OCP - Time Appliances Project

Introduction to PTP

Maciej Machnikowski

Legal Disclaimers

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT. INTEL PRODUCTS ARE NOT INTENDED FOR USE IN MEDICAL, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS.

Intel may make changes to specifications and product descriptions at any time, without notice.

All products, dates, and figures specified are preliminary based on current expectations, and are subject to change without notice.

Intel processors, chipsets, and desktop boards may contain design defects or errors known as errata, which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Alta, Red Rock Canyon, Seacliff Trail, and other code names featured are used internally within Intel to identify products that are in development and not yet publicly announced for release. Customers, licensees, and other third parties are not authorized by Intel to use code names in advertising, promotion, or marketing of any product or services and any such use of Intel's internal code names is at the sole risk of the user.

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more complete information visit www.intel.com/benchmarks.

Intel processor numbers are not a measure of performance. Processor numbers differentiate features within each processor family, not across different processor families. Go to: www.intel.com/products/processor_number.

Intel product plans in this presentation do not constitute Intel plan of record product road maps. Please contact your Intel representative to obtain Intel's current plan of record product roadmaps.

Intel does not control or audit the design or implementation of third-party benchmark data or websites referenced in this document. Intel encourages all of its customers to visit the referenced websites or others where similar performance benchmark data are reported and confirm whether the referenced benchmark data are accurate and reflect performance of systems available for purchase.

Agenda

PTP overview

Overview of PTP

Protocol Introduction

Types of timestamping

PTP on Linux

Kernel interfaces

- Ptp sysfs
- ethtool

Linuxptp

- ptp4l
- ts2phc
- phc2sys
- ptm

Agenda

PTP overview

Overview of PTP

Protocol Introduction

Types of timestamping

PTP on Linux

Kernel interfaces

- Ptp sysfs
- ethtool

Linuxptp

- ptp4l
- ts2phc
- phc2sys
- ptm

Protocol introduction

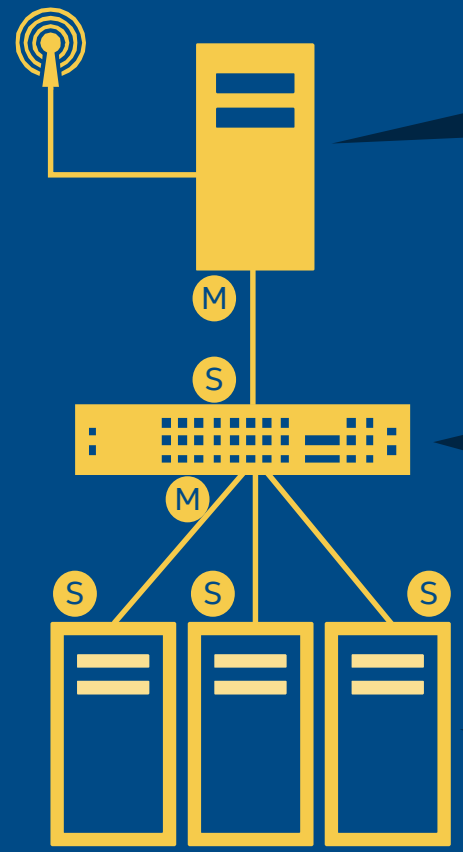
- PTP – Precision Time Protocol
- IEEE 1588 standard
- Synchronize timers to sub-microsecond accuracy
- Hierarchical M-S architecture for clock distribution
- Administration-free operation
- For both high-end devices and low-end devices

Definitions

- grandmaster clock
- ordinary clock
- boundary clock

- master clock
- slave (subordinate) clock
- PHC

Sample PTP network

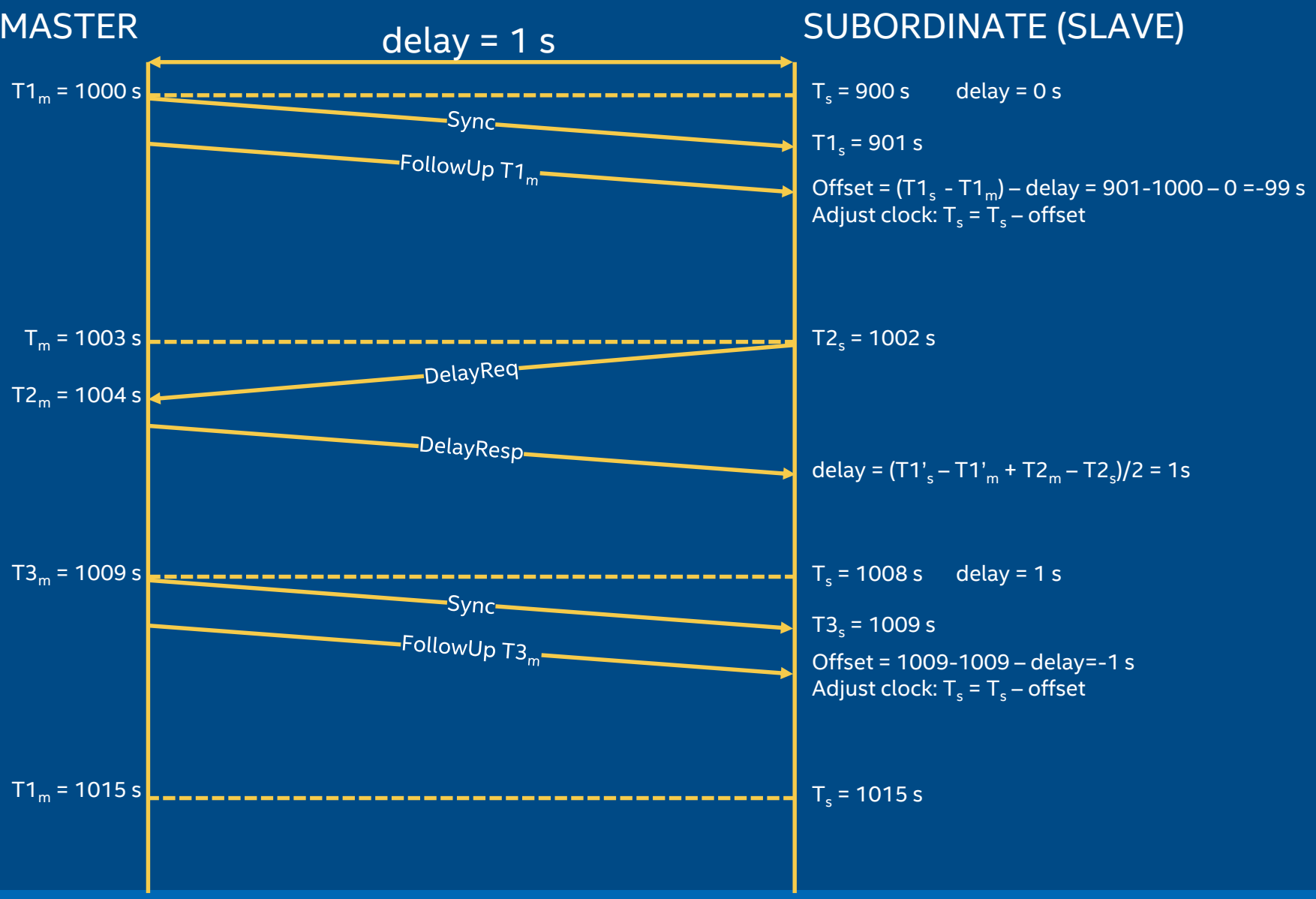


Grandmaster Clock
- Ordinary time source for the PTP network
- Typically synchronized to a very precise source (like a GPS)

Boundary Clock
- Multiple network connections
- Synchronized to a master
- Can synchronize network segment
- When Grandmaster clock is lost it may become a Grandmaster

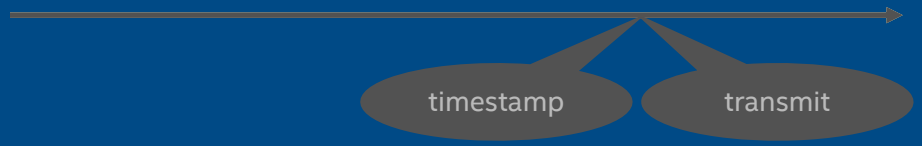
Subordinate Clock
- Ordinary clock
- Synchronizes to the master (grandmaster or boundary clock)
- May become Grandmaster if other Grandmasters are not available

IEEE 1588

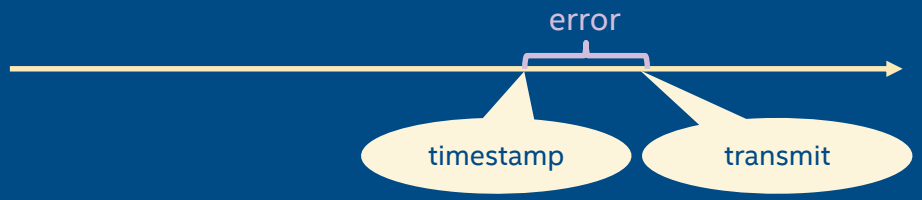

MASTER
SUBORDINATE (SLAVE)
delay = 1 s
T1m = 1000 s
Ts = 900 s
delay = 0 s
Sync
T1s = 901 s
FollowUp T1m
Offset = (T1s - T1m) – delay = 901-1000 – 0 =-99 s
Adjust clock: Ts = Ts – offset
Tm = 1003 s
T2s = 1002 s
DelayReq
T2m = 1004 s
DelayResp
delay = (T1's – T1'm + T2m – T2s)/2 = 1s
T3m = 1009 s
Ts = 1008 s
delay = 1 s
Sync
T3s = 1009 s
FollowUp T3m
Offset = 1009-1009 – delay=-1 s
Adjust clock: Ts = Ts – offset
T1m = 1015 s
Ts = 1015 s

Timing of the timestamp

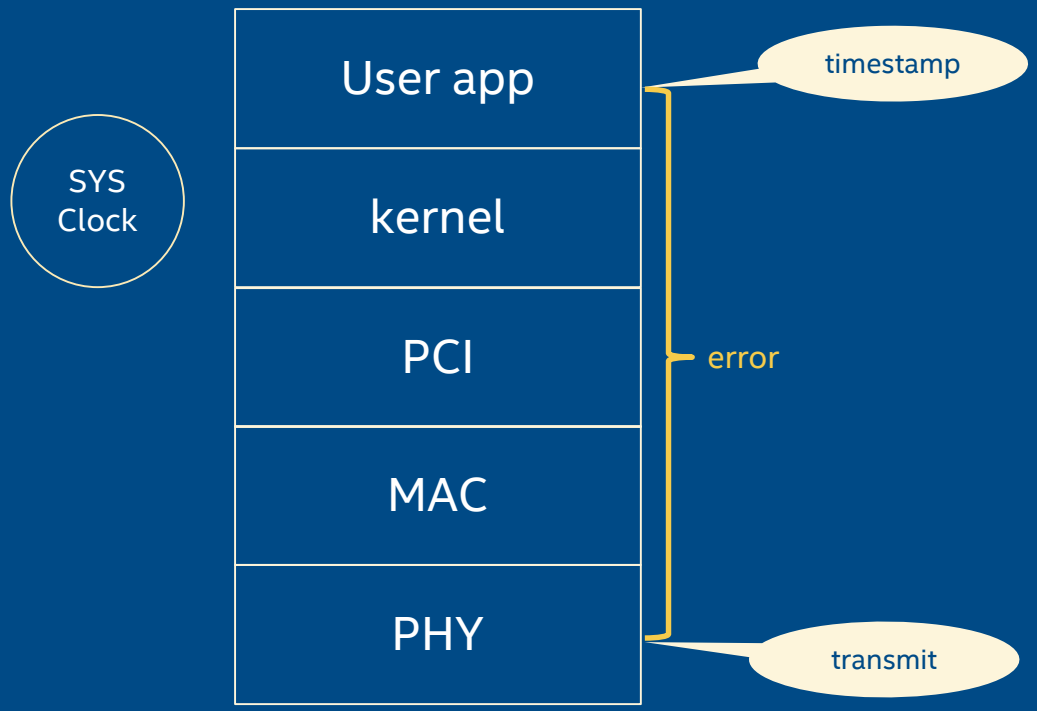
- Ideal timestamp timing:



- Real timestamp timing:



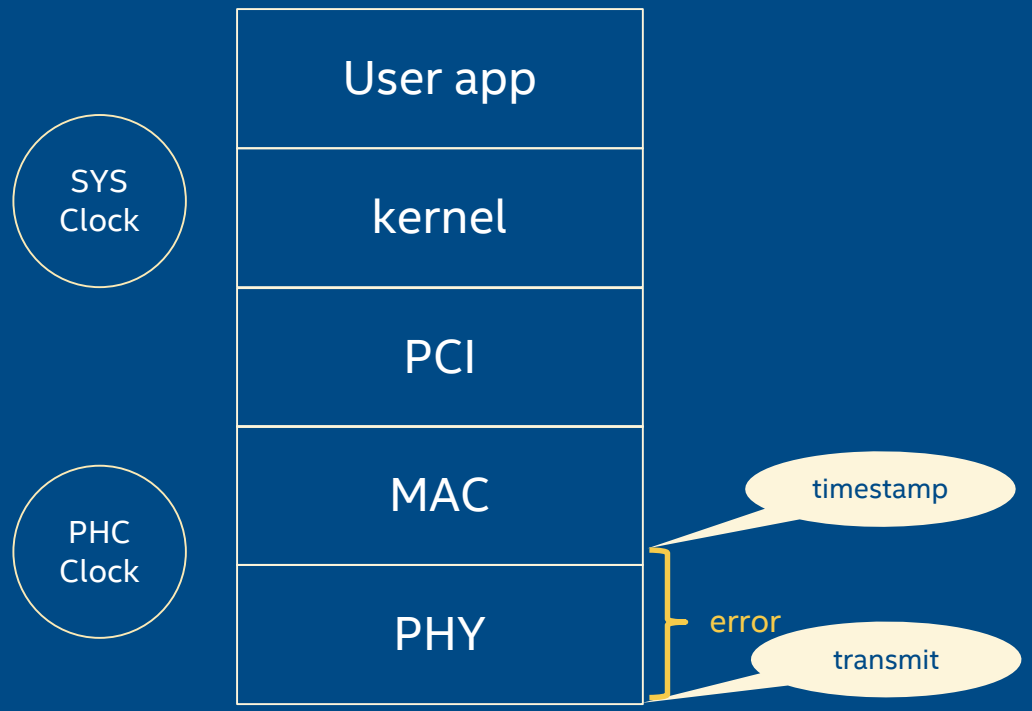
Software timestamping



1. Read system clock
2. Timestamp the packet
3. Send the packet through the stack

- Relatively BIG error.

Hardware timestamping



1. Set appropriate flags in the descriptor
2. MAC reads the PHC (PTP Hardware Clock), stamps and sends the packet

- Relatively small error.

Agenda

PTP overview

Overview of PTP

Protocol Introduction

Types of timestamping

PTP on Linux

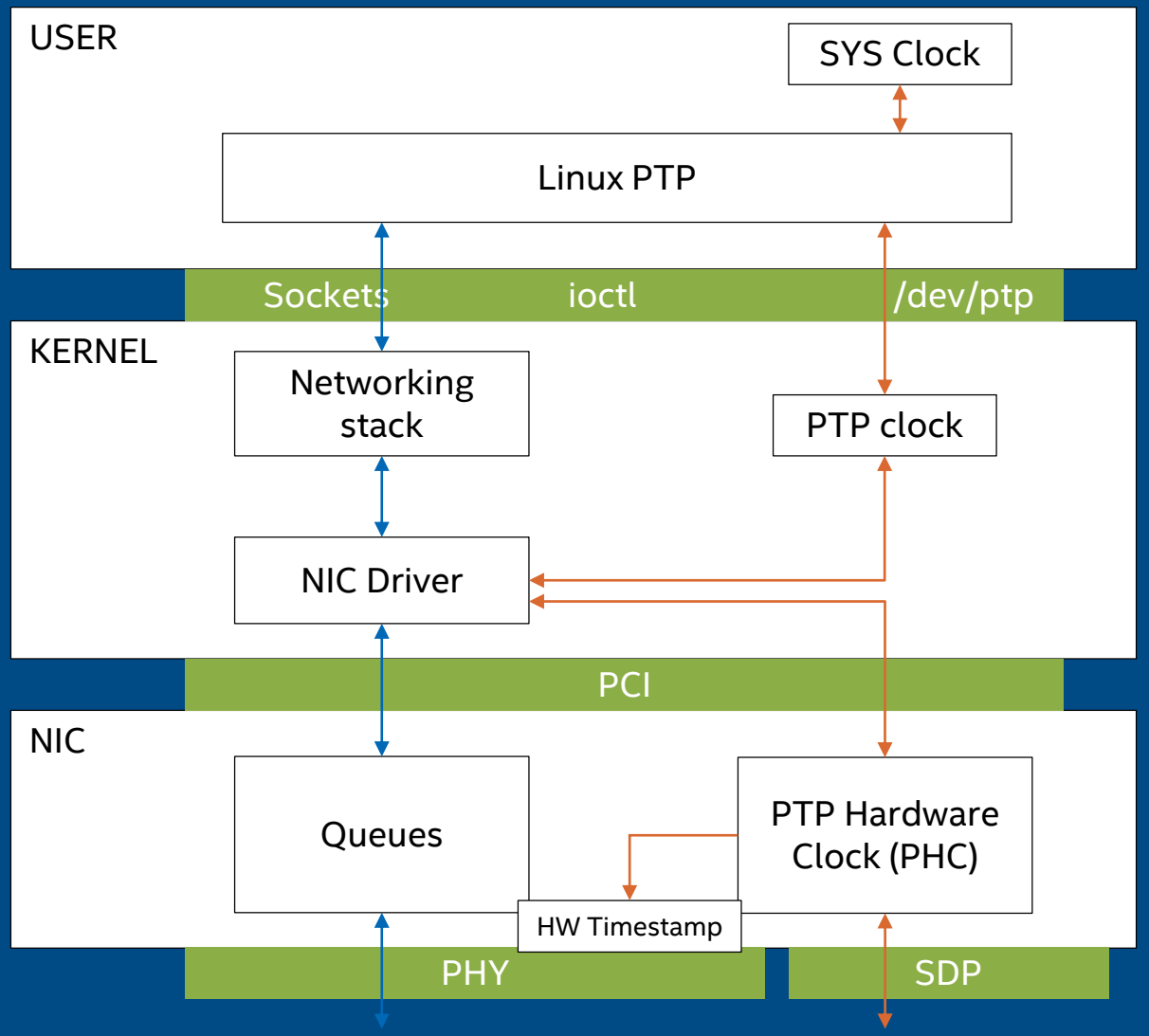
Kernel interfaces

- Ptp sysfs
- ethtool

Linuxptp

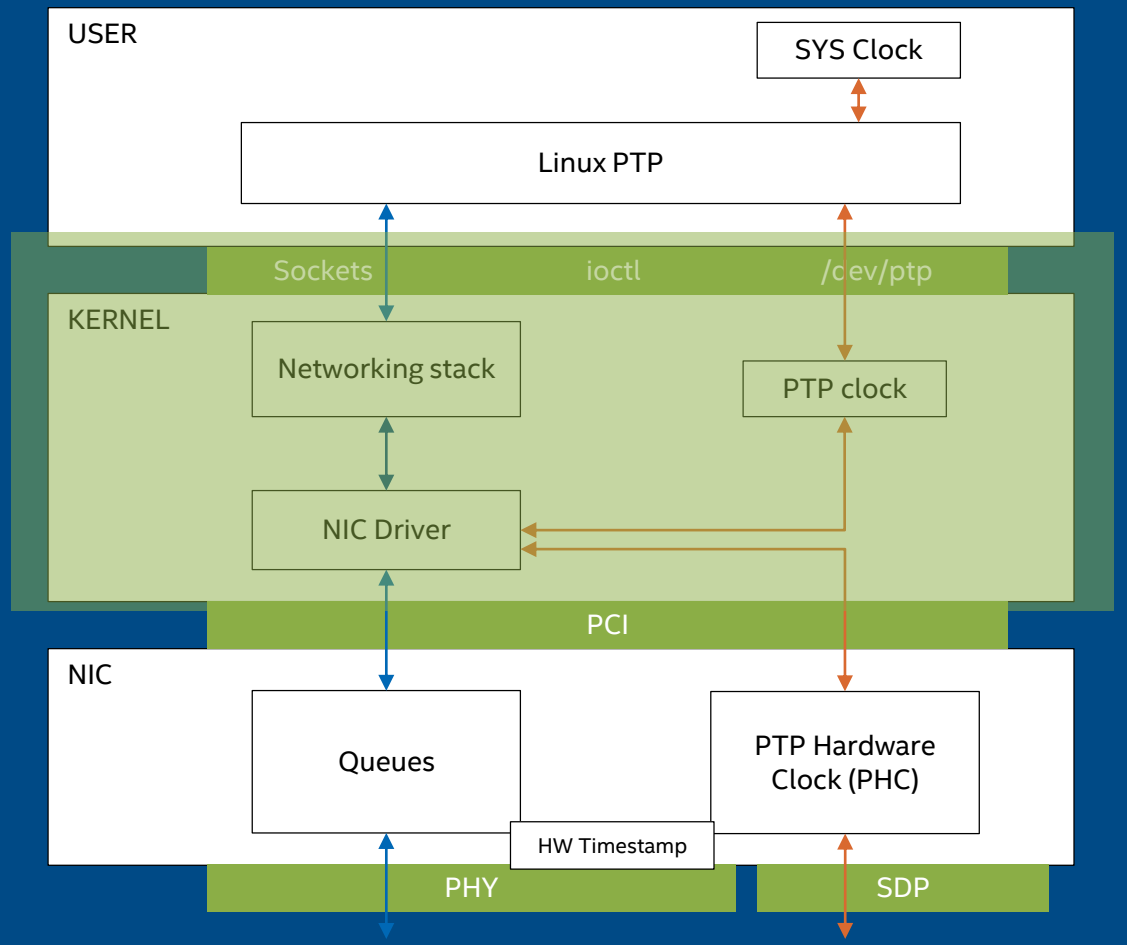
- ptp4l
- ts2phc
- phc2sys
- ptm

PTP on Linux


USER
SYS Clock
Linux PTP
Sockets
ioctl
/dev/ptp
KERNEL
Networking stack
PTP clock
NIC Driver
PCI
NIC
Queues
PTP Hardware Clock (PHC)
HW Timestamp
PHY
SDP

Linux Kernel

- External sync pins
  - standard sysfs interface
  - external timestamping input
  - periodic output
- Socket options
  - Control hardware timestamping
  - Receive timestamps
- POSIX Clock API
  - IOCTLs



PTP Pins interface

```
# echo <function> <channel> > pins/GPIO_4
```

function – desired pin function:

- 0 – none
- 1 – ext_ts
- 2- perout

channel - channel index

https://www.kernel.org/doc/Documentation/ABI/testing/sysfs-ptp

```
[root@localhost /sys/class/ptp/ptp4]# tree
.
├── clock_name
├── dev
├── device -> ../../../0000:af:00.0
├── extts_enable
├── fifo
├── max_adjustment
├── n_alarms
├── n_external_timestamps
├── n_periodic_outputs
├── n_programmable_pins
├── period
├── pins
│   ├── GPIO_4
│   ├── SDP3_2
│   └── SDP3_3
├── power
│   ├── autosuspend_delay_ms
│   ├── control
│   ├── runtime_active_time
│   ├── runtime_status
│   └── runtime_suspended_time
├── pps_available
├── pps_enable
├── subsystem -> ../../../../../../class/ptp
└── uevent
```

PTP sysfs interface

```
# echo <channel> <enable> > extts_enable
```

channel - channel index
enable – set to 1 to enable or 0 to disable

```
# echo <channel> <st_s> <st_ns> <per_s> <per_ns> > period
```

channel - channel index
st_s - start time seconds
st_ns - start time nanoseconds
per_s - period seconds
per_ns - period nanoseconds

https://www.kernel.org/doc/Documentation/ABI/testing/sysfs-ptp

```
[root@localhost /sys/class/ptp/ptp4]# tree
.
├── clock_name
├── dev
├── device -> ../../../0000:af:00.0
├── extts_enable
├── fifo
├── max_adjustment
├── n_alarms
├── n_external_timestamps
├── n_periodic_outputs
├── n_programmable_pins
├── period
├── pins
│   ├── GPIO_4
│   ├── SDP3_2
│   └── SDP3_3
├── power
│   ├── autosuspend_delay_ms
│   ├── control
│   ├── runtime_active_time
│   ├── runtime_status
│   └── runtime_suspended_time
├── pps_available
├── pps_enable
├── subsystem -> ../../../../../../class/ptp
└── uevent
```

POSIX Clock API

- Every POSIX clock is represented by a character device (/dev/ptpX)
- clock_adjtime: Adjust the clock
  - Adjfreq (adjfine)
  - adjtime
- clock_gettime: Read the current time
- clock_settime: Set the current time
- ioctl: Optional IOCTL methods

ethtool

```
# ethtool -T ens260f0
Time stamping parameters for ens260f0:
Capabilities:
        hardware-transmit     (SOF_TIMESTAMPING_TX_HARDWARE)
        software-transmit     (SOF_TIMESTAMPING_TX_SOFTWARE)
        hardware-receive      (SOF_TIMESTAMPING_RX_HARDWARE)
        software-receive      (SOF_TIMESTAMPING_RX_SOFTWARE)
        software-system-clock (SOF_TIMESTAMPING_SOFTWARE)
        hardware-raw-clock    (SOF_TIMESTAMPING_RAW_HARDWARE)
PTP Hardware Clock: 4
Hardware Transmit Timestamp Modes:
        off                   (HWTSTAMP_TX_OFF)
        on                    (HWTSTAMP_TX_ON)
Hardware Receive Filter Modes:
        none                  (HWTSTAMP_FILTER_NONE)
        ptpv1-l4-sync         (HWTSTAMP_FILTER_PTP_V1_L4_SYNC)
        ptpv1-l4-delay-req    (HWTSTAMP_FILTER_PTP_V1_L4_DELAY_REQ)
        ptpv2-l4-event        (HWTSTAMP_FILTER_PTP_V2_L4_EVENT)
        ptpv2-l4-sync         (HWTSTAMP_FILTER_PTP_V2_L4_SYNC)
        ptpv2-l4-delay-req    (HWTSTAMP_FILTER_PTP_V2_L4_DELAY_REQ)
        ptpv2-l2-event        (HWTSTAMP_FILTER_PTP_V2_L2_EVENT)
        ptpv2-l2-sync         (HWTSTAMP_FILTER_PTP_V2_L2_SYNC)
        ptpv2-l2-delay-req    (HWTSTAMP_FILTER_PTP_V2_L2_DELAY_REQ)
        ptpv2-event           (HWTSTAMP_FILTER_PTP_V2_EVENT)
        ptpv2-sync            (HWTSTAMP_FILTER_PTP_V2_SYNC)
        ptpv2-delay-req
```

Agenda

PTP overview

Overview of PTP

Protocol Introduction

Types of timestamping

PTP on Linux

Kernel interfaces

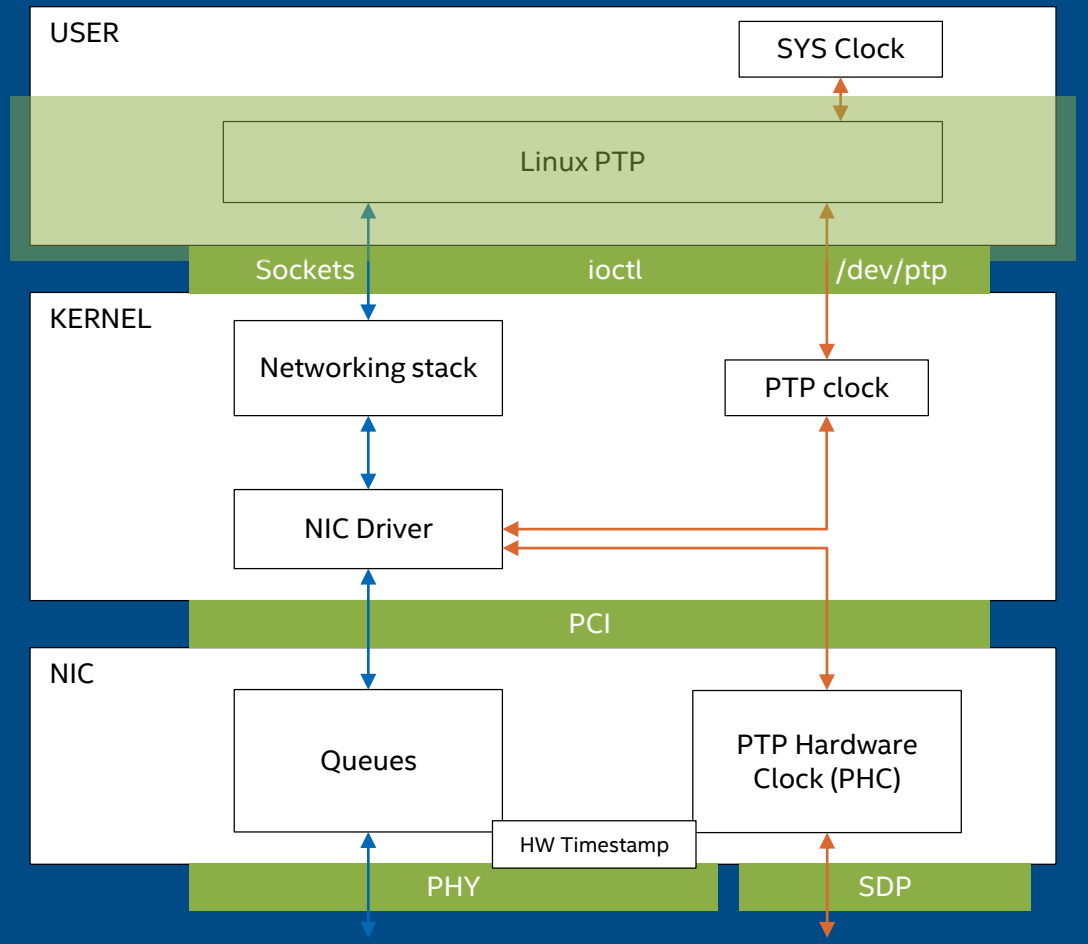
- Ptp sysfs
- ethtool

Linuxptp

- ptp4l
- ts2phc
- phc2sys
- ptm

Linux PTP project

- Set of userspace tools
- Implement PTP (IEEE 1588)
- Use standard kernel APIs to synchronize and manage the clocks
- Supports HW and SW time stamping
  - Best precision w/ HW time stamping
- http://linuxptp.sourceforge.net/



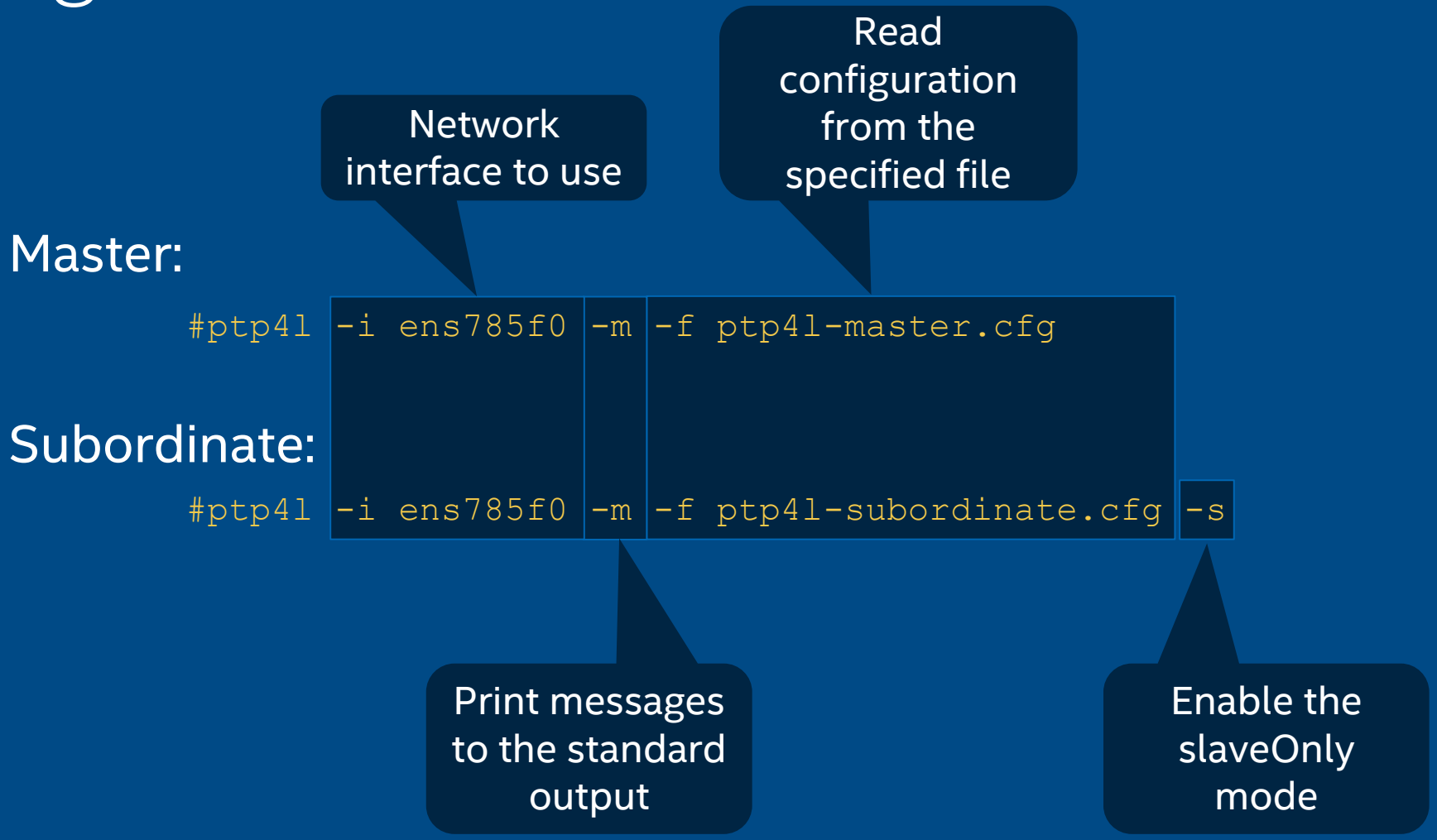
Linux PTP project

- ptp4l
  - Synchronize 2 PHCs using the PTP protocol (IEEE 1588)
- ts2phc (new tool)
  - Synchronize PHCs to external time stamp signal (1PPS signals).
- phc2sys
  - synchronize two (or more) POSIX clocks
  - Can synchronize PHC and SYS
- pmc
  - PTP management client

ptp4l

- Implements the PTP boundary clock and ordinary clock
- Synchronizes the PTP hardware clock to the master clock
- Can run as:
  - Standalone tool
  - Service
    - Options are specified in the /etc/sysconfig/ptp4l
    - Can be started using # systemctl start ptp4l
- Use PI servo to slew frequency of PHC
- https://www.mankier.com/8/ptp4l

Starting the Master and the Subordinate



ptp4l - master

```
#ptp4l -i ens785f0 -m -f ptp4l-master.cfg
ptp4l[212068.052]: selected /dev/ptp4 as PTP clock
ptp4l[212068.053]: port 1: INITIALIZING to LISTENING on INIT_COMPLETE
ptp4l[212068.054]: port 0: INITIALIZING to LISTENING on INIT_COMPLETE
ptp4l[212075.790]: port 1: LISTENING to MASTER on ANNOUNCE_RECEIPT_TIMEOUT_EXPIR
ptp4l[212075.790]: selected local clock b49691.fffe.5cea1c as best master
ptp4l[212075.790]: assuming the grand master role
```

We're the grandmaster

ptp4l - subordinate

```
#ptp4l -i ens785f0 -m -f ptp4l-subordinate.cfg -s
ptp4l[4605984.309]: selected /dev/ptp4 as PTP clock
ptp4l[4605984.311]: port 1: INITIALIZING to LISTENING on INIT_COMPLETE
ptp4l[4605984.312]: port 0: INITIALIZING to LISTENING on INIT_COMPLETE
```

Initialization

```
ptp4l[4605985.875]: port 1: new foreign master b49691.fffe.5cea1c-1
ptp4l[4605987.443]: selected local clock b49691.fffe.5ce960 as best master
ptp4l[4605989.875]: selected best master clock b49691.fffe.5cea1c
ptp4l[4605989.875]: port 1: LISTENING to UNCALIBRATED on RS_SLAVE
ptp4l[4605989.971]: port 1: minimum delay request interval 2^0
```

Finding the master

```
ptp4l[4654022.546]: master offset     -23628 s0 freq   -5470 path delay      1074
ptp4l[4654023.546]: master offset     -23636 s1 freq   -5478 path delay      1074
ptp4l[4654024.546]: master offset      -1695 s2 freq   -7173 path delay      1074
ptp4l[4654024.546]: port 1: UNCALIBRATED to SLAVE on MASTER_CLOCK_SELECTED
ptp4l[4654025.546]: master offset          5 s2 freq   -5981 path delay      1074
```

Synchronizing the PHC

ptp4l - subordinate

Synchronization state:
- s0 – servo unlocked
- s1 – clock step
- s2 – servo locked

Delay of the path

```
ptp4l[466010.881]: master offset         -5 s2 freq      -5 path delay      1073
ptp4l[466011.881]: master offset          0 s2 freq      -1 path delay      1073
ptp4l[466012.881]: master offset          2 s2 freq      +1 path delay      1073
ptp4l[466013.881]: master offset          0 s2 freq      -1 path delay      1073
ptp4l[466014.881]: master offset          0 s2 freq      -1 path delay      1073
ptp4l[466015.881]: master offset         -1 s2 freq      -2 path delay      1073
ptp4l[466016.881]: master offset         -7 s2 freq      -8 path delay      1074
ptp4l[466017.881]: master offset          6 s2 freq      +3 path delay      1074
ptp4l[466018.881]: master offset          0 s2 freq      -1 path delay      1074
ptp4l[466019.881]: master offset          1 s2 freq      -0 path delay      1074
ptp4l[466020.881]: master offset          0 s2 freq      -1 path delay      1074
ptp4l[466021.881]: master offset          0 s2 freq      -1 path delay      1074
```

Offset to the master clock (in ns)

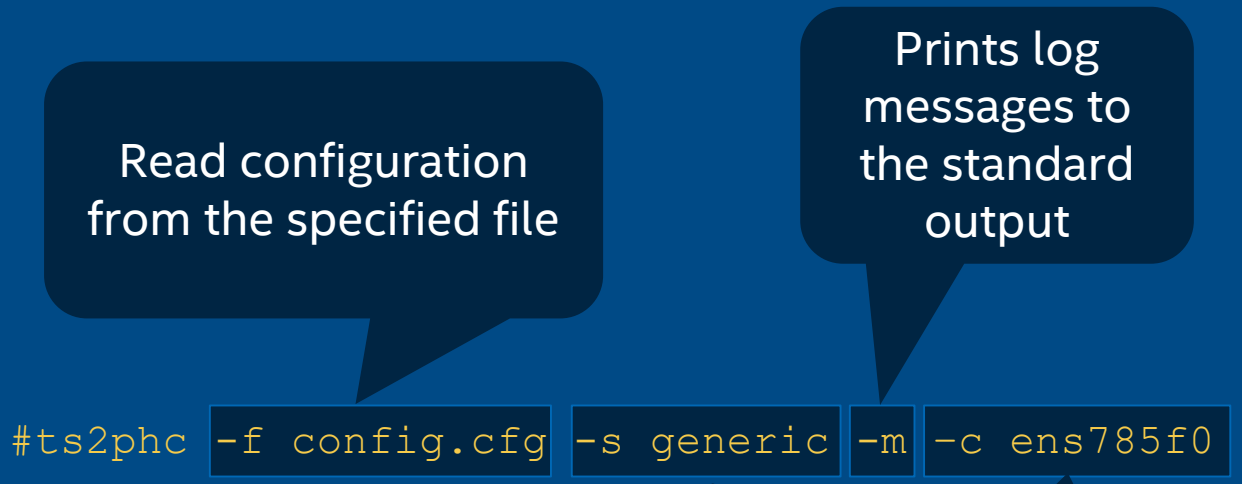
Frequency difference to the master clock (in ppb)

ts2phc

- New tool in linuxptp-3.0
- Synchronizes one or more PHC using external time stamps.
- Can sync PHC to 1pps
- New option – sync to NMEA GPS output (GPRMC)

- https://www.mankier.com/8/ts2phc

ts2phc – command line



phc2sys

- Synchronizes the system clock to the PHC on the NIC
- Can run as:
  - Standalone tool
  - Service
    - options are specified in the /etc/sysconfig/phc2sys
    - Can be started using # systemctl start phc2sys

- https://www.mankier.com/8/phc2sys

phc2sys – command line

Specify the master clock by:
- device (e.g. /dev/ptp0)
- interface (e.g. eth0)
- by name (e.g. CLOCK_REALTIME for the system clock)

Wait until ptp4l is in a synchronized state

```
#phc2sys -s ens260f0 -c CLOCK_REALTIME -w -m
```

Specify the subordinate clock by:
- device (e.g. /dev/ptp1)
- interface (e.g. eth1)
- Name

The default is CLOCK_REALTIME

Print messages to the standard output

phc2sys - output

Synchronization state:
- s0 – servo unlocked
- s1 – clock step
- s2 – servo locked

Delay of the path

```
phc2sys[4649082.619]: CLOCK_REALTIME phc offset  -4717525 s0 freq      +0 delay   1274
phc2sys[4649083.619]: CLOCK_REALTIME phc offset  -4728162 s1 freq  -10632 delay   1273
phc2sys[4649084.620]: CLOCK_REALTIME phc offset        40 s2 freq  -10592 delay   1278
phc2sys[4649085.620]: CLOCK_REALTIME phc offset         7 s2 freq  -10613 delay   1276
phc2sys[4649086.620]: CLOCK_REALTIME phc offset        -5 s2 freq  -10623 delay   1280
phc2sys[4649087.620]: CLOCK_REALTIME phc offset        13 s2 freq  -10606 delay   1277
phc2sys[4649088.621]: CLOCK_REALTIME phc offset       -13 s2 freq  -10628 delay   1278
phc2sys[4649089.621]: CLOCK_REALTIME phc offset        -5 s2 freq  -10624 delay   1290
phc2sys[4649090.621]: CLOCK_REALTIME phc offset         5 s2 freq  -10616 delay   1277
phc2sys[4649091.621]: CLOCK_REALTIME phc offset       -18 s2 freq  -10637 delay   1294
phc2sys[4649092.622]: CLOCK_REALTIME phc offset         7 s2 freq  -10618 delay   1288
phc2sys[4649093.622]: CLOCK_REALTIME phc offset        12 s2 freq  -10611 delay   1286
```

Offset to the master clock

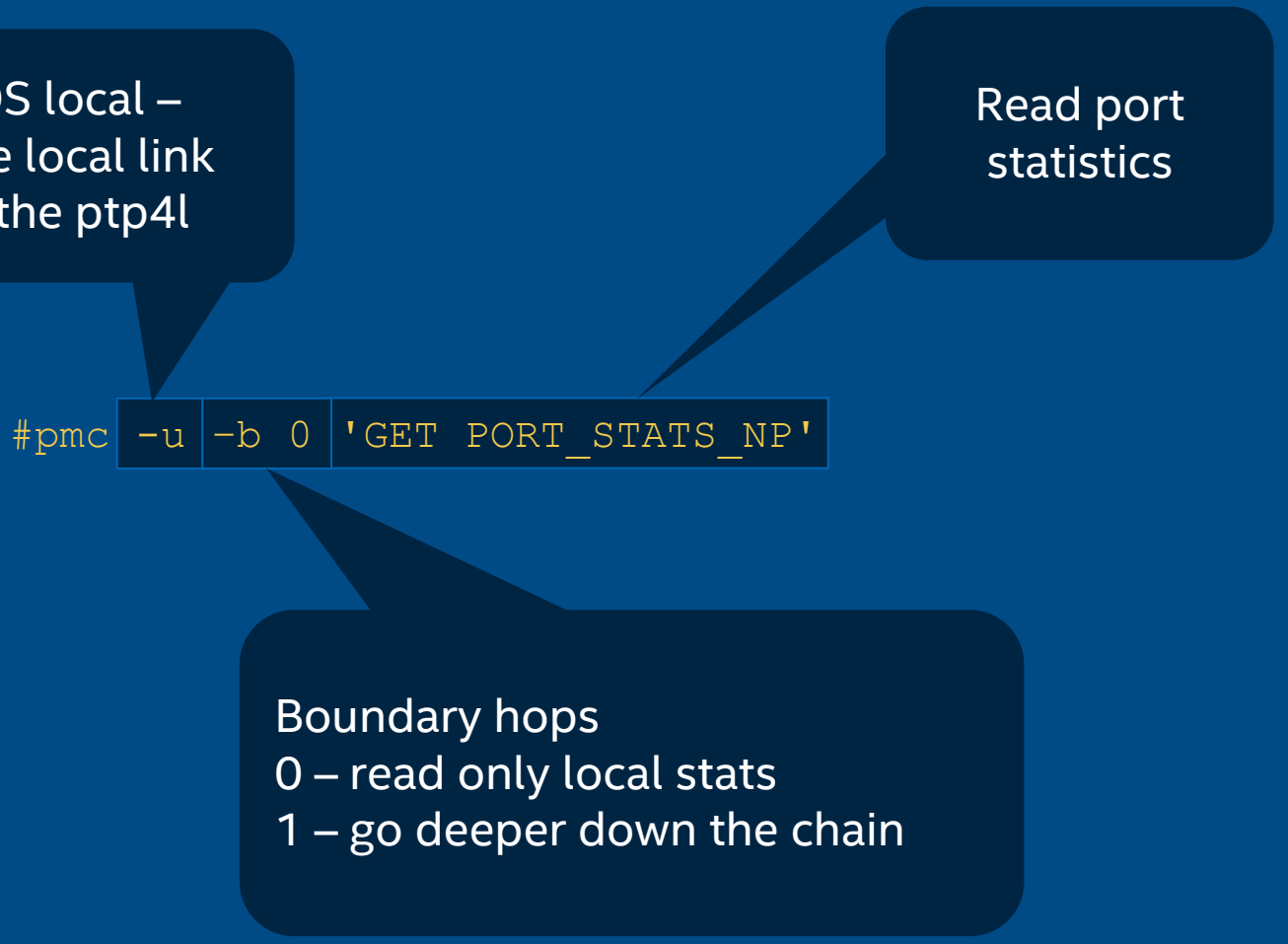
Frequency difference to the master clock (in ppb)

pmc

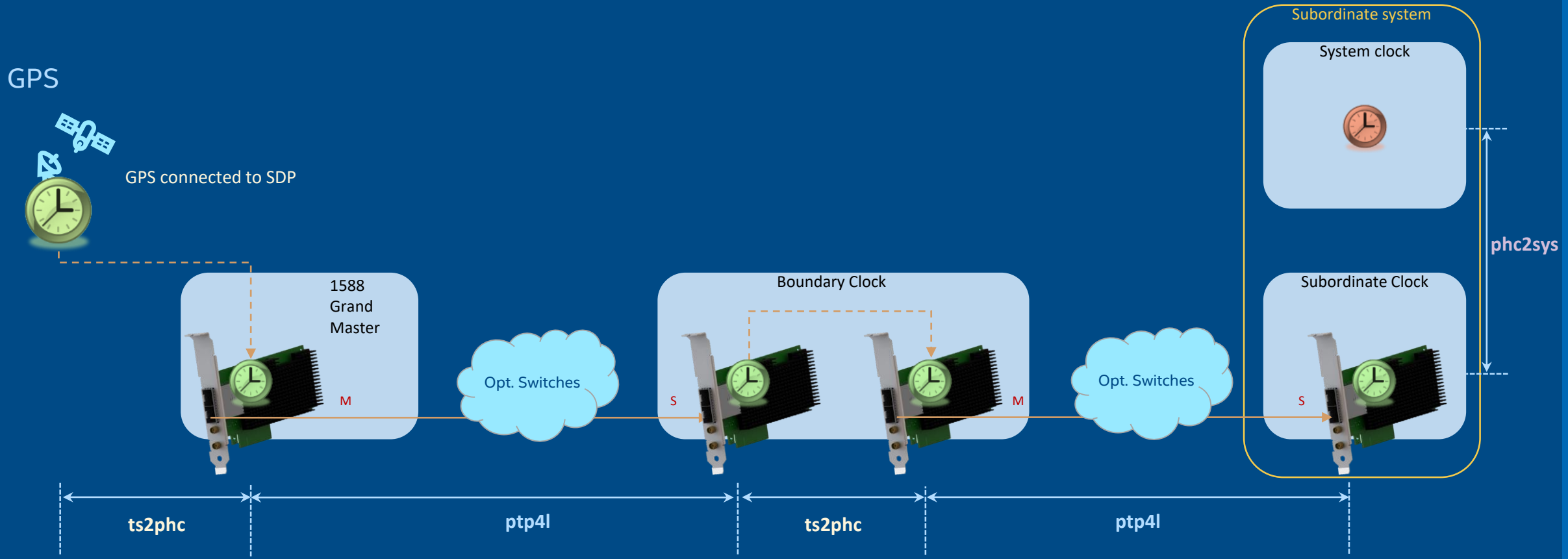
- Management client
- Can obtain additional information from a running ptp4l

- https://www.mankier.com/8/pmc

pmc – command line



Putting it all together



Putting it all together

- MASTER

Config the hw pins (via /sys/class/ptp/ptpX)

```
echo 1 0 > /sys/class/ptp/ptp4/pins/SDP3_2
echo 0 1 > /sys/class/ptp/ptp4/extts_enable
```

```
# ts2phc -f config.cfg -s generic –m –c ens260f0
```

```
# ptp4l -f ptp4l-master.cfg -i ens260f0 -m
ptp4l[254799.658]: assuming the grand master role
```

- SUBORDINATE

```
# ptp4l -f ptp4l-subordinate.cfg -i ens260f0 -m -s
```

```
# phc2sys -s ens260f0 -c CLOCK_REALTIME -w -m
```


intel

Useful links

- https://docs.fedoraproject.org/en-US/fedora/rawhide/system-administrators-guide/servers/Configuring_PTP_Using_ptp4l/
- https://tsn.readthedocs.io/timesync.html
- https://www.mankier.com/8/phc_ctl