Survivable Real-Time Network Services - the Nanokernel

DARPA High Confidence Networks Meeting
13-14 April 1999

David L. Mills
University of Delaware
mills@udel.edu

PostScript and PowerPoint versions of this presentation are available at
http://www.eecis.udel.edu/~mills

From *pogo*, Walt Kelly

Introduction

- NTP synchronizes time for networks of computers from PCs to Crays
- Provides submillisecond accuracy on LANs and low milliseconds on WANs in both unicast and broadcast/ multicast modes
- Provides submicrosecond accuracy with precision kernel modifications and external timing signals.
- Modern workstations with modified kernels can synchronize in principle with a resolution of one nanosecond in time and one nanosecond per second in frequency.
- Improved performance at this level required a comprehensive analysis and redesign of the NTP clock discipline algorithm and various modifications to the Unix kernel implementations

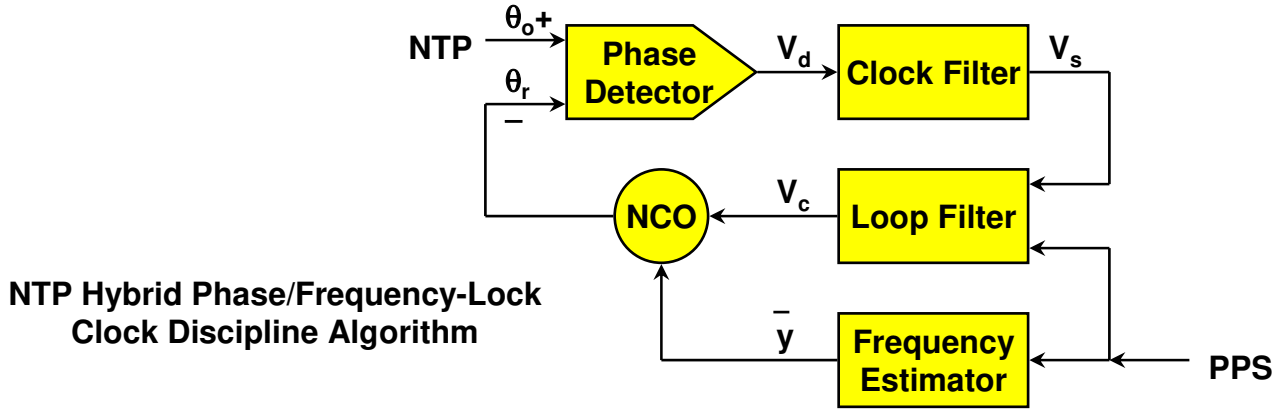
NTP capsule summary

- Network Time Protocol (NTP) synchronizes clocks of hosts and routers in the Internet
- Primary (stratum 1) servers synchronize to UTC via radio, satellite and modem; secondary (stratum 2, ...) servers and clients synchronize via hierarchical subnet
- Reliability assured by redundant servers and diverse network paths
- Engineered algorithms used to reduce jitter, mitigate multiple sources and avoid improperly operating servers
- Unix NTP daemon ported to almost every workstation and server platform available today - from PCs to Crays - Unix, Windows and VMS
- Well over 100,000 NTP peers deployed in the Internet and its tributaries all over the world

NTP enhancements for precision time

- Reduced hardware and software latencies
  - Serial driver modifications
  - Early timestamp capture in network drivers
- Precision time kernel modifications
  - Time and frequency discipline from NTP or other source
  - Pulse-per-second (PPS) signal interface and user API
- Improved local clock discipline algorithm
  - Time and frequency discipline
  - Reduced impact of jitter and glitches
- Precision time and frequency sources
  - External hardware clock
  - LORAN-C timing receiver
  - WWV/H DSP program for TI 320C25
  - Sun audio codec drivers for IRIG and CHU

Precision time performance improvements

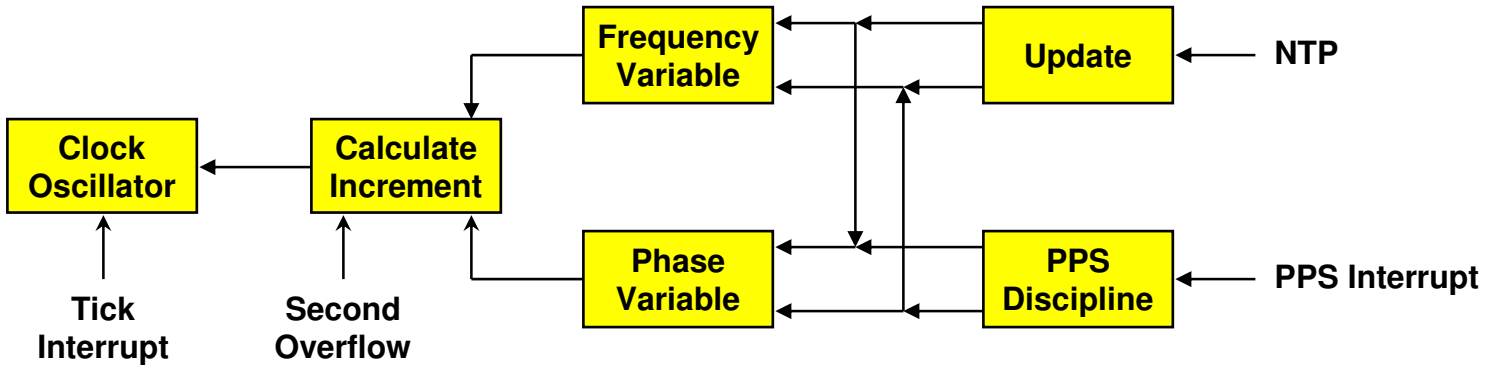


- Improved clock stability allows increased NTP update intervals without compromising accuracy
- Improved dynamic parameter estimation tolerates wider range of systematic clock errors and larger network jitter
- Optional nanokernel kernel support improves time resolution to 1 ns and frequency to 1 ns/s
- Improved time and frequency steering using PPS signal

Kernel modifications for nanosecond resolution

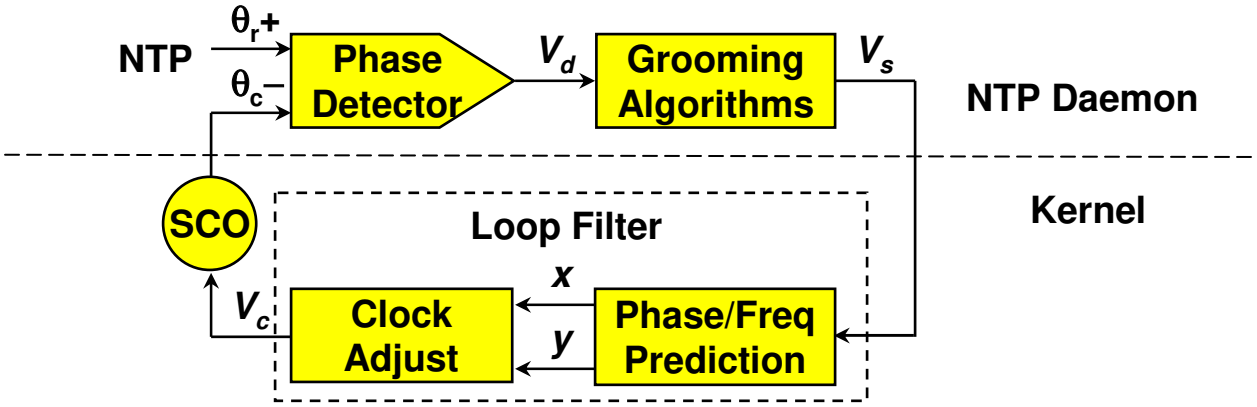
- Package of routines compiled with the operating system kernel
- Represents time in nanoseconds and fraction, frequency in nanoseconds per second and fraction
- Implements nanosecond system clock variable with either microsecond or nanosecond kernel native time variables
- Uses native 64-bit arithmetic for 64-bit architectures, double-precision 32-bit macro package for 32-bit architectures
- Includes two new system calls `ntp_gettime()` and `ntp_adjtime()`
- Includes new system clock read routine with nanosecond interpolation using process cycle counter (PCC)
- Supports run-time tick specification and mode control
- Guaranteed monotonic for single and multiple CPU systems

Nanokernel architecture



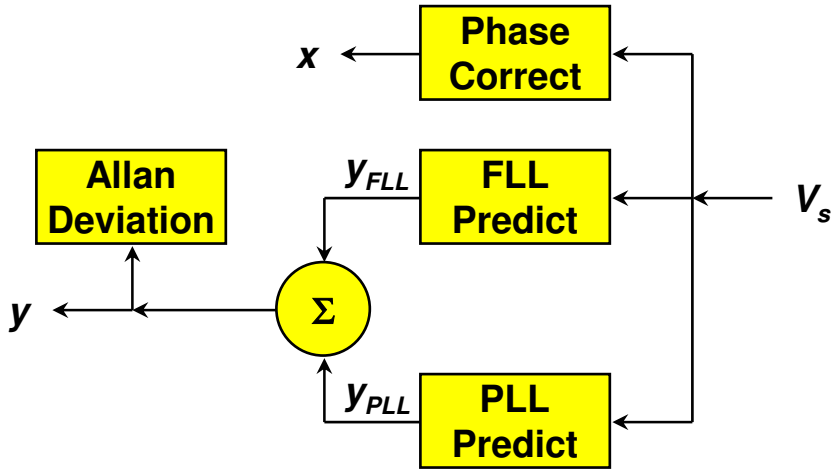
- NTP updates adjust phase and frequency according to time constant at intervals from 64 s to over one day
- On overflow of the clock second, a new increment is calculated for the tick adjustment
- Adjustment is added to system clock at every tick interrupt
- Auxiliary oscillator used to interpret microseconds or nanoseconds between tick interrupts
- PPS discipline adjusts phase at 64-s intervals, frequency at 256-s intervals

Improved NTP kernel clock discipline



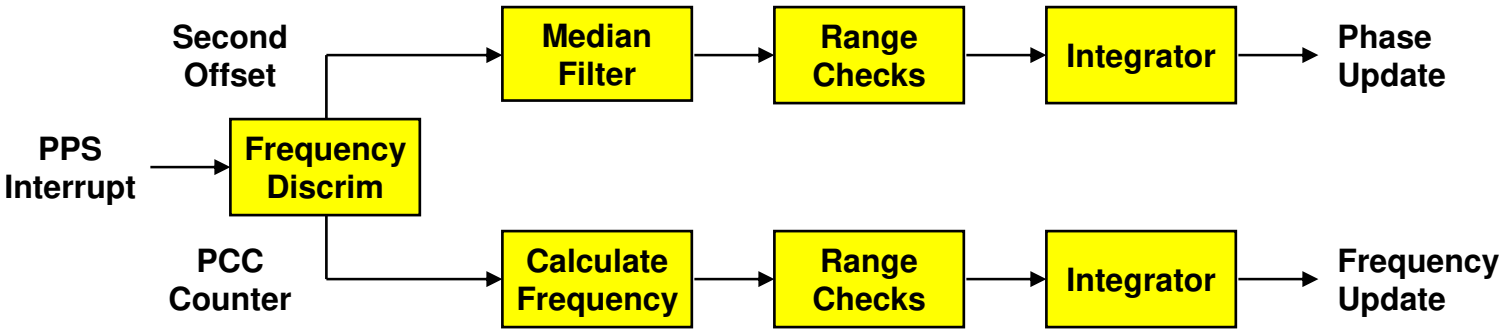
- Type II, adaptive-parameter, hybrid phase/frequency-lock loop estimates system clock oscillator (SCO) phase and frequency
- NTP daemon computes phase error $V_d = \theta_r - \theta_o$ between source and SCO, then grooms samples to produce control signal V_c
- Loop filter computes phase and frequency updates and provides tick adjustments V_c
- SCO adjusted at each hardware tick interrupt

Improved FLL/PLL prediction functions



- V_s is the phase offset produced by the data grooming algorithms
- x is the phase correction computed as a fraction of V_s
- y_{FLL} is the frequency adjustment computed as the average of past frequency offsets
- y_{PLL} is the frequency adjustment computed as the integral of past phase offsets
- y_{FLL} and y_{PLL} are combined according to weight factors computed from update interval and Allan deviation predictor

Improved PPS phase and frequency discipline

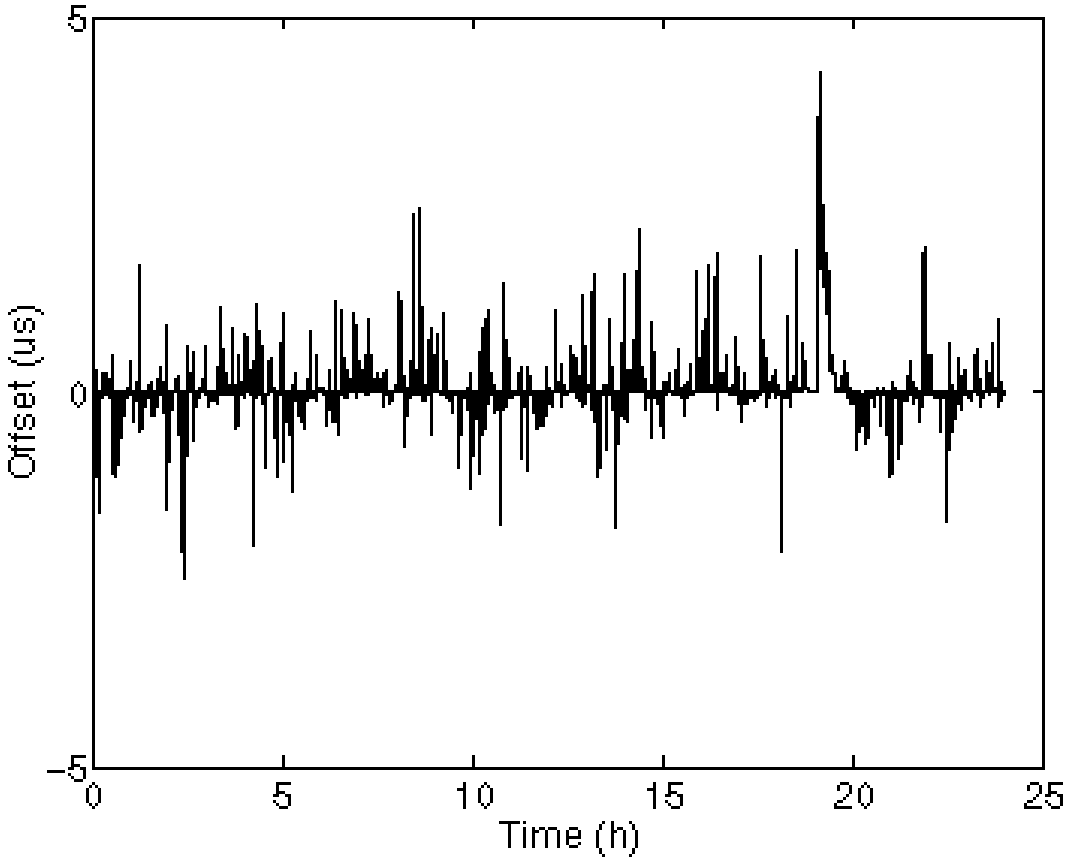


- Phase and frequency disciplined separately - phase from system clock second offset, frequency from process cycle counter (PCC)
- Frequency discriminator rejects noise and misconfigured connections
- Median filter rejects sample outlyers and provides error statistic
- Nonlinear range check filters reject burst errors in phase and frequency
- Phase offsets integrated over 64-s interval
- Frequency offsets integrated over 256-s interval

Support for multiple CPU systems

- Each processor has a read-only processor cycle counter (PCC) readable by a special instruction
- The elected master processor increments the kernel clock at each hardware timer interrupt, which occurs at intervals of 1-10 ms
- Once each second, each processor reads the master processor time and estimates the particular CPU frequency, as well as time and frequency offsets of its PCC relative to the kernel clock using an atomic interprocessor interrupt
- When the clock is read on a processor, the current time is computed from the saved master processor time plus an increment computed from the current PCC and estimated time and frequency offsets
- The kernel PLL is adjusted in time and frequency as the result of NTP updates and the PPS signal

Residual errors with Digital 433au Alpha



- Graph shows jitter with PPS signal from GPS receiver
- Principal error contribution is due to long unterminated signal cable

Current progress and plans

- Nanokernel code designed and simulated over wide range of conditions
  - Software source files in C and documentation in HTML
  - Includes simulator for testing and proof of performance in new kernel architectures
- Implemented and tested in experimental kernels for SunOS, Digital Unix and FreeBSD; Linux soon
- Future plans
  - Port to FreeBSD 3.1 for CAIRN; will be standard in FreeBSD 4.0
  - Deploy, test and evaluate using NTP Version 4 daemon in CAIRN testbed, then at friendly sites in the US, Europe and Asia
  - Port to Sun Solaris after fixing the bugs they left in the previous version

Further information

- Network Time Protocol (NTP): *www.eecis.udel.edu/~ntp*
  - Current NTP Version 3 and 4 software and documentation
  - FAQ and links to other sources and interesting places
- David L. Mills: *www.eecis.udel.edu/~mills*
  - Papers, reports and memoranda in PostScript and PDF formats
  - Briefings in HTML, PostScript, PowerPoint and PDF formats
  - Collaboration resources hardware, software and documentation
  - Songs, photo galleries and after-dinner speech scripts
- FTP server *ftp.udel.edu* (`pub/ntp` directory)
  - Current NTP Version 3 and 4 software and documentation repository
  - Collaboration resources repository
- Related project descriptions and briefings
  - See “Current Research Project Descriptions and Briefings” at *www.eecis.udel.edu/~mills*