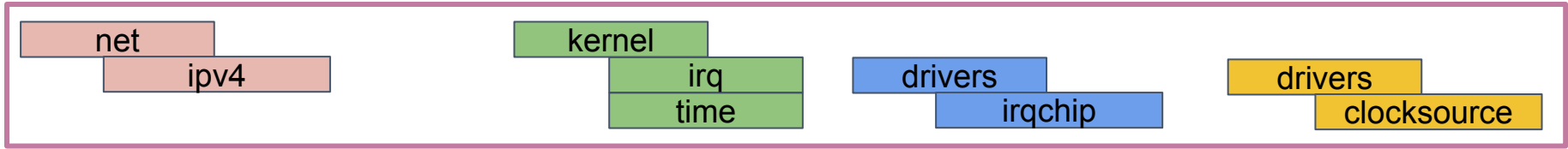
# Example of a merge process - step 2

Linus Torvald's tree - v4.10++++

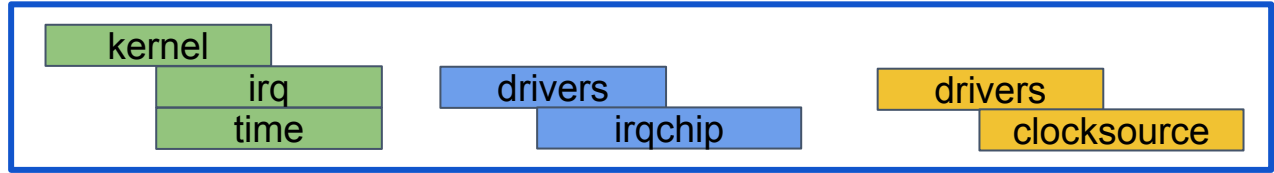


# Example of a merge process - step 3

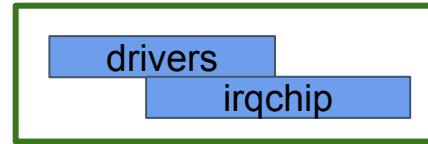
Linus Torvald's tree - v4.10-rc1



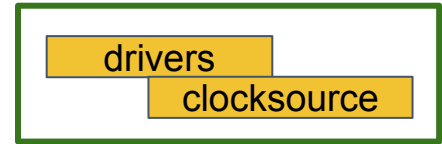
v4.10+net



v4.10+tip



v4.10+tip+irqchip



v4.10+tip+clocksource





**Linaro  
connect**

San Francisco 2017

ENGINEERS  
AND DEVICES  
WORKING  
TOGETHER

# Communication

- All development communication through the mailing lists
  - Important mailing lists for Linaro: lkml@, lakml@
  - MAINTAINERS file gives the subsystem <-> mailing list
  - <http://vger.kernel.org/vger-lists.html>
- Public discussion, no point-to-point
  - Reply-all always
- Opensource events: Linux Plumbers Conference, Embedded Linux Conference, Linaro Connect





Linaro  
connect

San Francisco 2017

ENGINEERS  
AND DEVICES  
WORKING  
TOGETHER

# Where to begin ?

- Send a simple contribution
  - But don't flood with too trivial patches
    - Read [trivial@kernel.org](mailto:trivial@kernel.org) rules
- One example with a checkpatch script
  - Run it in a directory where you will be working in the future
    - Eg. drivers/acpi
  - Target one **ERROR** spotted by checkpatch
  - Beware of false positives

```
for i in $(ls drivers/acpi/utils.c); do  
    ./scripts/checkpatch -f $i  
done
```



**Linaro  
connect**

San Francisco 2017

# A simple contribution

- Compile, test first and then commit
  - Even if the change is trivial
- Write a nice description:
  - Why and what
  - One simple sentence prefixed with the subsystem name ...
  - ... followed by a more detailed description

ENGINEERS  
AND DEVICES  
WORKING  
TOGETHER







Linaro  
connect

San Francisco 2017

ENGINEERS  
AND DEVICES  
WORKING  
TOGETHER

# A simple contribution

- Read the **Developer Certificate of Origin:**
  - Documentation/process/submitting-patches.rst
- Make sure you fully understand what that means
  - **You are legally responsible of your changes**
- When committing, add your Signed-off-by
- Don't send more than 2 trivial changes





Linaro  
connect

San Francisco 2017

ENGINEERS  
AND DEVICES  
WORKING  
TOGETHER

# A simple contribution

- A list of examples:
  - <https://goo.gl/q7NqcZ> : Fixing checkpatch errors
  - <https://goo.gl/JCn2y> : Fixing missing kfree
  - <https://goo.gl/ASrb0U> : Remove unused parameter
  - <https://goo.gl/uiktEV> : Remove pointless code





Linaro  
connect

San Francisco 2017

ENGINEERS  
AND DEVICES  
WORKING  
TOGETHER

# A more complex contribution

- Split the changes into several patches
- Bring the changes step by step, incremental changes
- Make sure the changes are git-bisect safe
  - Not following this rule will hurt your karma in the community
- Set the scene by cleaning up the place before sending a complex contribution



Linaro  
connect

San Francisco 2017

ENGINEERS  
AND DEVICES  
WORKING  
TOGETHER

# A more complex contribution

- Some examples:
  - Changing the loopback to be multi-instantiated:
    - <https://goo.gl/q6w1Ay> : Change the static variable to a pointer
    - <https://goo.gl/tdNth4> : Dynamically allocate the loopback
  - A cleanup to catch clocksource initialization error
    - <https://lkml.org/lkml/2016/6/16/781>
  - A very complex change for CPU hotplug:
    - <https://goo.gl/c2NpDY> : A long description of the changes





**Linaro  
connect**

San Francisco 2017

# Before going to Upstreaming 201

Be sure you have the right mindset

ENGINEERS  
AND DEVICES  
WORKING  
TOGETHER





**Linaro  
connect**

San Francisco 2017

ENGINEERS  
AND DEVICES  
WORKING  
TOGETHER



# Before going to Upstreaming 201

Be altruist



**Linaro  
connect**

San Francisco 2017

ENGINEERS  
AND DEVICES  
WORKING  
TOGETHER



# Before going to Upstreaming 201

Be polite



**Linaro  
connect**

San Francisco 2017

# Before going to Upstreaming 201

## Be patient

ENGINEERS  
AND DEVICES  
WORKING  
TOGETHER







**Linaro  
connect**

San Francisco 2017

ENGINEERS  
AND DEVICES  
WORKING  
TOGETHER



# Before going to Upstreaming 201

Be humble



**Linaro  
connect**

San Francisco 2017

ENGINEERS  
AND DEVICES  
WORKING  
TOGETHER



# Before going to Upstreaming 201

Be factual



**Linaro  
connect**

San Francisco 2017

ENGINEERS  
AND DEVICES  
WORKING  
TOGETHER



# Before going to Upstreaming 201

**Always take into account the  
comments**



**Linaro  
connect**

San Francisco 2017

ENGINEERS  
AND DEVICES  
WORKING  
TOGETHER

# Before going to Upstreaming 201

- Read all the documentation Documentation/process/\*:
  - Coding style rules
  - Give all the details introduced in this presentation
  - Developer Certificate of Origin
  
- More material at:
  - <https://kernelnewbies.org>





Linaro  
connect

San Francisco 2017

ENGINEERS  
AND DEVICES  
WORKING  
TOGETHER

# Before going to Upstreaming 201

- Ready to send patches for review ?
  - No ... I'm scared of what they'll think of my code
- Be sure you have the right mindset:
  - Be altruist
  - Be polite
  - Be patient
  - Be humble
  - Always take into account the comments
    - You can disagree, stick to technical reasons
  - Comments can be harsh, stay polite and factual
    - Stick to technical reasons
  - Don't be afraid, increase self confidence
    - Read the kernel documentation
    - Be altruist to encourage yourself to have a positive attitude





**Linaro  
connect**  
San Francisco 2017

# Thank You

#SFO17

For further information: [www.linaro.org](http://www.linaro.org)

SFO17 keynotes and videos on: [connect.linaro.org](http://connect.linaro.org)

