NetTimeLogic GMBH

Introduction to PTP

PTP Basics

Content

What is PTP?

- **Precision Time Protocol standardized in IEEE1588-2008 (or IEC61588-Ed.2)**
- **Packet based time synchronization protocol**
  - It describes the mechanisms how to distribute time (phase, frequency and absolute time) over a packet based network (Ethernet)
  - Data and synchronization is using the same network
- **Allows sub-microsecond accuracy**
- **Determines the synchronization hierarchy automatically**

Why PTP?

- **Why PTP and not another protocol**
  - Why better than PPS?
    - Requires a dedicated PPS synchronization network
    - Path delays have to be manually set
    - Does not provide absolute time
  - Why better than IRIG-B?
    - Requires a dedicated IRIG synchronization network
    - Path delays have to be manually set
  - Why better than (S)NTP
    - Lower accuracy than PTP
  - Why better than GPS
    - Requires outdoor antenna

Why PTP?

- **But:**
  - PTP requires hardware assistance
  - Is designed for engineered environments, e.g. all devices shall support PTP

Why PTP?

- **Where is sub-microsecond accuracy required?**
  - Automation and control systems
    - Synchronize multi axis drive systems
    - Synchronize subsystems with cyclic operation
  - Measurement and automatic test systems
    - Correlation of decentral acquired values
    - Time stamping of logged data
  - Power generation, transmission and distribution systems
    - Control of switching operations
    - Reconstruction of network activities and events
  - Etc.

Why PTP?

- **In Power applications:**
  - Components of the power grid have to be protected from critical load situations and turned off
  - Protection switching guarantees high service availability
  - U/I is measured at all critical points within the grid at precise time points and in a high rate
    - To monitor the network
    - To predict critical load situations
    - To protect the network from overload
    - To measure delivered/consumed power

History and Future

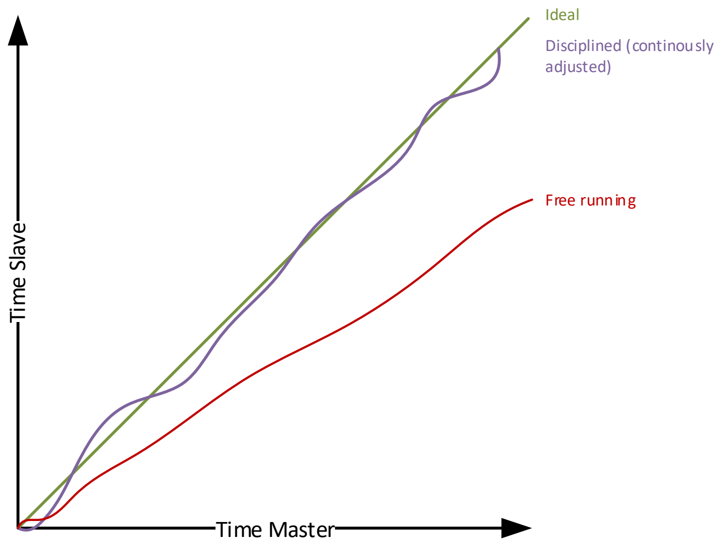
- **First version IEEE1588-2002**
  - Approved 12.09.2002
  - Also known as PTPv1
  - Adopted by IEC as IEC61588-Ed.1 in 2004
  - Not widely deployed
- **Second version IEEE1588-2008**
  - Approved 12.09.2002
  - Also known as PTPv2
  - Adopted by IEC as IEC61588-Ed.2 in 2008
  - Widely deployed
  - Current version (and version of IP cores)

History and Future

- **Third version IEEE1588-2018?**
  - Not approved yet, planed for 2018
  - Also known as PTPv2.1
  - Not deployed yet
  - Compatible with PTPv2, only optional new features:
    - Security extensions
    - Sub-nanosecond accuracy
    - Etc.

How does it work?

- A free running clock has not the same frequency and the frequency will slightly change and the phase is unknown
- Time is never ideal therefore has to be disciplined
  - Interval of adjustment is crucial



How does it work?

- **PTP runs in TAI Time**
  - Start 1.1.1970 Midnight => Second 0
  - 48bit Seconds 32bit Nanoseconds => overflow in 8‘925‘512 years
- **PTP provides UTC offset and Leap second to convert time between TAI and UTC in the end node**

How does it work?

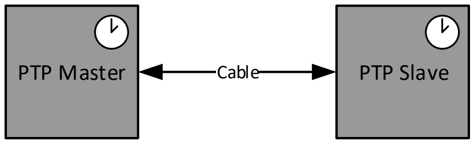
- **To have synchronous time the Frequency and Phase have to be corrected**
- **Frequency Correction**
  - The Slave‘s oscillator does not have exactly the same frequency as the Master
  - The Slave‘s oscillator frequency varies over time (due to environmental conditions)
- **Phase Correction**
  - The Slave and Master don’t start at the same time
  - The Master makes a jump in time

How does it work?

- **The standard defines how Mater and Slaves communicate and where timestamps are generated and how the differences are calculated**
- **The standard says nothing about how to correct the Slave**
  - Hard set/Time Jump?
  - Smoothed out?
  - Via a PI Servo Loop?
  - It is up to the vendor and its requirements how the synchronization shall be done

How does it work?

- The simplest setup consists of two PTP nodes
- To simplify the calculations we have a direct connection between them (just a cable)



How does it work?
Determining the topology

1. **All nodes listen for so called «Announce» message**
   - An Announce message contains quality information of the Clock (Class, Priorities and Qualities) which sends it
2. **When no «Announce» messages was received for a defined interval the nodes become Master and start to send their own «Announce» messages**
3. **If a node receives an «Announce» message which is better by its quality, the node stops to send «Announce» messages and becomes Slave**

How does it work?
Determining the topology

4. **If a node receives an «Announce» message which is worse by its quality, the node stays in Master and continues to send «Announce» messages in a defined interval**
   - When the network has determined the best node in the network, this is the only one sending Announce messages => 1 Master, N Slaves
   - This algorithm runs all the time, means if another node becomes better or the current Master gets worse than another node the topology changes

- **This algorithm is called Best Master Clock Algorithm (BMCA)**

How does it work?
BMCA

- **The comparison is based on the following attributes in the respective order:**
  1. Priority1: a configurable clock priority
  2. ClockClass: a clock ‘s traceability
  3. ClockAccuracy: a clock’s accuracy
  4. OffsetScaledLogVariance: a clock’s stability
  5. Priority2: a configurable second order clock priority
  6. ClockIdentity: a clock’s unique identifier (the tie-breaker if all other attributes are equal)
- **Based on this comparison a state decision is taken the port state is set to either MASTER, SLAVE, or PASSIVE**

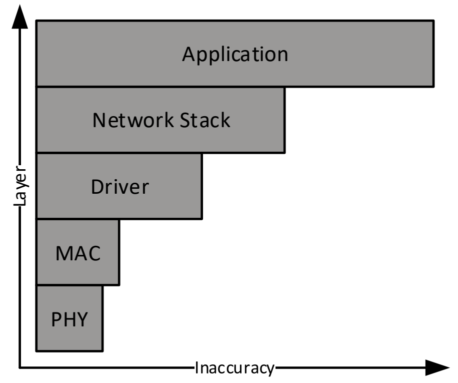
How does it work?
BMCA

- **This was only half the part of the BMCA, it only determined which is the Best Clock in the network, however the BMCA also checks other values to determine the network topology**
  1. StepsRemoved: over how many hops the frame came
  2. SenderPortIdentity: a port’s unique identifier (the first tie-breaker if all other attributes are equal)
  3. ReceiverPortIdentity : a port’s unique identifier (the tie-breaker if all other attributes are equal)
  - This is needed e.g. in a multiport PTP device which receives Announce frames over multiple ports, which one is the one to synchronize to

How does it work?

Taking Timestamps

- **The accuracy of the PTP system heavily depends on the accuracy of the timestamps**
- **There are several levels where timestamps can be taken**
  - Accuracy differs, the higher in the Network stack the worse
  - The timestamp point for Ethernet is the detection of the Start of Frame Delimiter (SFD) on the Cable
  - Most implementations take timestamps between the MAC and PHY, which needs hardware support



How does it work?
Taking Timestamps

NetTimeLogic GMBH

- **When Timestamping is done above the PHY two values have to be compensated for**
  - The RX PHY delay has to be subtracted from the timestamp (too late)
  - The TX PHY delay has to be added to the timestamp (too early)
- **RX and TX PHY delays are not the same and have to be handled separately**
  - Otherwise asymmetries are introduced
- **PHYs introduce additional jitter on the timestamps due to FIFOs and clock domains**

How does it work?
Adjust the frequency

1. **The node which is Master sends a so called «Sync» messages and takes a timestamp (T1)**
   - A Sync message contains the timestamp when the Sync message was sent (T1)
   - If a node can insert the sending timestamp (T1) into the Sync message on the fly this is called a «OneStep» clock
2. **If the node can not insert the sending timestamp (T1) on the fly, it will send a so called «FollowUp» message**
   - A FollowUp message contains the sending timestamp of the Sync message (T1)

How does it work?
Adjust the frequency

- In this case it puts either an estimate of the sending timestamp into the Sync or sets it to 0
- This is called a «TwoStep» clock

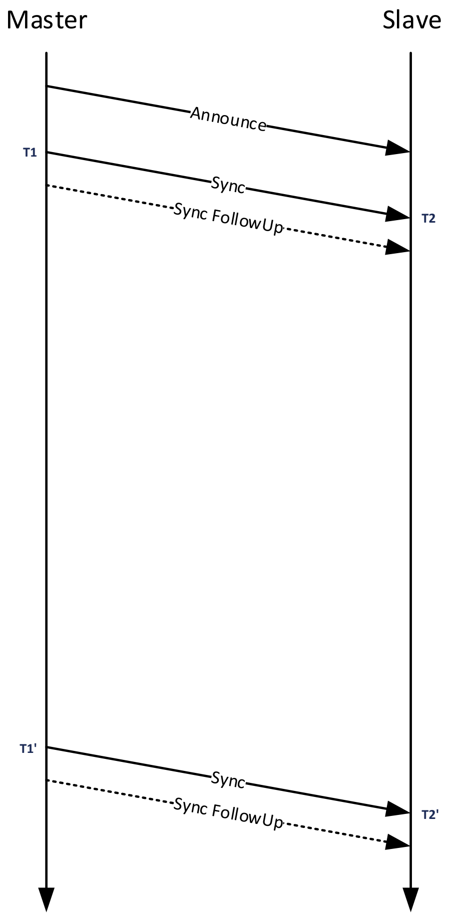
3. **The Slave node takes a timestamp when it receives the «Sync» message (T2)**
   - This timestamp is stored for further
   - No timestamp is taken when a FollowUp is received
4. **The master repeats the sending of a «Sync» and optional «FollowUp» in a defined interval**
   - Timestamps are taken on both sides again (T1' & T2' …)

How does it work?
Adjust the frequency

- **After two «Sync» messages the Slave can calculate the frequency difference to its master**
  - This is called «Drift»
  - The calculation is a following:

$$Drift = \frac{(T2' - T2) - (T1' - T1)}{(T1' - T1)}$$

  - This Drift can then be adjusted to align the frequency of the Slave with the one of the Master



How does it work?
Adjust the phase

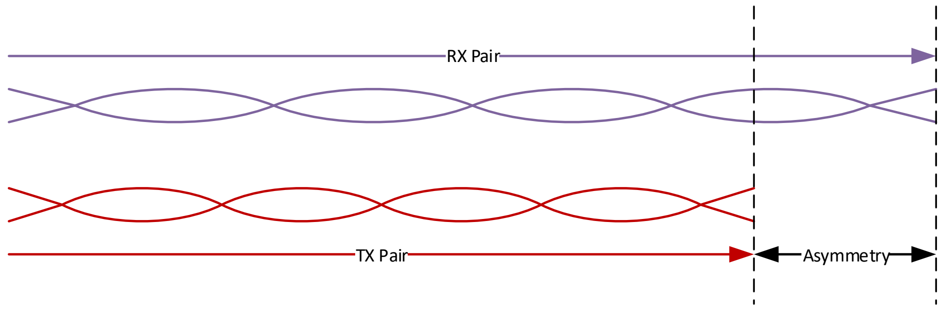
- **To adjust the phase the timestamps of the sending (T1) and receiving (T2) of the «Sync» are used**
- **Unfortunately these two timestamps (T1 & T2) are not enough to calculate the phase**
  - If the Slave just substracts T2 from T1 and adjusts this the two nodes would still be off.
- **The delay which it takes from the sending to the receiving of the «Sync» needs to be calculated first**
  - There are two modes to measure the delay: End to End (E2E) and Peer to Peer (P2P)

How does it work?
Calculate the delay

- **The E2E delay mechanism measures the delay from the Slave to the Master**
- **The P2P delay mechanism measures the delay between two nodes only independent of their states**
  - We will see later in detail how this works for a larger topology
- **Both delay mechanisms assume a symmetrical transmission delay**

How does it work?
Calculate the delay

- **Different twist rate of twisted line pairs leads to delay skew (difference between propagation delays of transmit and receive lines)**
  - CAT 5/6: allows up to 50 ns per 100 meter cable (IEC 11801)
  - CAT 7: allows up to 30 ns per 100 meter cable (IEC 11801)
  - Real cables are typically better than IEC 11801 allows



How does it work?
Calculate the delay (E2E)

1. **The Master sends a «Sync» and an optional «FollowUp» message**
   - Timestamps are taken on both sides T1 & T2
2. **The Slave sends short after the reception of the «Sync» message a so called «DelayReq» message and takes a timestamp (T3)**
   - The DelayReq does not contain any timestamp
   - It takes a timestamp (T3) when it sent the DelayReq and stores it

How does it work?
Calculate the delay (E2E)

3. **The Master receives the «DelayReq» message and takes a timestamp (T4)**
   - It takes a timestamp (T4) when it received the DelayReq and stores it
4. **The Master sends a so called «DelayResp» message**
   - The DelayResp message contains the timestamp when the DelayReq was received (T4)
5. **The Slave receives the «DelayResp» message**
   - No timestamp is taken when a DelayResp is received
   - Now it can calculate the delay between the nodes

How does it work?
Calculate the delay (E2E)

- The Master does not initiate any delay measurement
- The timestamps used for this measurement must come from the synchronized clock.

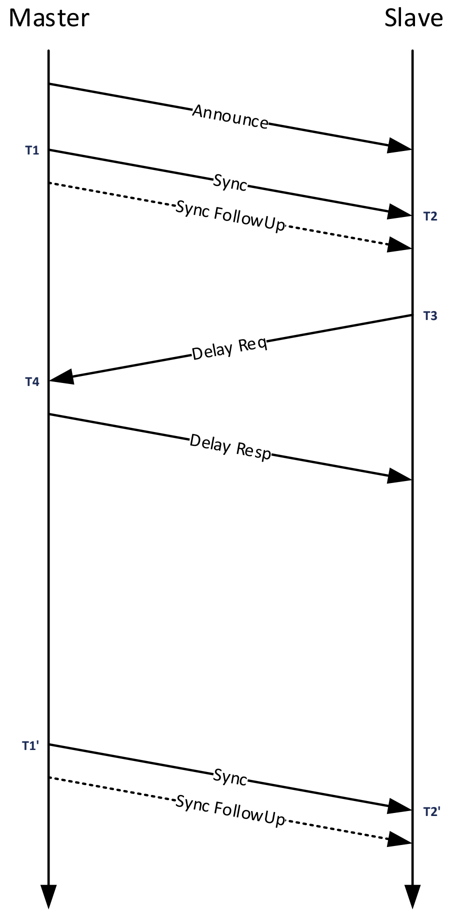
How does it work?
Calculate the delay (E2E)

- **After the Slave has received all frames it can calculate the delay**
  - The calculation is a following:

$$Delay = \frac{(T4 - T1) - (T3 - T2)}{2}$$

  - This delay is then used for adjusting the phase of the Slave
- **This measurement is repeated in a defined interval**

E2E



How does it work?
Calculate the delay (P2P)

1. **The Slave sends a so called «PDelayReq» message and takes a timestamp (T3)**
   - The PDelayReq does not contain any timestamp
   - It takes a timestamp (T3) when it sent the PDelayReq and stores it
2. **The Master receives the «PDelayReq» message and takes a timestamp (T4)**
   - It takes a timestamp (T4) when it received the DelayReq and stores it

How does it work?
Calculate the delay (P2P)

3. **The Master sends a so called «PDelayResp» message and takes a timestamp (T5)**
   - It takes a timestamp on sending of the PDelayResp (T5)
   - The PDelayResp message contains either the timestamp when the PDelayReq was received (T4) in "TwoStep" mode or for "OneStep" mode the delta between the timestamps when sending the PDelayResp (T5) and receiving the PDelayReq (T4) which is inserted on the fly

How does it work?
Calculate the delay (P2P)

4. **If the node can not insert the delta (T5-T4) on the fly, it will send a so called «PDelayRespFollowUp» message**
   - A PDelayRespFollowUp message contains the sending timestamp of the PDelayResp message (T5)
5. **The Slave receives the «PDelayResp» message and takes a timestamp (T6)**
   - Now it can calculate the delay between the nodes
   - No timestamp is taken when a PDelayRespFollowUp is received

How does it work?
Calculate the delay (P2P)

NetTimeLogic GMBH

- **There are 3 options how to get the timestamps from the responder to the requestor:**
  - PDelayResp(T5-T4), «OneStep»
  - PDelayRespFollowUp(T5-T4), «TwoStep»
  - PDelayResp(T4)
    PDelayRespFollowUp(T5) , «TwoStep»
- **In this example the Slave measured the delay to the Master, the same is done also from the Master to the Slave (between all peers)**
- **The timestamps used for this measurement can come from a different clock than the synchronized one.**

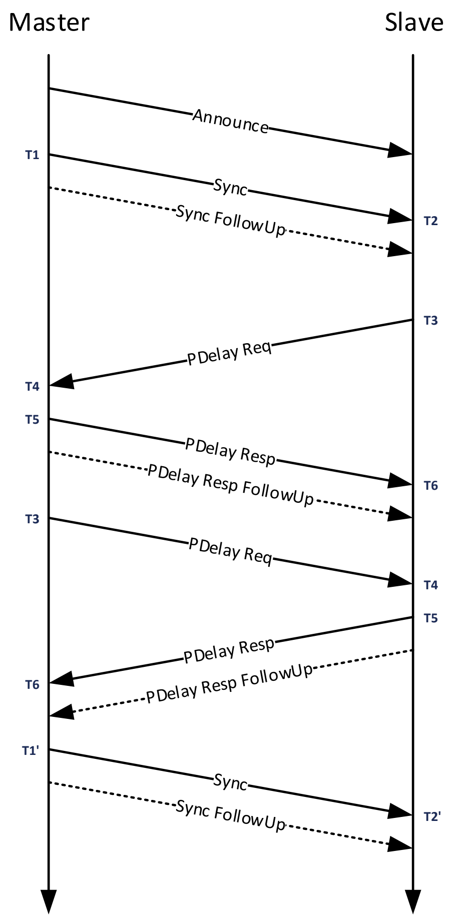
How does it work?
Calculate the delay (P2P)

- **After the Slave has received all frames it can calculate the delay**
  - The calculation is a following:

$$Delay = \frac{(T6 - T3) - (T5 - T4)}{2}$$

  - This delay is then used for adjusting the phase of the Slave
- **This measurement is repeated in a defined interval**

P2P



How does it work?
Adjust the phase

- **Now that the Slave has calculated the delay it can calculate the phase**
  - This is called «Offset»
  - The calculation is a following:
    $$Offset = (T2 - T1) - Delay$$
  - This offset is then used for adjusting the phase of the Slave
  - It doesn’t matter which Delay Mechanism is used in this example

How does it work?
Synchronizing

- Topology changes are considered every «Announce» interval
- Offset and drift are adjusted every «Sync» interval
- Delays are calculated every «Delay» interval
- «Announce», «Sync» and «Delay» intervals don’t have to be (and often are not) the same

How does it work?
PTP Nodes

- **A PTP network (or network in general) does not only contain two nodes and therefore also more PTP node types**

- **Ordinary Clock (OC)**
  - A PTP node with only one port
  - Can be Master or Slave
  - If it is the best clock according to the BMCA it will act as Master otherwise as Slave

How does it work?
PTP Nodes

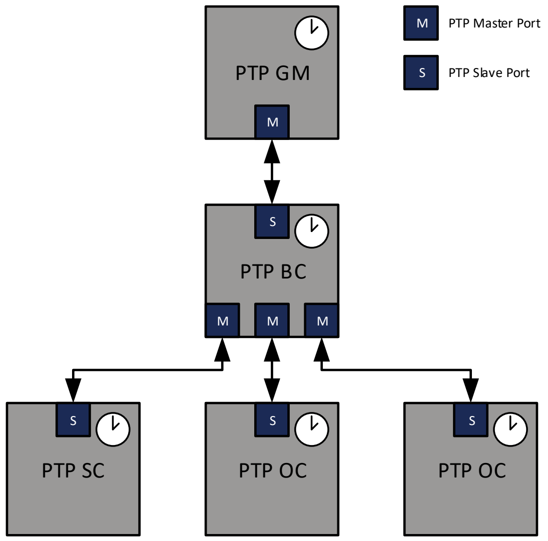
- **Grandmaster Clock (GM)**
  - A OC with either an external time source (GPS…) or a very high accuracy time (ATOM)
  - Can only be Master
  - If it is the best clock according to the BMCA it will act as Master otherwise it will go in a passive State
- **Slave Only Clock (SO)**
  - A PTP node with only one port
  - Can only be Slave
  - If no Master capable device is in the network it will be just free running

How does it work?
PTP Nodes

- **Boundary Clock (BC)**
  - A PTP node with more than one port
  - Can be Master or Slave
  - Normally a Switch
  - PTP frames are not forwarded through the Switch, the BC is source and Sink for all PTP frames
  - Each port has its own state
  - Slave on one port and Master on all other ports, or Master on all ports determined by the BMCA
  - The BC synchronizes itself to a Master on its Slave port and distributes the time on all its Master ports

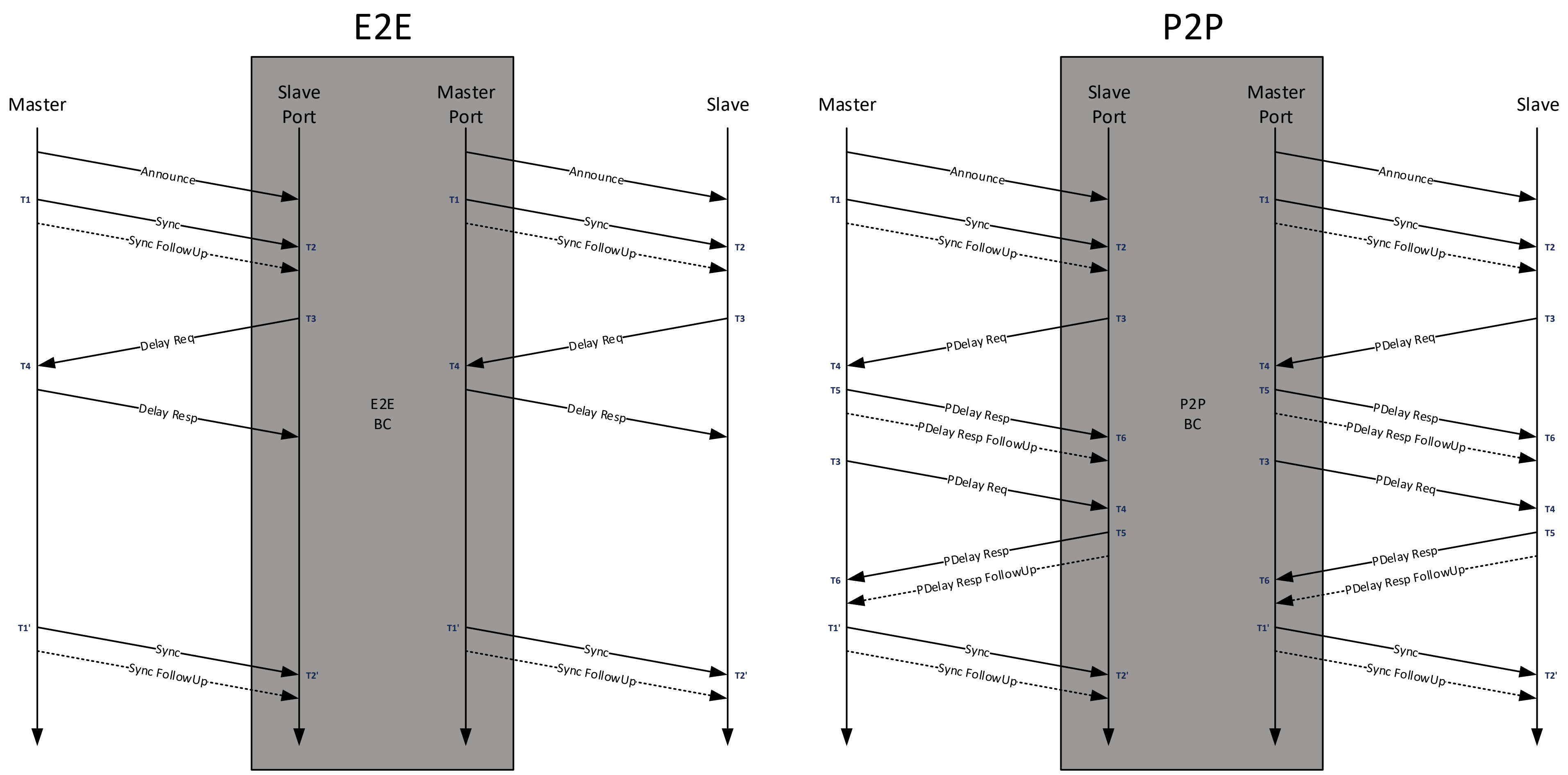
How does it work?
PTP Nodes

- On each port a different delay mechanism and frame rates can be used
- Normal Switches (without PTP) have an non-deterministic forwarding delay which has a really bad influence on the accuracy



How does it work?

PTP Nodes (BC)

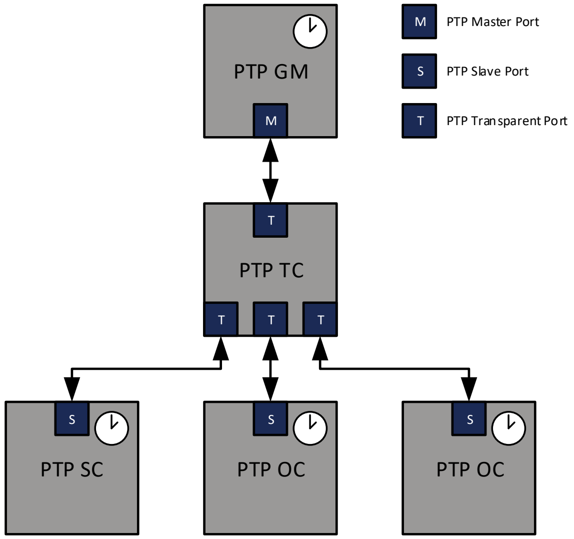


How does it work?
PTP Nodes

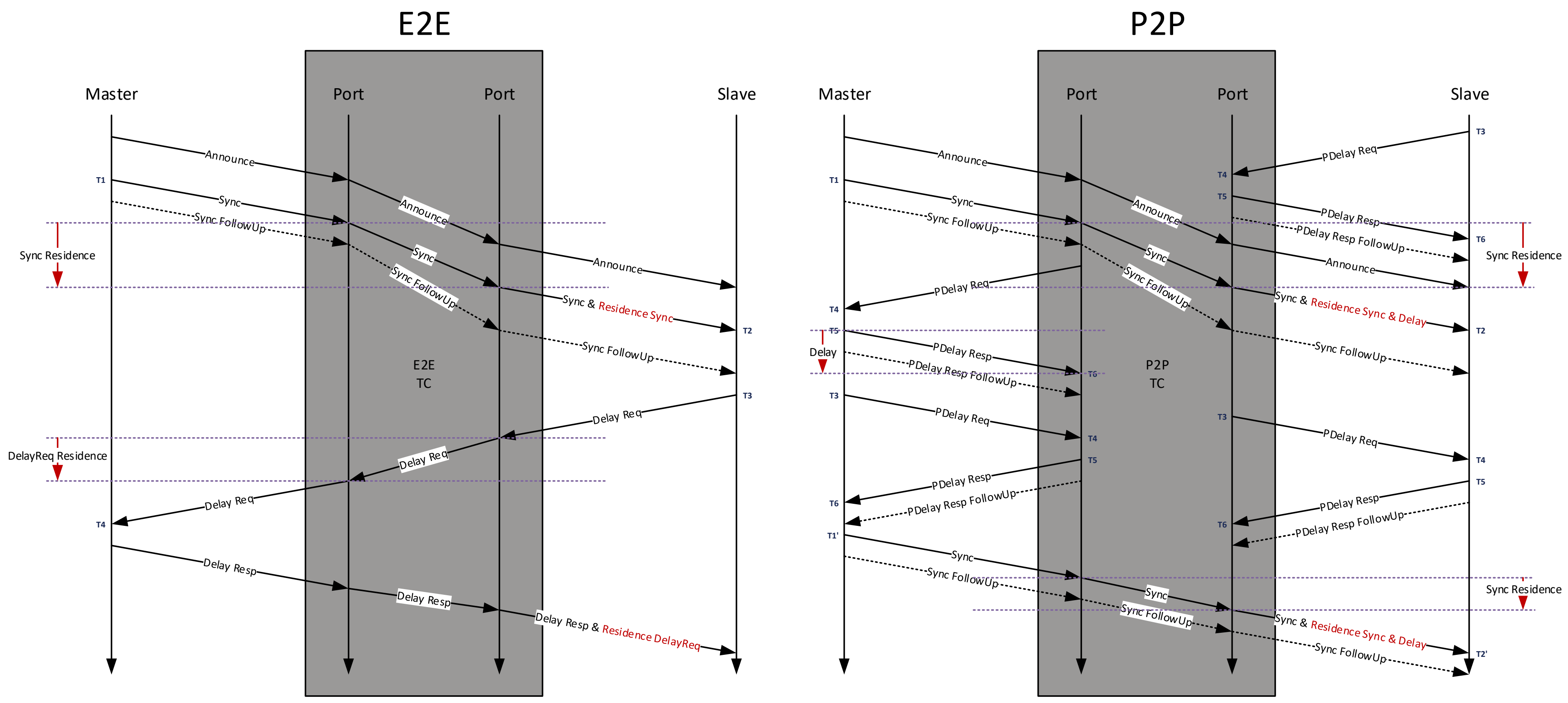
- **Transparent Clock (TC)**
  - A PTP node with more than one port
  - Stateless, does not take part in the BMCA
  - Normally a Switch
  - PTP frames are forwarded through the Switch (except P2P messages) and their resident time is added to a so called «correction field» of the Sync and DelayReq if in «OneStep» mode and in the corresponding «FollowUp» or non-time-critical messages accordingly if in «TwoStep» mode
  - On each port the same delay mechanism has to be used

How does it work?
PTP Nodes

- The TC might syntonize (frequency align) itself to the Master to calculate the resident times
- When in P2P mode the TC measures the delay on all its ports and adds the delay of the corresponding port to the correction field where the Sync is received



How does it work?
PTP Nodes (TC)



How does it work?
PTP Nodes (TC)

- **P2P TC:**
  - All link delays are measured on a peer to peer basis.
  - Sum of switch residence time and link delay along the path is reported to the Slave.
- **E2E TC:**
  - Delay measurement end to end between slave and master.
  - Sum of switch residence time along the path is reported to the slave.
- **TC can also be «OneStep» or «TwoStep»**
  - Residence time either on the fly added to Sync or later in the Follow Up

How does it work?
PTP Nodes (TC)

- «Correction Fields» of the «Sync» and «FollowUp» have to be added together at the Slave and T2 corrected accordingly.

BC vs. TC

- BC
  - + Different Delay mechanisms and message rates on each port
  - + Can lower the network load for E2E
  - + Can take over the Master rule
  - - Higher complexity, requires a PTP stack
  - - Cascaded PI Servo Loops (e.g. bad for Daisy-Chain)
  - - No fast topology changes possible

BC vs. TC

- **TC**
  - + No cascaded PI Servo Loops
  - + Fast topology changes possible
  - + Lower complexity (no BMCA, no Synchronization)
  - + No PTP stack required
  - - Requires the same Delay mechanisms and message rates on each port
  - - Can not take over the Master rule

How does it work?
PTP Nodes

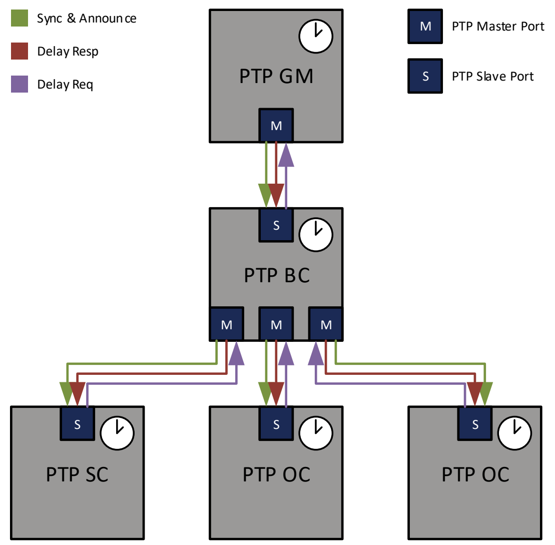
- **Hybrid Clock (HC)**
  - A combination between a Transparent Clock and Ordinary Clock
  - Often used in Daisy-Chains
  - Often a 3 port TC, two external and one internal where the OC is connected and is the uplink to the CPU.
- **Management Node (MN)**
  - Does not take part in the synchronization or BMCA
  - Can read and write parameters in the other PTP nodes via Management messages (which is often not supported due to security reasons)

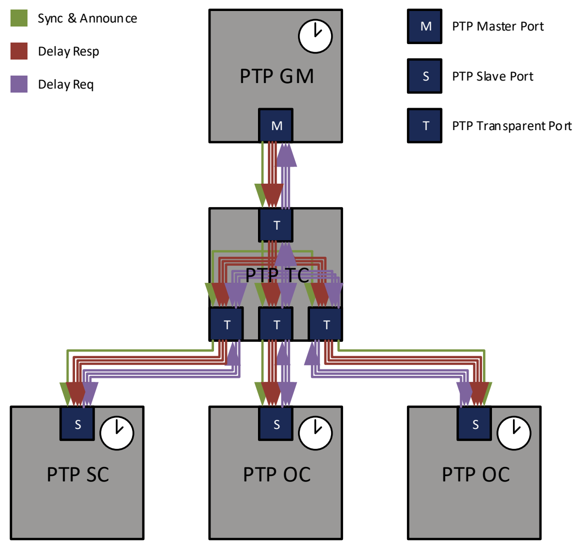
How does it work?
OneStep

- **«OneStep» operation is not trivial**
  - Frame must already partly sent and can not starve
  - Fields have to be completely received before modifications can be done due to endianness
  - Ethernet CRC has to be recalculated and inserted
  - UDP checksum can be either cleared (IPv4) or has to be fixed with additional bytes at the frame end (IPv6)
  - Layer violations according to OSI model

E2E vs. P2P

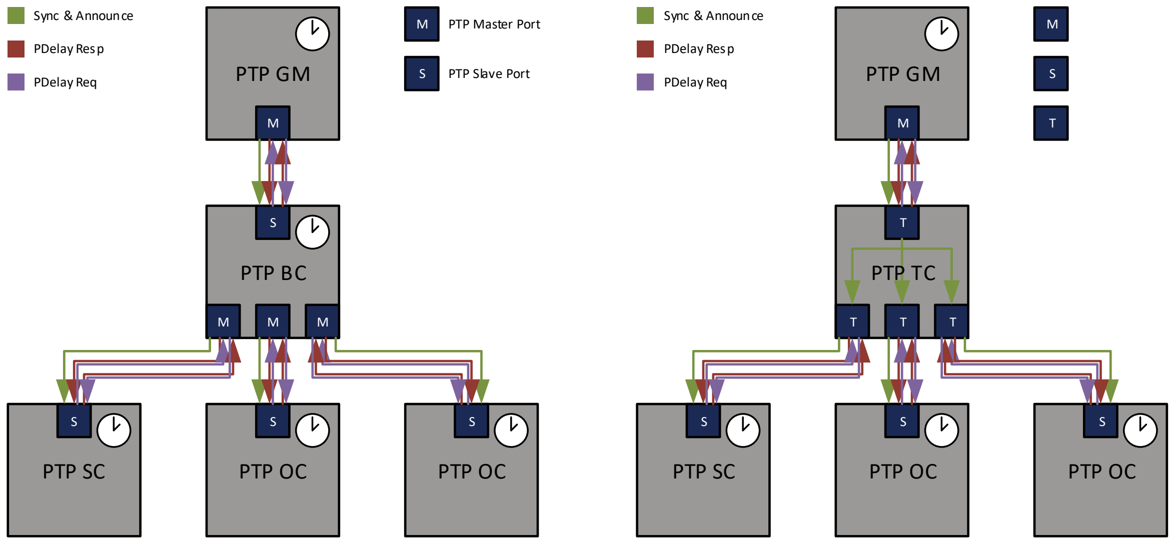
- **E2E**
  - Always from the Slave port to a Master port
  - In case of a TC all nodes see the Delay messages from all other nodes, doesn't scale well





E2E vs. P2P

- **P2P**
  - Between every two neighbor ports
  - Each node sees only the Delay messages of his neighbor, scales well



E2E vs. P2P

- **E2E**

  + Works with legacy Switches (no PTP support)

  - High network load when TCs are used

  - Doesn’t scale well

  - Can not handle topology changes seamless, has to measure the path to the new master first

  - Most industrial profiles do not support E2E

E2E vs. P2P

- **P2P**
  - + Low network load when TCs or BCs are used
  - + Scales well
  - + Can handle topology changes seamless, all delays to all neighbors are pre measured
  - + Easier to combine with HSR/PRP
  - + Most industrial profiles support P2P
  - - Doesn’t works with legacy Switches (no PTP P2P support)

Profiles

- **PTP comes in different flavors (Profiles), depending on the environment it shall be used in.**
- **Profiles define communication medium mappings (Ethernet, Profinet, etc.), message rates, the delay mechanisms, default values of datasets and sometimes much more:**
  - Default Profile uses either Layer 2 or 3 with Multicast and either E2E or P2P Delay mechanism
  - Power Profile uses Layer 2 with Multicast and P2P Delay mechanism and additional TLV
  - Utility Profile uses also Layer 2 with Multicast and P2P Delay mechanism in combination with HSR/PRP

Profiles

- There are many other Profiles with other feature sets and mappings.
    - Telco Profiles
    - Enterprise Profile
    - 802.1AS Standard/Profile
    - Etc.
- Some Profiles are subsets of the Default Profile and compatible, some are supersets and therefor incompatible with other Profiles.
- This makes interoperability difficult.

Profiles

- There are many other Profiles with other feature sets and mappings.
  - Telco Profiles
  - Enterprise Profile
  - 802.1AS Standard/Profile
  - Etc.
- Some Profiles are subsets of the Default Profile and compatible, some are supersets and therefor incompatible with other Profiles.
- This makes interoperability difficult.

Profiles
Communication Scheme

NetTimeLogic GMBH

- **PTP over UDP/IPv4**
  - UDP Port 319 for Sync, DelayReq, PDelayReq & PDelayResp messages (time critical messages)
  - UDP Port 320 for all other messages (non critical messages)
  - IP Addr. 224.0.0.107 for PDelayReq & PDelayResp & PDelayRespFollowUp messages
  - IP Addr. 224.0.1.129 for all others
  - Ethertype 0x0800 for IP

Profiles
Communication Scheme

- **PTP over UDP/IPv6**
  - UDP Port 319 for Sync, DelayReq, PDelayReq & PDelayResp messages (time critical messages)
  - UDP Port 320 for all other messages (non critical messages)
  - IP Addr. FF02:0:0:0:0:0:0:6B for PDelayReq & PDelayResp & PDelayRespFollowUp messages
  - IP Addr. FF0x:0:0:0:0:0:0:181 for all others
  - Ethertype 0x0800 for IP

Profiles
Communication Scheme

- **PTP over 802.3**
  - MAC Addr. 01-80-C2-00-00-0E for PDelayReq & PDelayResp & PDelayRespFollowUp messages
  - MAC Addr. 01-1B-19-00-00-00 for all others
  - Ethertype 0x088F7 for PTP

Profiles
Management Scheme

- **PTP foresees a Management mechanism where any node can GET and SET parameters in any other node**
  - Via Management messages
- **Each GET or SET will be acknowledged**
  - Either OK or NOK
- **Mostly disabled due to security reasons**
- **Alternatively a SNMP MIB can be provided**

Profiles
Domains

- **PTP Networks are divided into Domains**
- **Each Domain has its own Master**
- **Multiple domains could in principle be run over a single Physical network**
  - This is highly not recommended
  - Most devices do not support multiple domains

Accuracy

- **To sum things up, the achievable accuracy depends on:**
    - Timestamp accuracy
    - Clock stability and adjustment steps
    - Sync and Delay intervals
    - Clock control loop characteristics
    - Drift compensated clocks (i.e. adjusted time base in Master and Slave clocks)
    - The communication channel symmetry (i.e. same delay in both directions and constant over a longer period of time)

Questions

NetTimeLogic GMBH

Thank you!

Questions?

NetTimeLogic GmbH
www.nettimelogic.com
contact@nettimelogic.com