Getting Timestamps Right

Martin Konrad
Control System Engineer

This material is based upon work supported by the U.S. Department of Energy Office of Science under Cooperative Agreement DE-SC0000661, the State of Michigan and Michigan State University. Michigan State University designs and establishes FRIB as a DOE Office of Science National User Facility in support of the mission of the Office of Nuclear Physics.

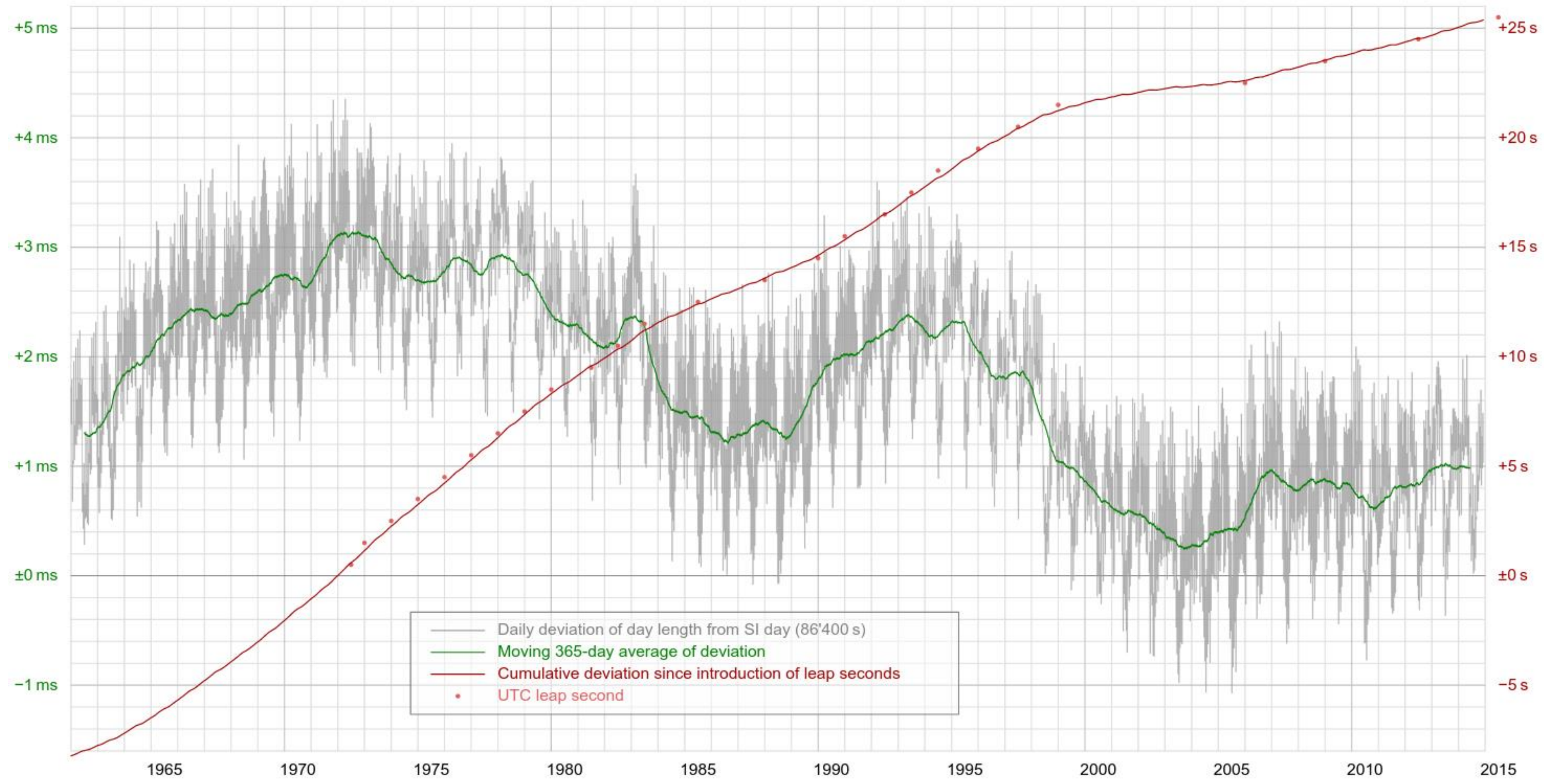
Common Causes of Bad Time Stamps

- Devices/IOCs might use different time servers which are not in sync
  - ➡ Synchronize all devices to one time source
  - ➡ Block access to other time servers
- IOCs might use different times (e. g. local time vs. UTC)
  - ➡ Use the same time on all devices
- Switching between standard time and daylight savings time causes gaps/ambiguities
  - It can be hard/impossible to establish the sequence of events with non-monotonic time
  - Archivers usually drop data that is older than the latest timestamp in the archive
    - » One hour of data will get lost!
    - » It might not be obvious to the operator which data got lost
  - ➡ Avoid daylight savings time
- Leap seconds

Leap Seconds I

- Coordinated Universal Time (UTC) is based on very stable atomic clocks
- Rotation speed of the earth varies slightly
- This causes UTC to slowly drift away from the mean solar time (UT1)
- If the deviation gets bigger than 0.9 s a leap second is inserted
- Always on June 30th or December 31th
- Next leap second is coming up on June 30th 2015 23:59:60 UTC
  - This will be the 26th leap second since 1972

Leap Seconds II



- https://commons.wikimedia.org/wiki/File:Deviation_of_day_length_from_SI_day.svg

How Leap Seconds Affect IOCs I

- UTC is subject to leap seconds
- By definition POSIX timestamps are supposed to be in UTC
- However, POSIX timestamps do not have a representation for a leap second (e. g. “2012-06-30 23:59:60” is not a valid POSIX timestamp)
- EPICS timestamps suffer from the same problem:

```
typedef struct epicsTimeStamp {
    epicsUInt32    secPastEpoch; /* seconds since 0000 Jan 1, 1990 */
    epicsUInt32    nsec;         /* nanoseconds within second */
} epicsTimeStamp;
```

How Leap Seconds Affect IOCs II

- PCs running ntpd need to “cheat”
  - ntpd is usually aware of leap seconds
  - Depending on the implementation of the kernel, systems might be able to skip the last second of the day or stretch it to 2 seconds
  - If the kernel does not support this ntpd will correct the time after it is already wrong
- Multiple machines using different ntpd implementations/operating systems might handle leap seconds in a different way
  - Timestamps might be off by up to 1 s!

But There Is More That Can Go Wrong

- If you want to calculate time differences between POSIX/EPICS timestamps accurately you need to know how many leap seconds have occurred between your two points in time
- It is impossible to write a self-contained function that calculates time differences accurately
  - Requires leap second data (e. g. from the Internet)
  - Leap second data is only known a few months in advance
    - » Needs to be updated during operation

Our Solution For FRIB

Issue	CST/CDT	CST	UTC	GPS Time	TAI
Time zone issues	Yes	Yes	No	No	No
Gaps	Yes (1h, 1s)	Yes (1s)	Yes (1s)	No	No
Duplicate timestamps	Yes (1h, 1s)	Yes (1s)	Yes (1s)	No	No
Slowly drifting away from wall clock	No	No	No	Yes	Yes

- Use International Atomic Time (TAI) to get rid of leap seconds
- We decided to live with the disadvantages
  - TAI is different from wall clock time (operators need to get used to that)
  - The offset to wall clock time changes whenever a leap second occurs

Implementation with EPICS

- Instead of UTC we will distribute International Atomic Time using NTP/PTP
- Time server does not announce leap seconds
- No further changes required (no need to touch any code) ☺