Development and Field Testing of a DSP-Based Dual-Frequency Software GPS Receiver

Brady W. O'Hanlon, Mark L. Psiaki, Paul M. Kintner, Jr., *Cornell University, Ithaca, NY*
Todd E. Humphreys, *The University of Texas at Austin, Austin, TX*

BIOGRAPHY

Brady W. O'Hanlon is a graduate student in the School of Electrical and Computer Engineering at Cornell University. He received a B.S. in Electrical and Computer Engineering from Cornell University. His interests are in the areas of ionospheric physics, space weather, and GNSS technology and applications.

Mark L. Psiaki is a Professor in the Sibley School of Mechanical and Aerospace Engineering. He received a B.A. in Physics and M.A. and Ph.D. degrees in Mechanical and Aerospace Engineering from Princeton University. His research interests are in the areas of estimation and filtering, spacecraft attitude and orbit determination, and GNSS technology and applications.

Paul M. Kintner, Jr. is a Professor of Electrical and Computer Engineering. He received a B.S in Physics from the University of Rochester and a Ph.D. in Physics from the University of Minnesota. His research interests include the electrical properties of upper atmospheres, space weather, and developing GNSS instruments for space science. He is a Fellow of the APS and a Jefferson Science Fellow at the Department of State.

Todd E. Humphreys is an assistant professor in the department of Aerospace Engineering and Engineering Mechanics at the University of Texas at Austin. He received a B.S. and M.S. in Electrical and Computer Engineering from Utah State University and a Ph.D. in Aerospace Engineering from Cornell University. His research interests are in estimation and filtering, GNSS technology, GNSS-based study of the ionosphere and neutral atmosphere, and GNSS security and integrity.

ABSTRACT

A real-time software GPS receiver for the L1 C/A and L2 C codes has been implemented on a Digital Signal Processor (DSP) and tested in both scintillating and non-scintillating environments. This receiver is being developed as a low-cost space weather instrument with improved tracking robustness in comparison to a traditional semi-codeless dual-frequency receiver and with flexibility in its choices of signal tracking algorithms and data outputs.

The receiver is capable of continuous background signal acquisition and utilizes the L1 C/A code to assist in acquisition of the L2 C signal. Efficient on-the-fly generation of oversampled PRN code replicas for the L2 CM and CL codes, which are required for real-time software radio signal processing, has been implemented to ensure a manageable requirement for memory. Bit-wise parallel correlation techniques have been implemented to reduce the number of operations needed for correlation. The receiver currently tracks both the L2 CL and CM codes for the purpose of calculating TEC.

Results are presented based on data generated by a signal simulator, on real data taken in Ithaca, NY (42.44 N, 76.48W), and on real data taken during ionospheric scintillation in Natal, Brazil (5.8S, 35.2W) in January 2009. Position and velocity solution accuracy is evaluated using both real and simulated data.

I. INTRODUCTION

As the computational brawn of multi-purpose processors has grown, processing GNSS signals using software has become ever more manageable. The use of programmable processors for GNSS receivers has been well explored in recent history[1-4] but as processors have become more powerful, GNSS systems have continued to add additional signals and codes. In this paper, we explore the use of a small, low-power digital signal processor as a dual-frequency GPS receiver utilizing the L1 C/A and L2 Civilian codes of the Global Positioning System. The current work can be viewed as a direct extension of the work presented in Ref. 1. The receiver in Ref. 1 was capable of parallel processing of 43 L1 C/A channels and continuous background signal acquisition; the present receiver is capable of 30 L1 C/A channels, 30 L2C channels, continuous background acquisition, and on-board position, velocity, and time calculations. With the planned launch of 12 Block IIF GPS satellites (that

provide the L2C signal) in the next three years[5] in addition to the eight Block IIRM satellites already in orbit and providing the L2C signal, this platform is well suited to become an inexpensive and science-worthy GNSS receiver that utilizes these signals.

The remainder of this paper is organized as follows:

This work will discuss the hardware components of the receiver, the work done in implementing acquisition and tracking of the L2C signal and the navigation solution, and benchmark receiver performance in terms of power usage and navigation accuracy. Additionally, preliminary results from testing the receiver in a scintillating environment during a week-long field campaign in Natal, Brazil are presented.

II. HARDWARE OVERVIEW

This dual-frequency DSP-based receiver is built from a combination of off-the-shelf parts and custom-fabricated circuit boards. In order to minimize additional development and to keep costs down, a frequency plan was devised that would allow reuse of existing radio-frequency hardware. An L1-only system based on the Plessey GP2015 has previously been developed and extensively used by Cornell University researchers for both embedded[1] and non-embedded[2] software receivers. As this design has an input bandwidth of approximately 2MHz, it is an acceptable solution for a receiver utilizing the L2C codes. To enable reuse of this existing hardware the input signal is split, with one path going to one GP2015-based front end, and the other path getting mixed with a 347.826 MHz sine wave generated by an Analog Devices voltage controlled oscillator. This up-converted signal is then sent to a separate and identical GP2015-based front end. As a result of the mixing, the L2 signal translated to a center frequency approximately 6 KHz below the L1 frequency, and no additional modifications to the RF hardware are required (see Fig. 1).

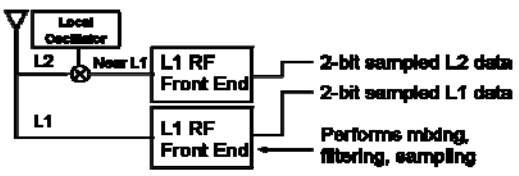

Figure 1. RF portion of dual-frequency GPS receiver.

After being mixed to near baseband and sampled at roughly 5.714 MHz by the front-end units, the resultant samples with a final intermediate frequency of about 1.405 MHz are sent to a Xilinx CPLD. The CPLD then packs these samples into 32-bit words consisting of 8 2-bit samples of L1 and L2 data, in the format

[L2 Mag | L1 Mag | L2 Sign | L1 Sign]

where Mag indicates a magnitude bit, and Sign indicates a sign bit. The mapping of 2-bit data to integer values is as given in Table 1.

Table 1: Sign/Magnitude bit mapping

	Magnitude = 0	Magnitude = 1
Sign = 0	-1	-3
Sign = 1	1	3

The CPLD also generates a frame synchronization pulse to indicate the start of a 32-bit word. The CPLD, voltage controlled oscillator, and both RF front ends all use the same clock source. A schematic representation of the receiver is shown in Fig. 2, and a picture of the receiver is shown in Fig. 3. Note that the dual-frequency front end is essentially the same at described in Ref. 6.

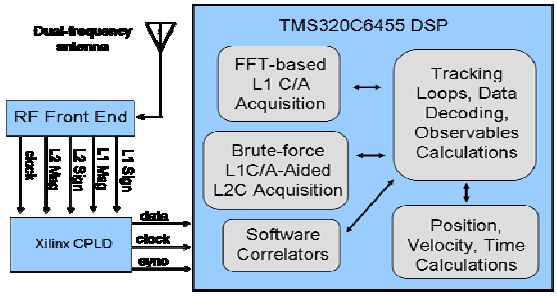

Figure 2. Schematic representation of DSP-based dual-frequency GPS receiver.

The Digital Signal Processor used here is a 1.2 GHz Texas Instruments TMS320C6455 DSP. This is a fixed-point DSP, meaning that it has no hardware floating-point unit but can do floating-point math through emulation. The sampled data is read into the DSP via one of its Multi-Channel Buffered Serial (MCBSP) Ports. The DSP has two such channels, each with a maximum speed of 100 MHz. As the data is sampled at 40/7 MHz, and each sample time produces 4 bits of data (L1 sign and magnitude, L2 sign and magnitude), 40/7*4/100 = 23% of the capacity of one of the ports is currently being used. The additional unused capacity could be used in the future for additional GNSS signals (e.g., Galileo). After the data is read via the DSP serial port, it is stored in a set of circular buffers for signal acquisition and tracking.

Figure 3. A photograph of the dual-frequency DSP-based GPS receiver prototype.

The DSP does all correlations, tracking loop control, and navigation solution calculations. The final piece of the hardware puzzle is a personal computer, used only for display of status indicators and data logging. Communication between the personal computer and DSP is currently being done via the Texas Instruments "Real Time Data Exchange" link.

Power consumption was measured with the DSP operating at the full 1.2 GHZ clock rate, though CPU utilization is only 55%. The DSP development kit consumed roughly 6 Watts of power. Presumably, some of this is being used by parts of the development kit that will no longer be present once the system moves to a custom-fabricated DSP circuit board. Power consumption could also be reduced by scaling back CPU frequency and reducing unutilized CPU cycles. The RF portion of the receiver consumed roughly 5 Watts. As a side note, the entire RF portion has recently been replaced by a custom-built dual-frequency solution that consumes only 1.75 Watts, but this will not be discussed further in this paper. Any power consumed by the antenna preamplifier or personal computer is neglected here, and that consumed by the CPLD is negligible.

III. SOFTWARE OVERVIEW

The software presented here takes advantage of several innovative processing techniques specific to GNSS receivers in an attempt to most efficiently process the incoming signals. For efficient code and carrier mixing, a bit-wise parallel technique has been implemented. The data is 2-bit quantized into a sign bit and a magnitude bit, and then 32 samples of sign data are packed into a single integer, while the corresponding 32 samples of magnitude data are packed into another 32-bit integer. These 32 samples are then processed in a parallel fashion. A full discussion of this technique can be found in Ref. 7.

C/A code replicas for all PRNs, each with a predetermined number of code phases, are pre-computed and stored in memory, as are local carrier replicas spanning a predetermined frequency range. Pre-generation of the L2C codes is not feasible, as will be discussed in Section V, so a method is implemented to efficiently create these up-sampled codes in real-time.

The default PLL discriminator used for tracking the L1 C/A signal is a decision-directed two-quadrant arctangent. Note, however, that two-quadrant arctangent, 4-quadrant arctangent, and other discriminators are also available. The default PLL bandwidth is 7.5 Hz, and the default accumulation interval (pre-detection time) is 10 ms. The only difference between tracking of L1 C/A signals and L2 C signals is the selection of tracking loop; for L2C the receiver leverages the lack of data bit modulation by implementing a 4-quadrant arctangent discriminator.

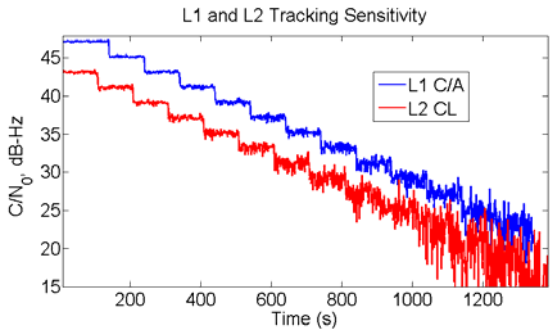

Figure 4. Tracking sensitivity for L1 C/A and L2 CL.

Tracking sensitivity for both L1 C/A and L2 CL was tested using data generated by a Spirent Simulator that included graded reductions in signal power. These results are shown in Fig. 4. Note that the transmitted power on L2C was less than on L1 C/A so that the plots would be offset. The L1 C/A and L2 CL signals were both successfully tracked down to a carrier-to-noise ratio of about 25 dB-Hz without any cycle slips.

The majority of the code is written in object-oriented C/C++. This coding paradigm was found to be the most conducive for code reuse and easy addition of new GNSS signals to the receiver.

IV. NAVIGATION SOLUTION

In the previous version of this receiver, the navigation solution was computed in post-processing using the pseudoranges measured by the receiver. Calculating the navigation solution on-board the DSP was the preferred solution as this would move all computation onto the embedded processor, thereby obviating the need for an external computer for anything other than displaying the data. This was the last thing required to make the DSP-based receiver a truly stand-alone solution. However, as has been noted, the CPU utilized for this project was a fixed point processor, with no hardware unit for doing floating point math. It was unknown whether or not implementing the navigation solution entirely in fixed

point would be possible given the required numerical precision. For example, to get accuracy on the order of one meter in resolving the satellite locations, one needs an angular resolution of approximately

$$\Delta\theta \approx 180/(\pi * 26{,}600km) = 2 * 10^{-6} \text{ deg}$$

where the satellite orbital radius has been taken to be 26,600 km. Implementing in fixed point the various trigonometric functions required with this degree of accuracy would either necessitate staggeringly large lookup tables or functions so complicated they would likely be slower than the emulated floating-point version. The alternative to this is using floating-point math, which the CPU can do via emulation. This was the first approach attempted in the interest of minimizing development time.

The navigation solution was written in accordance with the object-oriented paradigm utilized elsewhere; each satellite was considered its own object with associated state variables (e.g., position, velocity) and data (e.g., ephemeris data). Similarly, the navigation solution is considered an object with its state variables (such as position, velocity, and time), and data (pointers to the satellite objects used in the solution, the observables). This approach allows easy inclusion of additional GNSS signals in the navigation solution calculations.

Receiver position and velocity are calculated using only the L1 C/A code range. Although the receiver tracks the L2 CM and CL codes, they are currently used only for estimating TEC and observing effects due to signal propagation. Corrections to the pseudoranges due to ionospheric delay are calculated using the Klobuchar model[8] parameters transmitted in the navigation message. In the future, the ionospheric delay will be measured in real-time and corrections based on this applied to the pseudoranges, but due to the current incomplete population of the GPS constellation with satellites that transmit the L2C codes, it was decided that rather than apply measured corrections to some signals and modeled corrections to others, a single approach would be taken for all satellites. Additionally, corrections to the pseudoranges due to tropospheric delay are calculated using a combination of the Saastamoinen model[9] and the Neill mapping function[10].

The precision of the navigation solution has been evaluated in several different circumstances. First, testing was done using a Spirent signal simulator. This solution includes no multipath, ionospheric, or tropospheric effects; horizontal solution precision is shown in Fig. 5. The larger variance in the East direction is most likely due to satellite geometry, and there was a roughly 1 meter bias, the source of which is not fully understood, that was removed. This solution was calculated using 10 satellites.

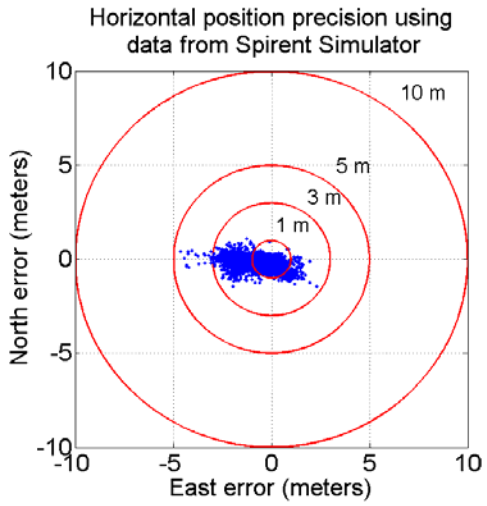

Figure 5. Navigation solution precision using simulated data from 10 satellites over 1 hour, 1 Hz solutions.

Navigation solution precision was also calculated using live data from a rooftop antenna located in Ithaca, New York (42.44 E, 76.48 W). There was a roughly 3 meter bias in this solution (as compared to a 3-day averaged position from a Cornell SCINTMON GPS receiver[11]) that was removed, possibly due to differences in ionospheric or tropospheric correction implementations between the two receivers. This result is shown in Fig. 6. The standard deviation of the East and North errors in the navigation solution are each on the order of a meter, and the vertical error standard deviation is roughly 2 meters. No averaging or smoothing was done in the calculation of these solutions, though the software has the capability to do so.

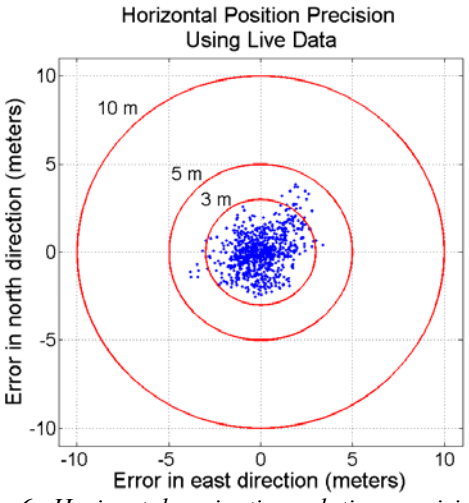

Figure 6. Horizontal navigation solution precision using data from a rooftop antenna in Ithaca, NY over 15 minutes, 1 Hz solutions.

The receiver velocity is also calculated as a part of the navigation solution. Velocity is calculated using the Doppler shift of the L1 C/A signal. To verify the velocity

calculation, a simulated data set was created using a Spirent signal simulator wherein the receiver was moving in a circle of radius 6 km with a speed of 500 m/s. To track this signal, the phase-locked loop bandwidth was increased to 10 Hz and the integration time (pre-detection interval) was decreased to 2 ms. A plot of error in horizontal speed is shown in Fig. 7.

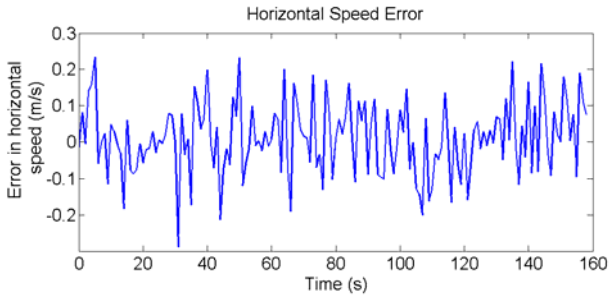

Figure 7. Error in horizontal speed when traveling at 500 m/s in a circle with a radius of 6 km (in the local horizontal plane).

V. L2 CIVILIAN CODES

With the ongoing modernization of the GPS constellation and expansion of other global navigation satellite systems, the number of signals available to the civilian user is rapidly expanding. Multiple-frequency measurements are of paramount importance for resolving ionospheric delays and producing more reliable estimates of user position, velocity, and time. The new GPS L2 civilian codes (CM, the medium length code, and CL, the long code) are particularly well suited for use in a software receiver due to their relatively low combined chipping rate of 1.023 MHz. The low chipping rate of the codes means the signals have a corresponding low bandwidth and can thus be processed by a receiver that samples at a lower rate. Processing requirements are roughly proportional to sampling rate, so a lower sampling rate eases the computational burden on the CPU.

Pre-generation and storage of the L2C PRN codes at the front-end sampling frequency is not practical due to the large amount of space required (approximately 2 MB per PRN per code phase offset). Similarly, brute-force generation of the codes in real time and upsampling to the RF front end sampling frequency is not practical because of the large computational cost, partly due to using floating point operations to achieve the necessary code timing precision and partly for sample-by-sample code generation and repackaging into 32-sample integer words. To allow the use of L2C, the technique presented in Ref. 12 has been implemented in a slightly modified form. This algorithm has been modified to ignore the effect of the Doppler shift on the code chipping rate over a single millisecond of code. Estimates of code phase are done each millisecond taking into account the effect of Doppler shift on chipping rate, but each 1-millisecond portion of the PRN code replica is created assuming chipping at the nominal rate. The net effect is a negligible loss in power (assuming Doppler shift magnitude less than about 5KHz). This technique has been previously used in a software receiver[2], but not in an embedded processor as in the current work.

The receiver currently acquires the L2 signals using a scheme whereby the acquisition is aided by the L1 C/A signal from the same satellite. Given that the Doppler shift depends mostly on the satellite motion and on transmitter and receiver clock rate errors (which affect the L1 and L2 signals similarly after accounting for their frequency difference), one can determine the expected Doppler shift of the L2 signal if one is tracking the L1 signal from the same SV. This is given as

$$F_{dopp,L2} = F_{dopp,L1} * F_{L2} / F_{L1}$$

Similarly, if one knows the L1 C/A code phase at a particular time, one can set bounds on the range of probable start times of the L2C code. The L2 CL code is nominally 1.5 seconds in length, and once every 4 periods its start time is coincident with the start of a data subframe ,which is 6 seconds in length. Since the beginning of data subframes on the L1 C/A signal are being tracked for purposes of data decoding, the receiver starts with these times as the base time for L2C code acquisition; let this time be T_0 for a particular channel.

There are several effects which cause the L2C code start time and the L1 C/A code start times to not be coincident. These effects are often collectively referred to as differential code bias. First, and usually most prominently, there is an unknown inter-frequency bias due to the different signal paths, hardware, and processing on the receiver plus antenna combination. Secondly, there is a similarly unknown inter-frequency bias due to satellite hardware. Let the receiver plus antenna portion of these biases be T_1.

Precisely measuring this receiver bias is a notable challenge that must be addressed for accurate measurements of ionospheric total electron content (TEC). This is an area of active research, and discussion of it can be found elsewhere[13,14]. It can, however, be roughly estimated, which will be shown to be useful for this acquisition technique. It should be noted that the typical value for this receiver (times the speed of light) is on the order of 13 meters.

A third source of differential code bias is the ionosphere. Electrons in the signal path alter the index of refraction, and the amount by which diffraction delays a given signal is inversely proportional to frequency squared. Let this ionosphere-induced delay be T_2. One can very conservatively estimate a maximum value for T_2 by using

a very large value for TEC; 150 TEC units (1 TEC unit = 10^{16} electrons / m^2) gives a differential code delay of T_2 = 52 ns.

Putting this together, the total amount of (code) space that the receiver must search to acquire the L2 CL signal spans T_2, and starts at T_0+T_1. To account for multipath errors, bias estimation errors, and other noise, the receiver expands this search space by a factor of 1.5. This code space is then searched with a brute-force algorithm using some predetermined step size that gives a minimal power loss due to code misalignment. A step size of roughly 0.05 chips has been used.

The L2C signal is composed of the "medium length" (CM) code interleaved with the "long" (CL) code. The CL code is a data-less pilot signal, while the CM code is modulated with data bits (though at the time of writing, CM code does not yet have data bit modulation). If one is interested in tracking only the CL (or CM) code, a method must be devised for separating the two. A naïve approach would be to interleave the desired code with zeros, and then perform the accumulations. However, because this receiver makes use of the bitwise parallel processing technique previously mentioned a value of zero has meaning for our processing algorithms (i.e., it indicates either a low value for sign or magnitude). The solution is to produce two replicas of the code for the period desired. Let these two PRN codes be defined as

$$L2C_+ = CL + CM \text{ and } L2C_- = CL + (!CM)$$

where ! indicates logical inversion and + indicates logical OR. Accumulations are then done with both of these replicas. To recover CL code accumulations, the receiver takes take the sum of $L2C_+$ and $L2C_-$ (and gets 2*CL), and for the CM code it takes the difference (and gets 2*CM). For determining the data bits modulating the CM code, this is the method that would need to be implemented.

If one uses only one of the two resultant pairs of accumulations, either 2*CL or 2*CM, then the tracking PLL and DLL experience a reduction in SNR of 3 dB. For illustration purposes, assume that the L1 C/A and L2C signals are received with exactly the same strength, and that both are subjected to the same intensity of additive white Gaussian noise. By creating accumulations using only CM or CL code, the receiver is using half the integration time as compared to a similar (temporal) length of C/A code since the CM and CL codes individually have a chipping rate half that of the C/A code. Suppose one defines the accumulation Signal-to-Noise ratio (SNR_A) to be

$$SNR_A = C/(N_0*Ba)$$

where C is carrier power, N_0 is noise power density, and Ba is the effective (single-sided) noise bandwidth of the integrate-and-dump operation. If one integrates the sinc function that results from integrate-and-dump integration, then one finds that the effective (single-sided) noise bandwidth Ba is equal to 1/(2*Ti) where Ti is the integration time (also referred to as the pre-detection interval). Substituting this in, one gets

$$SNR_A = (2*Ti*C)/N_0$$

One can then deduce that the SNR_A for the CM or CL codes alone, is half that of the C/A code over the same length of data even though the carrier-to-noise ratio (C/N_0) is the same for all of the signals because the effective integration time Ti for the two L2C codes is half that of the L1 C/A code due to the interleaving of the two codes. Note, however, that this 3dB loss could be avoided in a PLL or DLL which combined accumulations from the two L2 C signals.

There were no data bits modulating the CM code at the time of this writing, so a shortcut was taken. Observation indicates that the CM code is currently being modulated with the a constant +1 data bit value. Therefore, the best of both worlds can be had: only the $L2C_+$ replica needs to be generated and the corresponding accumulations computed. These accumulations have a higher SNR_A than either the CL or CM code alone, and the receiver can use a four-quadrant arctangent PLL discriminator for better tracking robustness. This ad-hoc modification nicely illustrates the flexibility of software receivers.

VI. BENCHMARKING

In this section, we will examine the computational costs of operations being performed and memory requirements.

A. Timing

For performing timing benchmarks the processor used is the aforementioned Texas Instruments C6455 Digital Signal Processor running at 1.2 GHz, with an RF front-end sampling frequency of $F_s \approx 5.714 MHz$ and 2-bit signal quantization. This subsection examines the processing time required for both L1 C/A and L2 C signal acquisition, for tracking of both the L1 C/A and L2 C signals, and for navigation solution computation.

Acquisition time for the L1 C/A signal using a Doppler search range of ± 6000 Hz, A Doppler search step size of 350 Hz, and a 2 ms non-coherent integration time is roughly 60.8 ms per attempt. The details of the acquisition routine used here are identical to those in Ref. 1; only the speed of the processor has increased. With the current hardware, a search of all 32 PRNs can be done in only 1.9 seconds with reliable acquisition down to C/N_0 = 42 dB-Hz.

Tracking the L1 C/A signal for 1 ms takes approximately 10 µs per channel. About three-fourths of this time is spent computing the prompt and early-minus-late in-phase and quadrature accumulations. The rest of the time is used by tracking loop updates, data bit decoding, and assorted bookkeeping operations.

As previously stated, the L2 C acquisition technique is aided by the L1 C/A signal. Acquisition attempts are limited to once per channel per subframe (i.e., once every 6 seconds). The main computation expense related to L2 C tracking is generation of the up-sampled code replicas. Generating one millisecond of L2 CM+CL code currently takes 17 µs; doing all of the other required operations for updating the channel (e.g., accumulations) takes only 10 µs. If one were treating the CM and CL signals separately rather than taking advantage of the lack of data modulation on the CM signals and treating CM+CL as a unified pilot signal, then the computational burden would increase. Time required for code generation would increase to roughly 19 µs, and the time required for performing the accumulations, tracking loop updates, and other required operations would increase to roughly 16 µs.

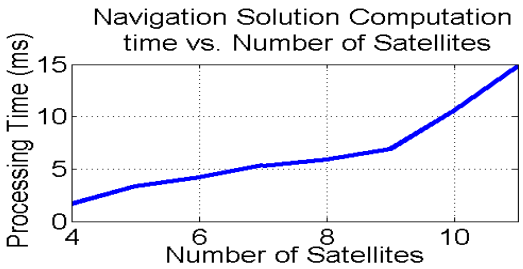

Figure 8. Computation time required for computing a navigation solution versus number of satellites used in the solution.

The navigation solution was implemented using floating point math. Although floating point operations are on the order of 100 times more expensive than fixed point operations on this platform, relatively large execution times can be tolerated if the navigation solution is being computed infrequently (compared to the frequency of operations related to signal tracking). The navigation solution was written to be fully interruptible by other processes that may have real-time deadlines, and the computed processing times shown below include such interrupts. It should be noted that if one is utilizing more satellites for the navigation solution, then more signals are being tracked, and there will be a correspondingly higher number of interrupts during the navigation solution calculations to service the real-time needs of those signals. Thus, the increase in computation time for larger numbers of satellites is due to both the non-linear growth in computational cost for certain operations (e.g., matrix inversion), and the fact that the calculation is interrupted more. The computational burden ranges from less than 2 ms to about 15 ms for 4 and 11 satellites, respectively. See Fig. 8 for a plot of navigation solution computation times. If at some point in the future navigation solutions are required at a rate higher than that possible using the current algorithm the viability of a fixed-point solution will be further explored.

A graphical representation of the distribution of computing time is shown in Fig. 9. This chart assumes the receiver is operating in steady-state (not doing acquisition), tracking 12 L2C signals and 12 L1 C/A signals, and computing position, navigation, and time at a 1 Hz rate using 12 C/A channels.

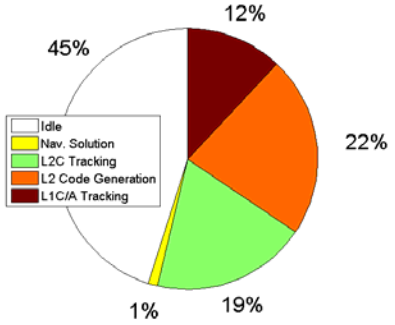

Figure 9. Distribution of computation time per second while tracking 12 L2C and 12 L1C/A channels.

B. Memory Requirements

The DSP currently being used has 2 MB of on-chip random access memory, and 256 MB of off-chip random access memory. There is a significant performance penalty imposed when using data or code stored in off-

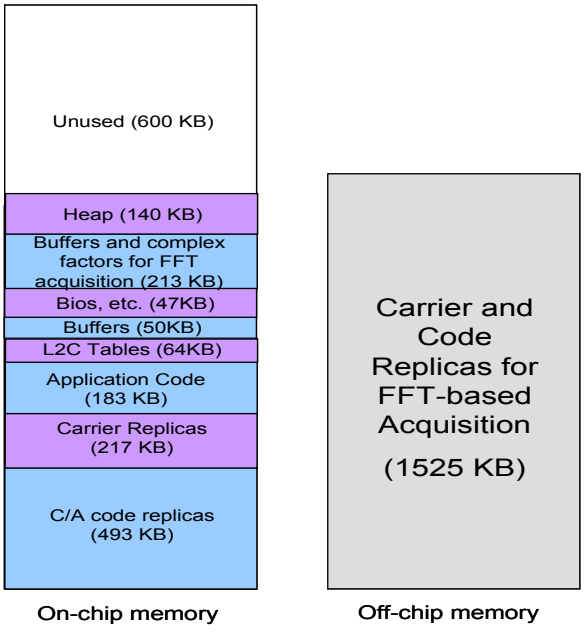

Figure 10. Memory usage for both on-chip and off-chip memory.

chip memory, so it behooves the developer to attempt to fit as much as possible into on-chip memory. The only things not stored in on-chip memory are the code and carrier replicas used in the FFT acquisition routine. The memory usage is shown in Fig. 10. Note that almost the entirety of off-chip memory is not being used (~253 MB), and there is ample room left in on-chip memory. The entire on-chip memory footprint is only roughly 1.4 MB.

If memory becomes a constraint in the future, L1 C/A code generation could be done using the same scheme being applied to L2 C code generation, at the cost of additional computational expense.

VII. FIRST FIELD TEST RESULTS

A week-long field campaign was conducted in Natal, Brazil during January, 2009 with the hopes of observing (simultaneous) scintillation of the L1 C/A and L2 C signals. Observations were made using the prototype DSP-based receiver that is the subject of this paper, as well as two "digital storage receivers" and a Cornell SCINTMON receiver. The digital storage receivers use the exact same RF front-end as used in the DSP receiver, but the data are stored on a hard drive for later processing. These data were processed using the same code running on the DSP, but not in real-time, and on a personal computer. All operating parameters were identical (e.g., tracking loop bandwidths, integrations times), with the sole exception being that the observables were available at a higher rate because there was no communications bandwidth constraint when not operating in real-time. A diagram of the receiver locations is shown in Fig. 11. The digital storage receivers are indicated with "DSR."

Figure 11. GPS receiver locations in Natal, Brazil (5.836° W, 35.207 ° S)

Moderate scintillation of signals from satellites transmitting the L2 C codes was observed on all three receivers, with the largest S4 index of the scintillation being around 0.6. Plots of carrier-to-noise (C/N_0) ratio for the three receivers during a period of such scintillation are shown in Fig. 12. In this plot, receivers 1 and 3 are digital storage receivers (50 Hz amplitude measurements), and receiver 2 is the DSP receiver (10 Hz amplitude measurements). The amplitude fades seen by all three receivers are quite similar, and show a slight time lag with the fades appearing first on receiver 1 and propagating eastward to receivers 2 and 3 after slight delays. It is believed that the apparent higher level of noise in the receiver 3 C/N_0 data is due to the antenna environment for this receiver. Although the fades are quite deep in places (exceeding 25 dB on L2), none of the receivers lost lock on the signal.

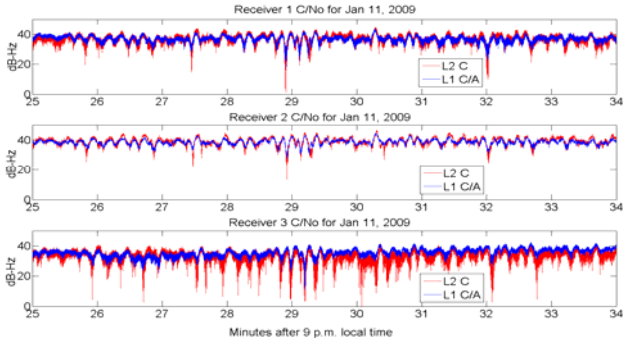

Figure 12. Amplitude scintillations of the L1 C/A and L2 C signals from PRN 15 observed by three dual-frequency receivers.

Measurements of phase-derived TEC were calculated as follows:

$$TECU = \frac{F_{L1}^2 \cdot F_{L2}^2}{40.3(F_{L1}^2 - F_{L2}^2)10^{16}}(\lambda_{L1}\varphi_{L1} - \lambda_{L2}\varphi_{L2} + N)$$

Where TECU indicates total electron content units (1 TECU = 10^{16} electrons / m^2), F, φ and λ indicate frequency, carrier phase and wavelength on either L1 or L2, respectively, and N indicates the unknown difference in initial phase between the L1 and L2 signals. The presence of N means that when using phase to measure TEC, there is an unknown (and possibly large) bias in the estimate.

This simple formula for TECU assumes that the phase scintillations are caused entirely by fluctuations in a presumed uniform "bulk" TEC of the ionosphere. In reality, fine-scale spatial TEC variations and the effects of diffraction imply that the true TEC is not quite equal to this computed value [15]. Nevertheless, this pseudo-TEC provides a useful indication of the phase effects of scintillation.

A plot of phase-derived differential TEC is shown in Fig. 13. The plotted TEC has been band-pass filtered with a pass-band of 0.01 – 1.0 Hz to remove the background TEC and high-frequency (measurement) noise. This plot shows variations in TEC as measured by receivers 1 and 3 over a roughly 20 minute period, and was taken in the absence of measurable amplitude scintillation. A high degree of correlation between the TEC fluctuations measured on the two receivers can be seen, again with a slight time lag between the two.

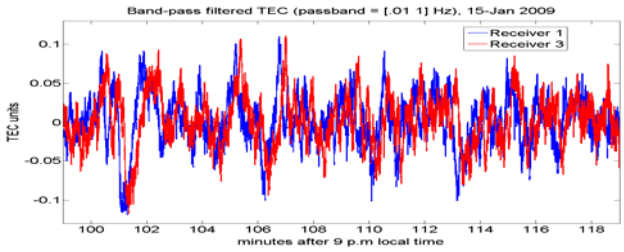

Figure 13. Band-pass filtered phase-derived TEC as measured by two receivers.

This limited field campaign has shown promising results in terms of the receiver operation and the quality of its observables in a scintillating environment.

VIII. CONCLUSIONS

A civilian dual-frequency GPS receiver has been implemented on a DSP and tested both in the field in scintillating and non-scintillating conditions, and in the laboratory. Tracking of the L2 C code has been added, and both position and velocity are being computed on the DSP. Approximately 55% of the available CPU cycles and 75% of the on-chip memory are being used. Reliable tracking of the L1 C/A and L2 C codes down to C/N_0 = 25 dB-Hz has been demonstrated.

ACKNOWLEDGMENTS

This work was generously supported in part by grant No. NNX08AM33G from NASA, grant No. N00014-09-1-0295 from the Office of Naval Research, and grant No. ATM-0720209 from the NSF.

REFERENCES

[1] Humphreys, T.E., Psiaki, M.L., Kintner, Jr., P.M, Ledvina, B.M., "GNSS Receiver Implementation on a DSP: Status, Challenges, and Prospects," *Proc. 2006 ION GNSS Conf.*, Institute of Navigation, Fort Worth TX, pp. 2370-2382.

[2] Ledvina, B.M., Psiaki, M.L., Sheinfeld, D.J., Cerruti, A.P., Powell, S.P., and Kintner, Jr., P.M. "A Real-Time GPS Civilian L1/L2 Software Receiver," *Proc. 2004 ION GNSS Conf.,* Institute of Navigation, Long Beach, CA, pp 986-1005.

[3] Akos, D. M., Normark, P., Hansson, A., Rosenlind, A., Stahlberg, C., and Svensson, F., "Global Positioning System Software Receiver (gpSrx) Implementation in Low Cost/Power Programmable Processors," *Proc. 2001 ION GPS Conf.*, Institute of Navigation, Salt Lake City, UT, September 2001, pp. 2851-2858.

[4] Won, J.-H., Pany, T., and Hein, G.W., "GNSS Software Defined Radio," *Inside GNSS*, Vol. 1, No. 5, July 2006, pp. 48-56.

[5] Anon., "Boeing Satellite Launch Schedule," The Boeing Company, September 25, 2009, http://www.boeing.com/defense-space/space/bss/launch/launch_sched.html

[6] Ledvina, B.M, Psiaki, M.L., Powell, S.P., and Kintner, Jr., P.M. "Real-Time Software Receiver Tracking of GPS L2 Civilian Signals using a Hardware Simulator," *Proc. 2005 ION GNSS Conf.,* Institute of Navigation, Long Beach, CA, pp. 1598-1610.

[7] Ledvina, B. M., Psiaki, M. L., Powell, S. P., and Kintner, Jr., P. M., "Bit-Wise Parallel Algorithms for Efficient Software Correlation Applied to a GPS Software Receiver," *IEEE Transactions on Wireless Communications*, Vol. 3, No. 5, September 2004.

[8] J. A. Klobuchar, "Ionospheric Effects on GPS," in Global Positioning System: Theory and Applications, Vol. I, B. W. Parkinson and J. J. Spilker Jr. , Eds., American Institute of Aeronautics and Astronautics, (Washington, 1996), pp. 485–515.

[9] Saastamoinen, J., "Contributions to the Theory of Atmospheric Refraction," *Bulletin Géodésique*, Vol. 105, September 1972, Vol. 106, December 1972, Vol. 107, March 1973.

[10] Niell, A.E., "Global mapping functions for the atmosphere delay at radio wavelengths," *Journal of Geophysical Research*, Vol. 101, No. B2, February, 1996, pp. 3227-3246.

[11] Beach, T.L. and Kintner, Jr., P.M., "Development and Use of a GPS Ionospheric Scintillation Monitor," *IEEE Transactions on Geoscience and Remote Sensing,* Vol. 39 No. 5, May 2001 pp 918-928.

[12] Psiaki, M. L., "Real-Time Generation of Bit-Wise Parallel Representations of Over-Sampled PRN Codes," *IEEE Transactions on Wireless Communications*, Vol. 5, No. 3, March 2006, pp. 487-491.

[13] Komjathy A., Sparks, L., Wilson, B.D., Mannucci, A.J., (2005), "Automated daily processing of more than 1000 ground-based GPS receivers for studying intense ionospheric storms," *Radio Science*, Vol. 40, RS6006, doi:10.1029/2005RS003279.

[14] Coster, A., and S. Skone (2009), "Monitoring storm-enhanced density using IGS reference station data", *Journal of Geodesy*, Vol. *83, No.* 3-4, March 2009, pp. 345-351.

[15] Psiaki, M.L., Bust, G.S., Cerruti, A.P., Kintner, P.M., Jr., and Powell, S.P., "Diffraction Tomography of the Disturbed Ionosphere Based on GPS Scintillation Data," *Proc. of the ION GNSS 2008*, Sept. 16-19, 2008, Savannah, GA, pp. 289-308.