GPS-Related Activities at the CSIRO National Measurement Laboratory, Australia

Peter Fisk[1], Tim Armstrong[2], Duncan Butler[1], Malcolm Lawn[1], and Bruce Warrington[1]

[1] *National Measurement Laboratory*
CSIRO Telecommunications and Industrial Physics
PO Box 218, Lindfield NSW 2070
Sydney, Australia
Email: peter.fisk@tip.csiro.au

[2] *Measurement Standards Laboratory, New Zealand*

Abstract

The CSIRO National Measurement Laboratory (NML) has been actively pursuing the development of useful, reliable and low-cost GPS based systems for precise time and frequency transfer with the integrity and reliability required of a national timing laboratory. This paper outlines the systems, and some applications.

1. Introduction

There is a need within Australia for systems capable of high-integrity time and frequency transfer and remote operation. The initial motivation for developing these systems at NML was to provide a "turnkey" solution for customers who wished to maintain high-accuracy, high-integrity frequency and time traceability from a Cs or Rb frequency standard to NML without the inconvenience of shipping the standard to NML for calibration.

The NML Timing Systems in their most basic form consist of an Intel-based PC running the Linux operating system, a commercially available GPS receiver board and antenna, and a commercially available counter-timer. NML's philosophy was to use relatively generic hardware, and to make the software as independent as possible of the hardware. This allows the system to be flexible, and capable of extensive future development as customer demand changes and hardware technology and availability evolves. Because the system is controlled by a Linux-based computer, it can perform many different functions in addition to GPS Common-View time transfer, and can be controlled and monitored remotely via the Internet or a telephone line.

There are presently ten NML Timing Systems operating outside NML's Sydney premises, including eight outside Australia. Extended versions of these systems have also proved to be useful and reliable as remotely-operable Network Time Protocol servers, TV-Sync monitoring stations, and GPS integrity monitoring stations.

2. System Architecture

The GPS Common-View technique as specified by BIPM's Consultative Committee on Time and Frequency (CCTF) was selected as the most suitable time transfer protocol for legal traceability to NML, due to its high immunity to undetected errors arising from poor reception of GPS signals and failures in the GPS constellation. Because this technique involves the comparison of electrical pseudoranges with geometric ranges to individual satellites, a receiver which is capable of outputting extensive raw GPS data is required. To our knowledge, the only low-cost receiver

satisfying this requirement is the Motorola VP Oncore, and this receiver is used in all of the currently operating NML Timing Systems. The VP Oncore is a single-frequency L1 receiver. The hardware of the Motorola-based version of the NML Timing System is shown schematically in Figure 1.

The Motorola company unfortunately ceased production of the VP Oncore in early 2000. The only source of presently available replacement receivers appears to be the Javad company (now Topcon Positioning Systems), which is advertising both single- and dual-frequency receivers; these will enable the NML Timing System to be extended to dual-frequency operation. However, at the time of writing, NML has not tested these receivers.

The software architecture is described in Appendix A. A modular approach was used to simplify the software development and debugging process, and to permit extension of the software to new GPS Common-View data processing protocols and the accommodation of new hardware such as dual-frequency receivers. The software is written in the C and Perl programming languages using the public domain compilers, interpreters and libraries typically distributed with the Linux operating system.

3. Performance

Figure 2 shows a zero-baseline comparison between the CCTF-format files generated by the Motorola-based NML Timing System and an Allen Osborne Associates TTR6 receiver sharing a common timing reference. Similar comparisons using a third GPS Common-View receiver (3S Navigation R-100T) indicate that the apparent drift (17.3 ps/day) is most likely not due to the Motorola-based system. The reason for this drift is not yet understood.

4. Application as an NTP Server

Since early 2000, NML has used these systems very successfully as remote Network Time Protocol (NTP) servers. These NTP servers consist of the basic NML Timing System as shown in Figure 1, together with a Rb or Cs clock which can be controlled via a serial connection to the Linux PC. The epoch of the 1 pps output of the clock is maintained with respect to UTC(AUS) using the standard GPS Common-View time transfer technique. The 1 pps signal is monitored from the Linux PC via a simple circuit which uses the pulse to trigger sending an ASCII carriage return character over a serial link. The arrival of this character at the PC serial port is used as a precise timing reference by an NML-developed driver installed in the standard NTP software suite on the Linux PC. A second GPS receiver is used to generate a timestamp for the 1 pps pulses, which is received by a second custom driver. NML operates these systems in Sydney, Melbourne and Perth, and the installation of further systems in other cities is planned.

5. Conclusion

Because the NML Timing System is based on a standard yet powerful multi-tasking UNIX-like networked operating system, and uses reasonably generic hardware, it will be maintainable and extendable into the foreseeable future. NML has installed remote systems in a number of locations, where they provide (in various combinations) precise frequency references, precise time references and NTP services. The systems have proved reliable, and the ability to log in remotely with complete access to all software and operating system functions has made remote upgrades, maintenance, user support and troubleshooting a practical proposition.

Appendix A: Brief descriptions of software comprising the NML CCTF Common-View time-transfer system

This appendix briefly describes the software components of the NML GPS common-view time transfer system. This system downloads pseudorange, ephemeris, almanac and other data on a continuous basis from a Motorola VP Oncore GPS engine, and processes it into the format specified by the BIPM CCTF working group on time transfer standards.

A.1 Hardware

The software in its current form (June 2000) requires the following hardware:

- PC compatible computer, Pentium 2, 233 MHz or better, with at least 16 MB of RAM
- CDROM drive and about 3 GB of hard disk space.
- Network card or modem
- Agilent HP53131 or HP53132 counter/timer
- IOTech Micro-488 serial/GPIB converter
- Motorola VP Oncore GPS engine, with 1 pps timing output option, and antenna

Minor modifications to the software would be required to use a different counter/timer. A version of the software for a Javad L1 and L1/L2 GPS engine (which has a suitable instruction set) is under development. Motorola's web page states that the VP Oncore receiver has been discontinued since 31 Dec 1999.

A.2 Main system

These are the programs which run either continuously or daily as part of the process of generating daily CCTF-format time transfer data. They run under the Linux operating system. It may be helpful to refer to the system block diagram included at the end.

Program: `process`
Author: Peter Fisk, Bruce Warrington, Duncan Butler
Description: A Perl script which orchestrates the processing of the raw data from the Motorola VP Oncore receiver and the Agilent counter. It executes a sequence of subsidiary processing programs, and refers to intermediate temporary data files placed in the `/tmp` directory which are deleted at the completion of processing.
Usage: `process [MJD]`
The MJD is optional. If it is omitted, the data for the previous MJD is processed.
Execution: Under normal circumstances, this script is executed shortly after 00 UTC by a `cron` job.

Program: `proc2`
Author: Peter Fisk, Bruce Warrington, Duncan Butler
Description: Exactly the same as `process`, except that it does not delete the temporary files. It is intended for debugging purposes.
Usage: `proc2 [MJD]`
The MJD is optional. If it is omitted, the data for the previous MJD is processed.
Execution: Normally executed manually.

Program: `onclog`
Author: Tim Armstrong, Bruce Warrington

Description: A C++ program which sets up the Oncore GPS receiver with parameters from the `cctf.setup` file, instructs the GPS receiver to output the relevant data and logs this data into the file `MJD.rxrawdata`. It is usually configured to maintain the computer system clock within about 0.2 seconds of UTC. This feature can be disabled where it is desired that the system clock be kept on time by another method (such as NTP).
Usage: `onclog <setup_file>`
The setup file (usually `cctf.setup`) contains the parameters for the receiver. Only the antenna coordinates should need to change.
Execution: Normally started by the Perl script `check_rx`, which in turn is normally executed as a regular `cron` job.

Program: `get_counter_data`
Author: Peter Fisk, Steve Quigg
Description: A Perl script which logs the data from the HP53131 counter into a file `MJD.cvtime`. Counter configuration parameters such trigger levels are in the file `counter.setup`, and may be edited by the user. Note that this file is read only when `get_counter_data` starts.
Usage: `get_counter_data`
Execution: Normally started by the Perl script `check_cntr`, which in turn is normally executed as a regular `cron` job.

Program: `rinexlog`
Author: Bruce Warrington
Description: A Perl script which extracts the relevant data from the file `MJD.rxrawdata` and formats it into a RINEX file. The output is piped by the `process` or `proc2` scripts to the file `MJD.rinex`.
Usage: `rinexlog <rawdatafile>`
Execution: Normally executed by the `process` or `proc2` scripts.

Program: `readlog`
Author: Peter Fisk
Description: A C program which extracts the pseudorange, receiver clock correction, receiver status, satellite health and other data from the `MJD.rxrawdata` file. The current number of leap seconds is read from the RINEX file, to convert from GPS time used by the receiver to UTC. The output is piped by the `process` or `proc2` scripts to the file `MJD.notime`.
Usage: `readlog <rawdatafile> <rinexfile>`
Execution: Normally executed by the `process` or `proc2` scripts.

Program: `combine`
Author: Peter Fisk
Description: A Perl script which merges the time-stamped counter reading extracted from the `MJD.cvtime` file with the corresponding pseudorange readings for the eight channels of the GPS receiver in the `MJD.notime` file. The output is piped by the `process` or `proc2` scripts to the file `MJD.cvdata`.
Usage: `combine <notimefile> <timefile>`
Execution: Normally executed by the `process` or `proc2` scripts.

Program: `schedule_extract_all`
Author: Peter Fisk, Duncan Butler
Description: A C program which extracts only the data relevant to the current BIPM tracking schedule for the present day. The BIPM tracking schedule is held in a file called `sched`. The output is piped by the `process` or `proc2` scripts to the file

MJD.cvdata.cctf.all.

Usage: schedule_extract <MJD.cvdata> <MJD>

Execution: Normally executed by the process or proc2 scripts.

Program: track_sort

Author: Duncan Butler

Description: A C program which sorts the multi-channel output from schedule_extract_all into a form which the remainder of the processing software will recognise. The input file contains data for every visible satellite during a scheduled track; the output is sorted into individual satellite tracks, and is piped by the process or proc2 scripts to the file MJD.cvdata.cctf.

Usage: schedule_extract <MJD.cvdata.all> <MJD>

Execution: Normally executed by the process or proc2 scripts.

Program: quadfits

Author: Peter Fisk

Description: A C program which performs the quadratic fitting specified in section (ii) of Annex II of the "Technical Directives for the Standardisation of GPS Time Receiver Software" document. The output is piped by the process or proc2 scripts to the file MJD.15_sec_fits_cctf.

Usage: quadfits <MJD.cvdata.cctf> <MJD.rinex>

Execution: Normally executed by the process or proc2 scripts.

Program: corrections2

Author: Peter Fisk, Bruce Warrington, Malcolm Lawn, Duncan Butler

Description: A C program which performs the corrections specified in sections (iii), (iv) and (v) of Annex II of the "Technical Directives for the Standardisation of GPS Time Receiver Software" document. It reads the antenna coordinates from the file specified in the command line. The output is piped by the process or proc2 scripts to the file MJD.15_sec_fits_corr.

Usage: corrections1 <MJD.15_sec_fits_cctf> <MJD.rinex> <MJD> <coordinate_file>

Execution: Normally executed by the process or proc2 scripts.

Program: linearfits

Author: Peter Fisk

Description: A C program which performs the linear fitting specified in section (iii) of Annex II of the "Technical Directives for the Standardisation of GPS Time Receiver Software" document. The output is piped by the process or proc2 scripts to the file MJD.cctf.final.

Usage: linearfits <MJD.15_sec_fits_corr>

Execution: Normally executed by the process or proc2 scripts.

Program: checksums

Author: Peter Fisk

Description: A Perl script which adds the checksums specified in Annex III of the "Technical Directives for the Standardisation of GPS Time Receiver Software" document. The output is piped by the process or proc2 scripts to the file MJD.cctf.final, which is the complete CCTF output file.

Usage: checksums <MJD.cctf.nocksum>
where MJD.cctf.nocksum is created by prepending cctf_header (containing the header information) to MJD.cctf.final.

Execution: Normally executed by the process or proc2 scripts.

A.3 Utility programs

These programs are run when configuring or troubleshooting the system.

Program: `oncstat_CV`
Author: Peter Fisk
Description: A Perl script which returns status information on the Oncore GPS receiver. Note that when the `onclog` process is running, `oncstat_CV`, or any other program which accesses the GPS receiver, should NOT be run. Remove the `crontab` (which would otherwise restart `onclog`), kill the `onclog` process, and execute `turnoff.data` before running this command.
Usage: `oncstat_CV`
Execution: The user.

Program: `turnoff.data`
Author: Peter Fisk
Description: A Perl script which tells the GPS receiver to stop sending data continuously. This program must be run before `oncstat_CV` will work, if the GPS receiver was set up by onclog. Note that when the `onclog` process is running, `turnoff.data`, or any other program which accesses the GPS receiver, should NOT be run. Remove the `crontab` (which would otherwise restart `onclog`) and kill the `onclog` process before running this command.
Usage: `turnoff.data`
Execution: The user.

Program: `coldstart_CV`
Author: Peter Fisk
Description: A Perl script which tells the Oncore GPS receiver to discard all ephemeris, almanac, setup and position data, and to start from scratch with default settings. Note that when the `onclog` process is running, `coldstart_CV`, or any other program which accesses the GPS receiver, should NOT be run. Remove the `crontab` (which would otherwise restart `onclog`) and kill the `onclog` process before running this command. The progress of the receiver restart can be monitored using `oncstat_CV`.
Usage: `coldstart_CV`
Execution: The user.

A.4 The `crontab` file

This file is used to schedule automatic execution of software, in particular the daily processing of data and the automatic check (usually every 5 minutes) that the receiver and counter logging processes are both running. This allows the system to start logging data automatically on power up once the `crontab` file has been installed, typically by executing `crontab crontab`. Once installed, it does not need to be re-installed if the machine is rebooted. It can be removed using `crontab -r` to prevent the logging processes from starting automatically.

Acknowledgments

The authors would like to thank Colin Coles and Steve Quigg for their extensive assistance with hardware development.

References

Allan, D. W. and Thomas, C., 1994, "Technical Directives for Standardisation of GPS Time Receiver Software", *Metrologia* **31**, 69-79.

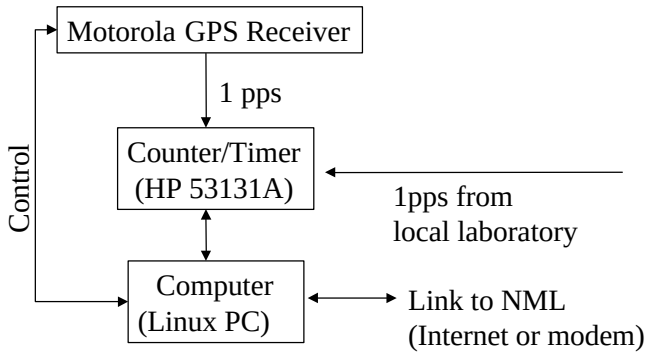

Figure 1: Schematic diagram of the NML GPSCV time transfer system based on the Motorola VP Oncore GPS engine.

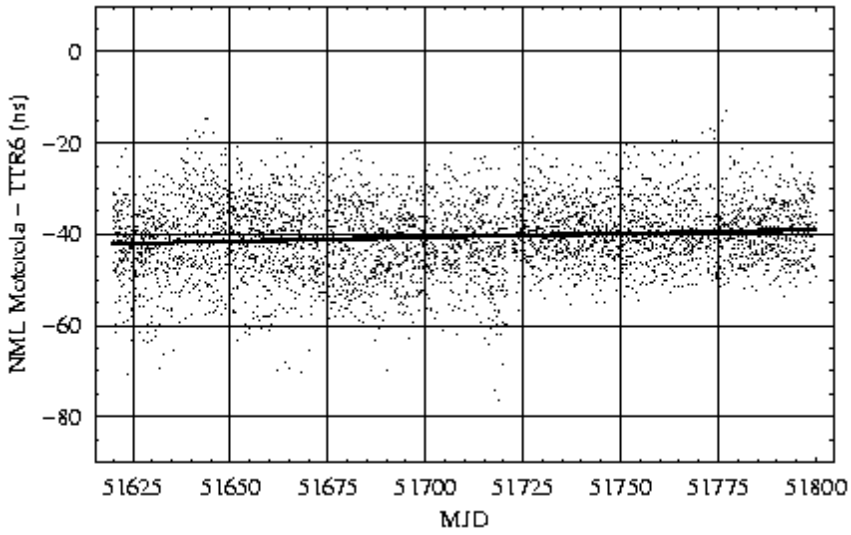

Figure 2: Zero-baseline comparison between the NML Motorola-based system and an AOA TTR6 system sharing a common timing reference. The solid line is a least-squares fit to the data, and has a slope of 17.3 ps/day and an RMS scatter of 7.8 ns.

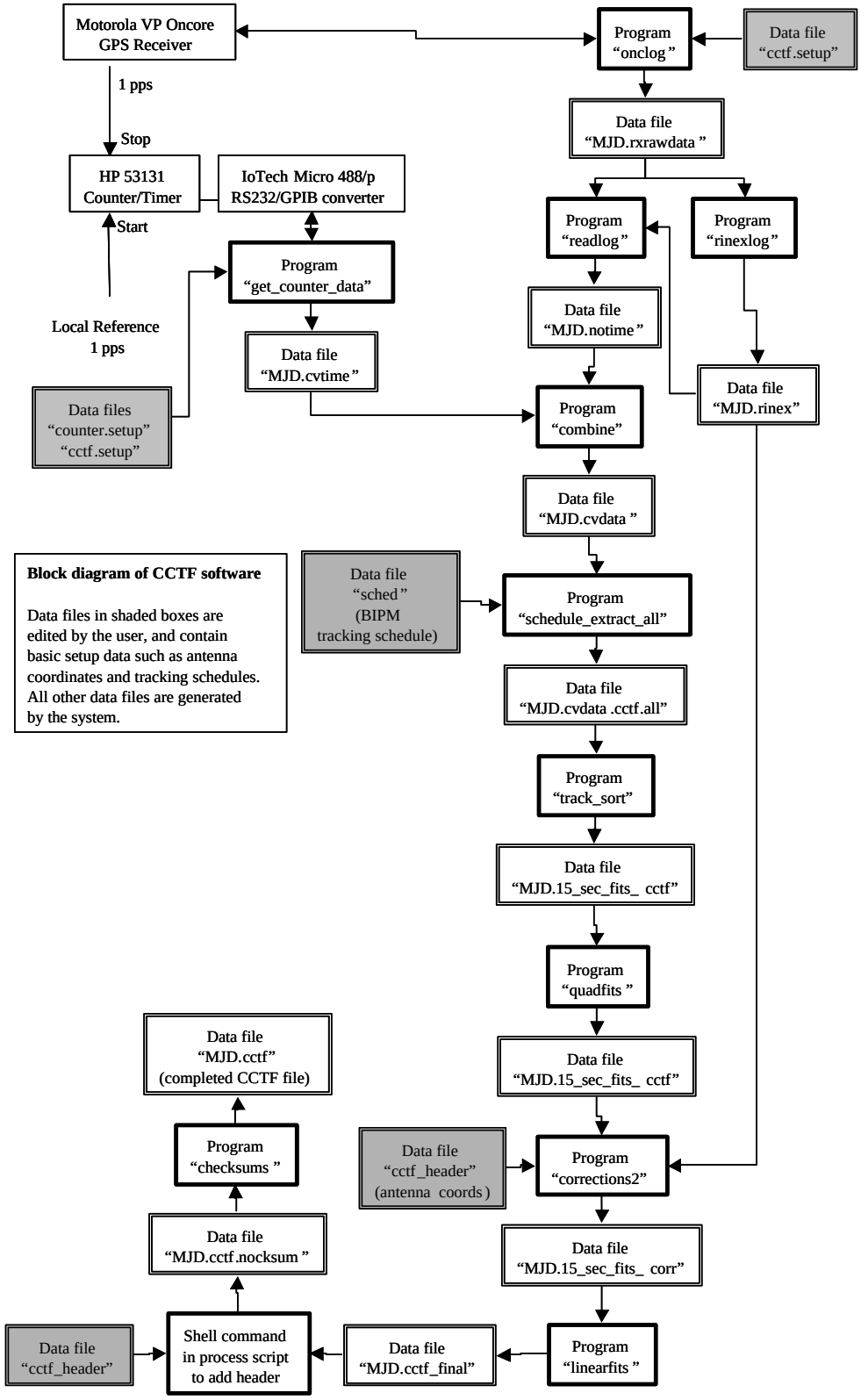

Figure 3. Block diagram of data processing software for CCTF GPS Common-View time transfer system.