Handling time

dr. Kovácsházy Tamás

9th topic,

Handling time in operating systems and computers

Méréstechnika és Információs Rendszerek Tanszék

Budapest University of Technology and Economics
Department of Measurement and Information Systems

MŰEGYETEM 1782

Time

- One of the seven base units of the SI system
  - Very special, it grows continuously with a constant speed
  - It is strictly monotonous and continuos
- How we use it in computer systems
  - Ordering events based on time (timestamping)
  - Measuring time between events
  - Measuring other physical units based on time
    - Example: Measuring speed based on distance and time (v = s/t)
  - Etc.
- The most frequently used physical unit, though rarely think about it in computer science…
- But that is changing...

Méréstechnika és Információs Rendszerek Tanszék

Measurement of time

- Temporal measurement, or chronometry
- Two fields:
  - Calendars
    - Splitting time to special, human units or invervalls (quite different from other units), there is a lot of problem due to it
    - 1 minute is 60 seconds, 1 hour is 60 minutes, 1 day is 24 hours
    - 1 week is 7 days, 1 month is 28, 30, 31, or sometimes 29 days
    - 1 year is 365 or 366 days
    - GPS leap second, and other compensation
    - Stellar periods (human concept)
    - Calendars: Gregorian calendar, but there are other national or religious calendars
  - Clocks
    - Physical devices to measure time

Calendars

- Gregorian calendar is used now
  - It was first used on 4th of October, 1582 in some part of the world, but gained wide scale use later
    - Most of the European countries joined later
    - Russia changed to it only in 1918
  - The Julian calendar was used before it
  - All of this is due to some stellar irregularities (how Earth rotates around the Sun)
    - The Gregorian calendar will be OK for the next 3000 years
- It is very hard to determine when a past event happened (Russian 1917 October revolution happened in November)
- Lot of countries use different calendars…
- This calendar mess is a real issue from the point of view of algorithms…

Coordinated Universal Time, UTC

- French and English people cannot make an agreement on the name
- Based on the International Atomic Time (TAI)
  - 34s difference now
  - Leap Second are introduced at approximately 18 month
    - The Earth rotates slower and slower due to various energy losses (except some rare situations)
  - TAI does not take into account this while UTC does take into account the rotation of Earth
- We tend to use UTC, however, TAI seems to be better
  - All the others are based on localization…
  - Summer/winter time is not an issue (it is only localization also)
    - UTC does not depend on it, only local time

UTC details

- Monotonous
  - A minute can be 60 s, but sometimes it can be 59 or 61 seconds...
    - It was never 59s, but it was 61s 34 times since 1972
      - When these events were? It is fundamental to handle time, there is a table for it...
      - Even a major earthquake can influence UTC (Japan 2011 EQ did it)
      - The computer knows this table (it is received in patches)
    - This is a mess, some better solutions are under research...
- All global systems should use UTC, or even better, TAI
  - Travel
  - Finance
  - Internet, Network Time Protocol (NTP)
- After a certain precision there are relativistic effects (twin paradox, atomic clocks on GPS satellites)
- It nor a simple nor a transparent system (mess)...

Clocks

- Physical device to measure time
- They show time from an epoch (starting point in time)
- Components:
  - Impulse source (oscillator)
    - Provides impulses with a given frequency
  - Counter
    - Count impulses from the epoch
  - Display
    - Shows time in a predefined format based on the counter
- Clock properties:
  - Stability (how much the frequency of the oscillator changes with time)
  - Precision (how much the shown time differs from a reference clock, that is typically UTC/TAI)
  - Resolution (resolution of time shown on the display)

Why clocks are inaccurate?

- Erroneous initial setting
  - We cannot set the clock when it starts properly according to the reference clock (delays in perception and action)
  - Setting the clock against a strictly monotonous and continuous time principle!
  - It means that a clock can be set when it is not used to check time
    - otherwise all time bases processes may fail
- The frequency error of the oscillator (offset)
  - Production error (difference from the nominal value)
  - Frequency drift
    - Temperature, movement, mechanical forces influence the frequency
    - Ageing
  - The unit of frequency error is PPM (parts per million) or PPB (parts per billion)
- Frequency error accumulates in the counter
  - The clock is late or in hurry
  - The frequency must be measured and corrected
  - The strictly monotonous and continuous time principle cannot be violated!
    - If the clock is late, we run it faster to catch up with the reference time
    - If the clock is in a hurry, we run it slower to let the reference time to reach it

Clock hardware

- Oscillators
  - Real-Time Clock (RTC) based on a 32 kHz quartz
  - System tick (Timer IT) and the system clock derived from it
  - NIC clock (for all network interfaces)
    - Timestamp unit for hardware timestamping (receive/send)
  - Clock of the sound card
    - How long it plays the same MP3 file on different computers
  - External time sources: GPS receiver, DCF77, NTP or IEEE 1588 network clock, etc.
- Which one is taken into account?
  - Clock ensemble is the best, but hard to do technically
  - Synchronization of clocks…

Typical HW and SW architecture, RTC

- Real-Time Clock
  - Measures time while the computer **is switched off**
  - Low power, battery based operation
  - Properties:
    - Inaccurate, especially when the battery is low
    - Medium temperature dependence (e.g. charging the main battery of a portable computer)
    - Slow access (typically connected by a slow bus such as I2C)
    - Capable of waking up the OS on a given time (most cases, not all)
  - The counter uses very “strange” data structure
    - Binary coded decimal numbers
    - In other words, it uses a human form, not a machine form (binary)

Typical HW and SW architecture, Sysclock

- System tick and derived system time
  - Initialized at startup frotm the RTC
  - At shutdown it is written to the RTC (can be also periodically updated to the RTC)
  - The stability of the oscillator and the accuracy of the clock depends on the machine temperature
    - So it depends on the machine load
    - It may be also used to detect malfunctions of FANs in the machine
  - Construction:
    - HW counter: N*1MHz clock divided to a 10-20 ms clock tick, which requests an interrupt (binary counter)
    - SW counter for low resolution clock (binary counter)
    - Subdivision: The HW counter or some other counters (Time Stamp Counter) my be accessed for increasing the resolution (us or ns resolution is required today)
    - SW timers are derived from the clock tick also (SW timeout, time based scheduling, etc.)

Linux timer

- Jiffies (system tick): Kernel dependent (100 Hz, 1000 Hz, 250 Hz, 300 Hz)
  - Can be changed by changing one constant in a header file in the kernel source and recompiling the kernel
  - Defines the resolution of the system clock also if no subdivision is used
- High resolution timer (since kernel 2.6.21): It depends on the available HW
  - clock_getres() returns resolution (if supported)
  - Tasks waiting for timers are stored in a binary tree
- If you want to know more about timers in your Linux machine:
  - cat /proc/timer_list | less
- More than one system clocks are available in Linux:
  - Settable system clock : **CLOCK_REALTIME**
  - Monotonous, non settable: **CLOCK_MONOTONIC**
  - Process and thread clock for time domain scheduling information, etc.
- Clock synchronization
  - adjtimex – synchronize the system clock to external reference clock
  - RFC 5905 (Network Time Protocol)
  - Tunes the oscillator of the clock (virtually, not really, hardware tuning is not supported on the hardware)
  - It implements a software Phased-locked or Frequency-locked loop by changing the division ratio of the HW part of the system clock

Typical errors in common hardware

- PCs and other devices use quartz crystals (cheap ones) for oscillator frequency determination:
  - Specification: 200 ppm max. error: the clock is maximum ± 17.28 s off a day
    - It adds up to a minute in less than 4 days!
  - Average error : 70-80 ppm (NTP based measurement of thousand of computers)
  - Temperature dependence:
    - 0.5-1 ppm/°C typical
  - Better oscillators are drastically more expensive nor they solve the problem (the clocks will be off slower)
    - TCXO 1-5 ppm max. error, but costs 3-5 USD in large orders
    - OCXO 1-10 ppb max. error, but costs around 100 USD or more
- The errors are to large
  - The clocks must be synchronized to the reference time (to reference clocks)
  - Solutions:
    - Out of band : E.g. GPS, DCF77, IRIG timecode
    - In band : Network Time Protocol (NTP), IEEE 1588

Out of Band

- A dedicated communication infrastructure for clock synchronization
- Global Positioning System (GPS)
  - Localization is based on the knowledge of precise time
    - An extremely accurate estimation of UTC is available in GPS receivers
  - Interface:
    - Timecode (time in UTC), typically through an asynchronous serial port
    - Pulse Per Second signal (for clock synchronization)
      - Typically under 1 uS accuracy, but GPS modules with 100 ns accuracy are readily available!
- DCF77 (Germany), similar service exists in other countries (e.g. USA)
  - Long-wave (77.5 kHz) radio station transmits the reference time
    - Quite inaccurate due to wave propagation
    - Availability is limited in Hungary (we are too far away from the transmitter)
  - Primarily for setting clocks, watches used by people
- IRIG (Inter-range instrumentation group) timecode
  - Professional distributed measurement
  - Developed in the USA for military and aerospace use but widely used everywhere
  - Dedicated cables are used to transmit the time information

In band

- We use the regular communication channel also to transmit time
- Major problems:
  - Delay, delay asymmetry, jitter
- Typically over TCP/IP, but over Ethernet or WI-FI with dedicated technologies it is also possible
- Network Time Protocol (NTP)
  - Hierarchical clock synchronization
    - Stratum 0 (reference clocks, GPS, atomic clocks, DCF77 with limitations, etc.)
    - Stratum 1 (NTP servers connected to reference clocks)
    - Stratum N (Level N. in the clock hierarchy)
  - Redundant (multiple servers can be used to minimize errors)
  - Optimized for Internet, and precise for human use (100 ms-10 ms offset to the reference time is possible)
- IEEE 1588 Precision Time Protocol
  - Master-slave protocol for LANs
  - High precision (under 1 us is not a problem, under 100 ns is possible)
  - Hardware timestamping on the participating hosts and network instruments must be used for that precision

Summary

- Time is a strictly monotonous, continuous physical unit growing with the same pace
- Clocks: Oscillator + counter + display
- Clocks are inaccurate
  - Initial setting is erroneous, frequency offset and drift
- Setting the clock
  - The clock jumps, dangerous in applications using time
- Synchronizing the clock
  - Clock is monotonous and continuous
  - We tune the frequency of the clock
- NTP and IEEE 1588 clock synchronization protocols are available to solve the problem