Synchronizing Linux Clock and Fieldbus Controllers

Gerhard Engleder

eg@keba.com

KEBA

Automation by innovation.

KEBA – Industrial Automation

- OSADL gold member
- PLCs and KeTop’s for
  - Injection molding
  - Painting
  - Welding
  - ...

KEBA

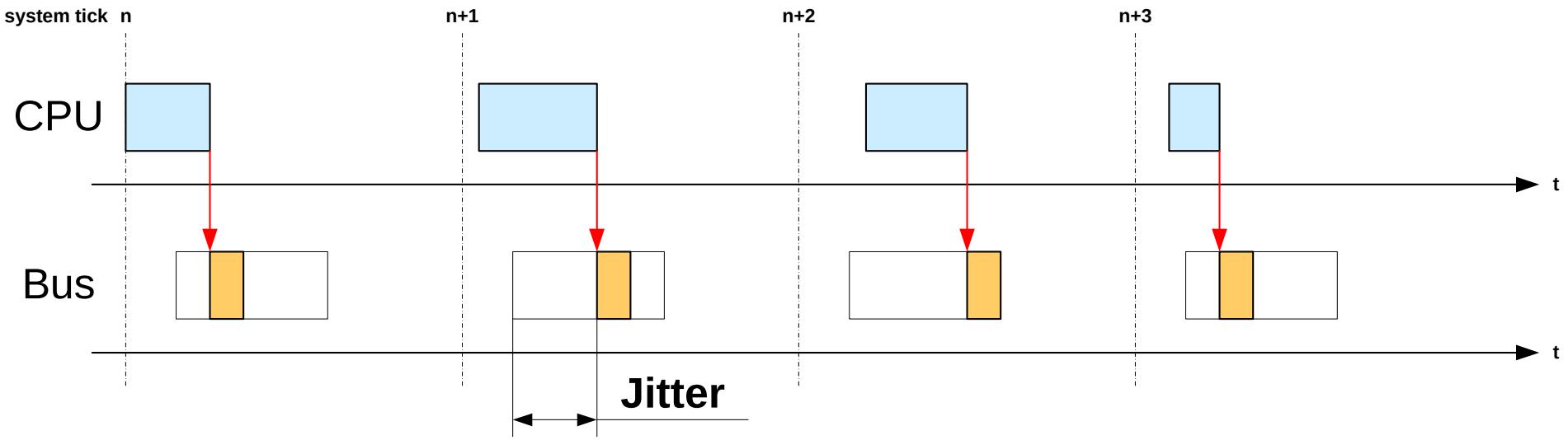
Automation by innovation.

Fieldbus

- real-time communication
- low latency
- low data rate
- isochronous
- deterministic
- application:
  - control loop
  - motion control
  - ...

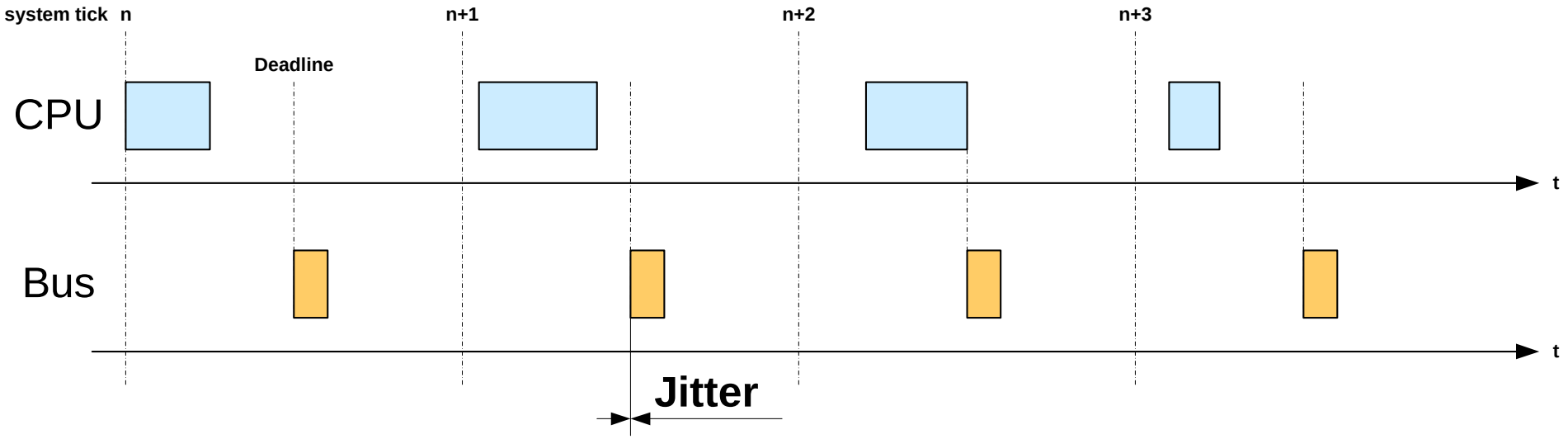
Software triggered communication

- **Software triggers transmission of datagrams directly.**
  - **e.g. register access TX_EN**
- **Transmission is affected by software jitter.**
- **CANopen (SJA1000), PROFINET-IO RT (NIC)**



Hardware triggered communication

- **Transmission of datagrams is triggered by hardware.**
  - **e.g. signal line or hardware clock**
- **Jitter of transmission is minimal.**
- **Sercos III (Sercon), EtherCAT (i210, KEBA ECM)**



Problem

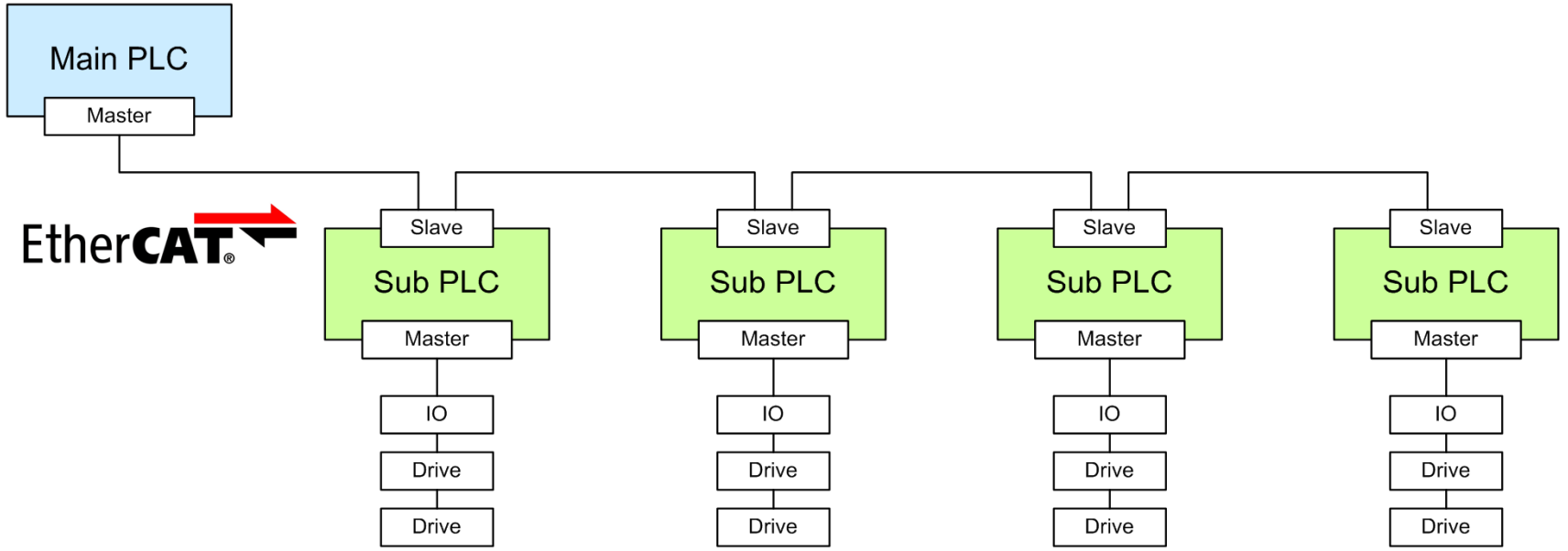
Hardware triggered communication requires synchronous operation of hardware and software.

- **How can synchronous operation of software and hardware be implemented?**
- **Which time source is the reference? OS timer or fieldbus controller?**

Slave synchronization

- **Synchronous operation of multiple PLCs is required.**
- **PLCs are synchronized over a common fieldbus.**

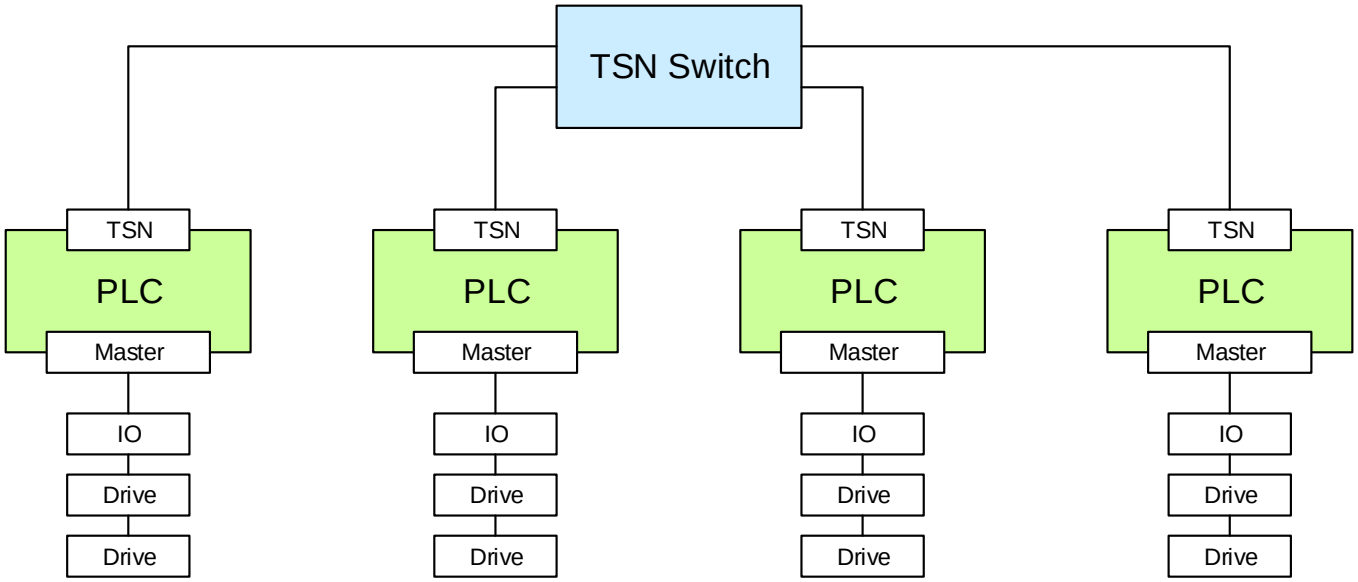
=> The common fieldbus is the reference time source.



Time Sensitive Networking

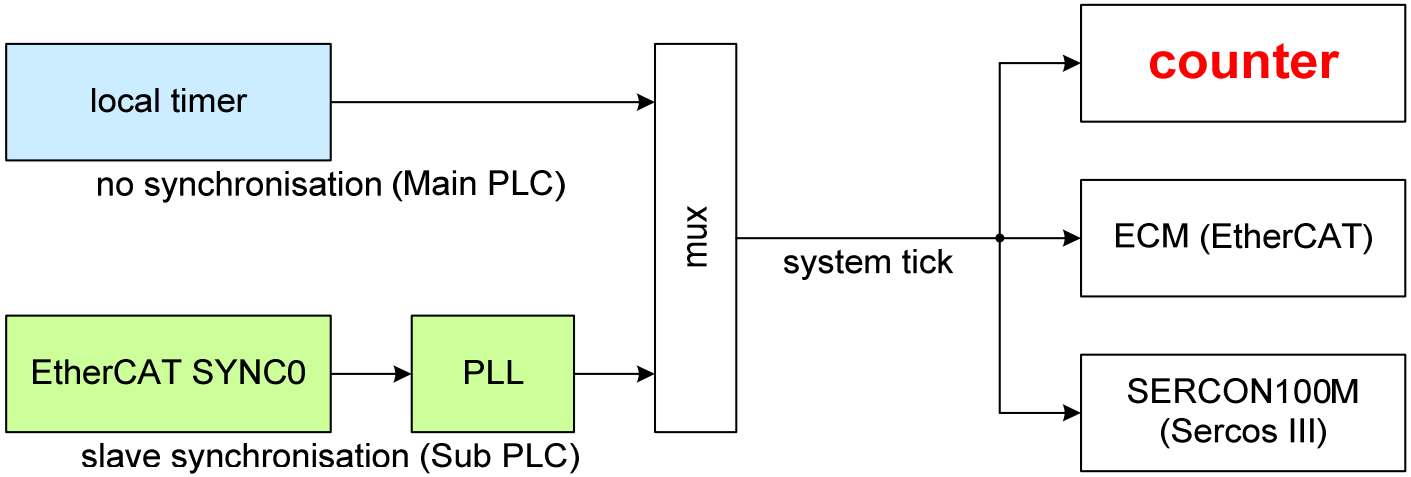
- **Synchronous operation of multiple PLCs is required.**
- **PLCs are synchronized over IEEE 1588.**

=> The TSN network is the reference time source.



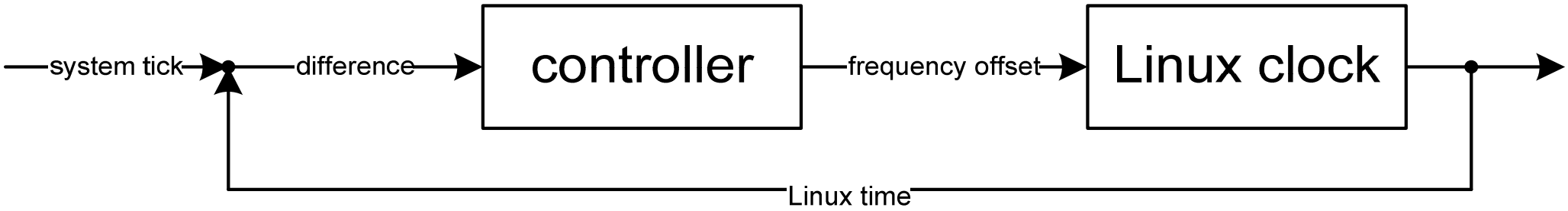
Solution (1)

- **CLOCK_MONOTONIC is synchronized to system tick (hardware signal).**
- **Counter register measures the elapsed time since the last system tick.**
- **Interrupt is eliminated.**



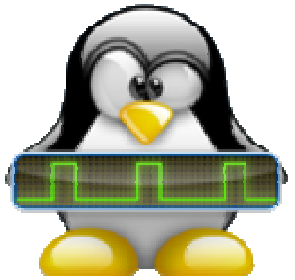
Solution (2)

- **Difference between system tick and CLOCK_MONOTONIC is measured (modulo operation).**
- **Difference is input to controller of frequency offset.**
- **Frequency offset accelerates or decelerates Linux clock (±500 ppm).**



Solution (3)

- Linux PPS subsystem is used.
  - Pulse Per Second
  - Reference time sources: GPS, DCF77, ...
- Provides access to frequency offset.
- PPS and NTP client are mutually exclusive.
- Modified to „PPMS“.
  - Pulse Per MilliSecond
  - difference measurement every millisecond for quick synchronization



Advantages

- **Accuracy is at least one order of magnitude lower than software jitter, ~ ±500 ns.**
- **CLOCK_MONOTONIC is used as expected by CoDeSys and cyclictest.**
- **Slave synchronization is supported.**
- **Only minimal kernel modifications are needed.**
- **Synchronization is optional.**

Drawback

There is only one drawback:

- **PPS and NTP client are mutually exclusive.**

  => NTP cannot be used for the synchronization of CLOCK_REALTIME.

- **CLOCK_REALTIME + offset = CLOCK_MONOTONIC**

  => Both clocks share the same frequency offset.

- **Two reference time sources for a single clock are not possible.**

Proposal (1)

Support for two reference clocks is needed:

- **NTP -> CLOCK_REALTIME, CLOCK_MONOTONIC**
  - **Standard**
  - **Logging**
  - **Security**
- **System time (EtherCAT, TSN, ...) -> CLOCK_*PLC***
  - **Synchronous operation of robots**
  - **Cooperation between injection mold machine and robot**
  - **...**

Proposal (2)

CLOCK_*PLC* should be like CLOCK_MONOTONIC

- **clock_gettime(CLOCK_*PLC*) uses VDSO and TSC**
- **clock_nanosleep(CLOCK_*PLC*) is possible**
- **(ftrace clock)**