

SOFTWARE DEFINED GLOBAL POSITIONING SATELLITE RECEIVER

Daniel Iancu (Sandbridge Technologies Inc., White Plains, NY 10601 USA; diancu@sandbridgetech.com); John Glossner (Sandbridge Technologies Inc. & Delft University of Technology, Computer Engineering, Electrical Engineering Mathematics and Computer Science, Delft, The Netherlands; jglossner@sandbridgetech.com); Vladimir Kotlyar (Sandbridge Technologies Inc., White Plains, NY 10601 USA; vkotlyar@sandbridgetech.com); Hua Ye (Sandbridge Technologies Inc., White Plains, NY 10601 USA; huaye@sandbridgetech.com); Mayan Moudgill (Sandbridge Technologies Inc., White Plains, NY 10601 USA; mayan@sandbridgetech.com); Erdem Hokenek (Sandbridge Technologies Inc., White Plains, NY 10601 USA; ehokenek@sandbridgetech.com);

ABSTRACT

Due to the computational complexity of Global Positioning Satellite (GPS) receivers, they have been traditionally implemented in hardware (H/W) employing multiple parallel channels. Usually each channel is responsible for tracking and demodulating one satellite. In some designs, one channel can be time shared by more than one satellite. In the case of multi protocol communication systems, a HW implementation becomes less attractive due to additional chip area requirements. In this paper we present a software (S/W) implementation of a GPS receiver. It is part of a software communication baseband processor being designed by Sandbridge Technologies. In our design all functions associated with the GPS receiver are implemented in S/W on the Sandbridge Technologies' Sandblaster Multithreaded SB9600 processor. The device is intended to be used in hand held devices as mobile phones and PDA's.

1. INTRODUCTION

A received GPS signal can be viewed as a superposition of N_s DS-CDMA signals coming from N_s visible satellites. Each satellite has a unique signature and slightly different carrier frequency due to the Doppler Effect [1] [2]. The GPS composite signal, for the CA code, can be modeled as

$$s(t) = \sum_{i=0}^{N_s-1} \sum_{n=-\infty}^{+\infty} \sum_{k=-\infty}^{+\infty} A_i d_i[k] g\left(t - k \frac{N_p}{f_i}\right) g\left(t - n \frac{1}{f_i}\right) \times P_i[(n + n_i) \% N_p] \cdot \cos(2\pi f_i t + \varphi_i) + \eta(t) \quad (1)$$

Where: i stands for the i^{th} satellite, $g(t - mT_i) = \sigma(t - mT_i) \sigma[(m+1)T_i - t]$, $f_i = 1/T_i$ is the carrier frequency, where

$$\sigma(t) = \begin{cases} 1 & \text{for } t \geq 0 \\ 0 & \text{for } t < 0 \end{cases}$$

is the unit step function, $d_i[k]$ is the data in the k^{th} msec, $P_i[n]$ is the n^{th} chip in the k^{th} millisecond, A_i is the amplitude,

N_p is the number of chips in the PN sequence (1023 for the CA code), and $\eta(t)$ is the thermal noise.

Without losing generality, the noise term in (1) will not be considered further. In order to extract the payload information embedded in the composite signal, for one satellite, we window $s(t)$ with the pulse functions $g(t-t')$ yielding:

$$\chi(t) = \sum_{i=0}^{N_s-1} \sum_{n=-\infty}^{+\infty} \sum_{k=-\infty}^{+\infty} \xi_{i,k,n} \cdot g\left(t - k \frac{N_p}{f_i}\right) g\left(t - k \frac{N_p}{f_i}\right) g\left(t - m \frac{1}{f_i}\right) \times x g\left(t - n \frac{1}{f_i}\right) \cdot \cos(2\pi f_i t + \varphi_i) e^{-j2\pi f t} \quad (2)$$

where $j = \sqrt{-1}$, f is the carrier frequency at the transmitter (the same for all satellites), $\%$ is the mod operation, and

$$\xi_{i,k,n} = A_i d_i[k] P_i[(n + n_i) \% N_p]$$

We can view f as the Local Oscillator (LO) frequency, equal to the IF frequency in the receiver. In equation (2) the product of pulse functions $g(t-t')$ will be non-zero if and only if $k=k'$ and $m=n$. This implies that for a specified $k=k'$ and $m=n$ there will be a single chip selected on the time scale.

After summation and integration, for all visible satellites, Equation (2) becomes

$$\sum_{i=0}^{N_s-1} \int_{-\infty}^{+\infty} \chi(t) dt = \sum_{i=0}^{N_s-1} \left(\frac{D_i(k, n + n_i) \cos \varphi_i}{2} \delta(f - f_i) + j \frac{D_i(k, n + n_i) \sin \varphi_i}{2} \delta(f - f_i) \right) + \sum_{i=0}^{N_s-1} E_i \quad (3)$$

where E_i is the total error caused by the windowing with the pulse functions due to lack of satellite-receiver clock synchronization and Doppler shift, and

$$D_i(k, n + n_i) = A_i d_i[k] P_i[(n + n_i) \% N_p]$$

2. IMPLEMENTATION

In equation (3) we desire to minimize the detection error for a particular satellite i . Therefore we force both conditions to $f - f_i = 0$ and $\varphi_i = 0$ implying a correction for the Doppler shift as well as for the phase shift for each satellite. Conforming to the Fourier transform shifting property, the condition $f - f_i = 0$ can be achieved either through frequency or time domain shifting.

In H/W implementations, the carrier is tracked by advancing or retarding the LO frequency and phase (frequency domain shift) conforming to the output of a Phase Locked Loop (PLL) circuit. For each visible satellite the integral defined in (3) is calculated separately resulting in multiple parallel processes with each process executed by a separate dedicated H/W block called a channel. Each channel must have its own LO, PLL and Pseudo Number (PN) generator.

A software implementation is more efficiently implemented in the time domain. The sampled data is time domain shifted by pointer manipulation; the pointer is shifted forward or retarded depending on the direction of the Doppler shift. The LO in this case is a sin-cosine table and is the same for all carrier frequencies. The phase condition is achieved by a digital (software) PLL, resulting in an additional shift on the already shifted data for each carrier separately. The PN sequences are stored in memory.

In a multithreaded environment, all the integrals in (3) are executed in parallel. The number of virtual channels is dynamically allocated by the processor, depending on the available resources and real time criteria.

Hardware can easily implement the condition $f - f_i = 0$ with a Voltage Controlled Oscillator (VCO) used for LO. In software, the frequency condition cannot be exactly achieved due to a limited sampling rate. Shifting in the time domain will also induce spurious frequencies which will increase the detection error. To minimize the detection errors, oversampling is required. It can be shown that the total energy of the spurious frequencies is converging to zero at a rate of $\frac{1}{N^2}$, where N is the number of over-samples per cycle (if we assume n cycles/chip and k samples/cycle then the total number of samples/chip will be $N = n k$).

3. SYSTEM DESIGN

The H/W platform used for implementation is shown in Figure 1. The RF signal is down converted to an intermediate frequency (IF1) by the Maxim MAX2740 GPS receiver chip. The required reference frequency is provided by a fractional PLL clocked by a high precision low drift crystal oscillator. The PLL also provides the input frequency for the mixer to further down convert the to a 4.092 MHz intermediate frequency. The output of the mixer is filtered and digitized at 32.736 Msps. The Sandblaster processor also performs the Automatic Gain Control (AGC) function in software and provides the AGC control signal.

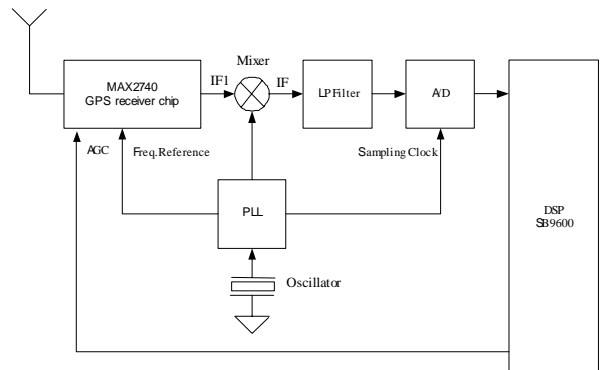


Figure 1. HW block diagram

3.1. BASE BAND PROCESSING

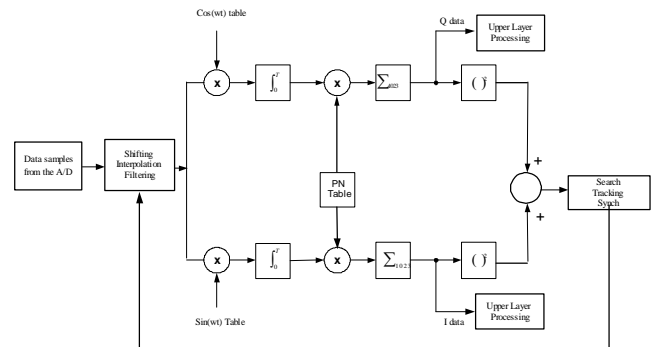


Figure 2. Receiver base band block diagram

The simplified block diagram of the receiver, for a single satellite, is depicted in Figure 2. A few milliseconds worth of data samples, received from the Analog-to-Digital (A/D) converter are buffered and processed separately for each satellite. First, the data is passed to the shaping, interpolation, and filtering block. The functions performed by this block are illustrated in Figure 3. In order to meet the frequency condition described in Section 2, the data is shifted one sample at a time every n samples, conforming to

the required Doppler shift. Also, to keep the carrier phase locked, an extra one sample shift can be performed, if it is indicated by the digital PLL. To increase the Delay Locked Loop (DLL) resolution, the block can perform sample interpolation using one of a number of well known interpolation techniques. In this system, we employed linear interpolation. The data is then filtered using an FIR filter and further passed to the demodulator. The demodulation is performed on the already shifted data using a sine/cosine lookup table with eight samples per cycle. The integration over eight samples provides one I/Q sample and simultaneously performs eight times decimation. The correlation with the PN sequence provides one data sample. The I/Q data is subsequently processed based on the intended function (e.g. Satellite search, I^2+Q^2 , PLL, IxQ , DLL, I^2+Q^2 , or De-spreading Q).

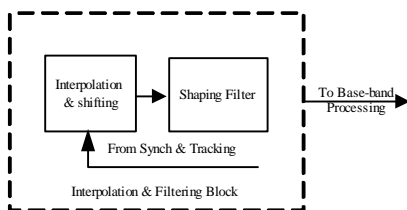


Figure 3. Interpolation, shifting and filtering block.

3.2. SKY SEARCH

Some heavily computational functions do not require real-time processing. Sky search is an example of this type of function. In the case of the sky search function, the time is tracked using a high resolution timer, driven by the crystal oscillator, such that the next buffer of data is precisely placed in time.

The sky search function is performed on a few tens of milliseconds of data. If the satellite constellation is known, the search is performed only for the PN numbers of the expected visible satellites and a few Doppler frequency bins around the expected Doppler frequencies. If the constellation is not known, the search is performed for every PN number and every Doppler frequency bin. The output from the demodulator, In and Quadrature phase, are correlated with a 1023 length PN sequence at one sample per chip. It is then squared and accumulated over several milliseconds. The result is compared against a threshold. If the result is higher than the threshold, the despreading process is started with the maximum correlation sample position. The despread data is correlated with an eight bit preamble repeated in every navigation data subframe. If the preamble is found then the search is successful and the data is further processed. Otherwise the search starts again.

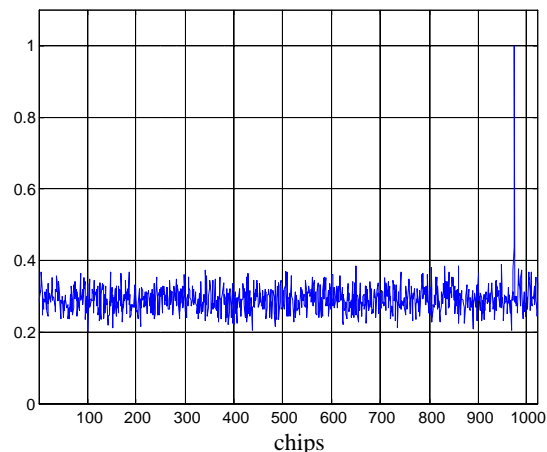


Figure 4. Satellite search, cross correlation output for PN 1, 6 satellites in view, -30 dB SNIR.

Figure 4 illustrates the search results for a constellation of six satellites at equal power and -30 dB signal to noise and interference ratio (SNIR). It shows about 1000 chips being processed and our software implementation correlating and locking onto the signal.



Figure 5. Digital PLL block diagram.

3.3. DIGITAL PLL

The digital PLL block diagram is shown in Figure 5. The PLL input consists of I and Q data. The output is -1, 0, or +1 where +1 means advance one sample, 0 do not shift and -1 means delay one sample.

Figure 6 and Figure 7 show the despread data with the PLL turned OFF and ON, respectively. In Figure 6 it can be seen that because of the accumulated errors in the shifting process the data cannot be properly decoded without the digital PLL running. As the digital PLL is turned ON, the decoded data sequence can be easily seen in Figure 7.

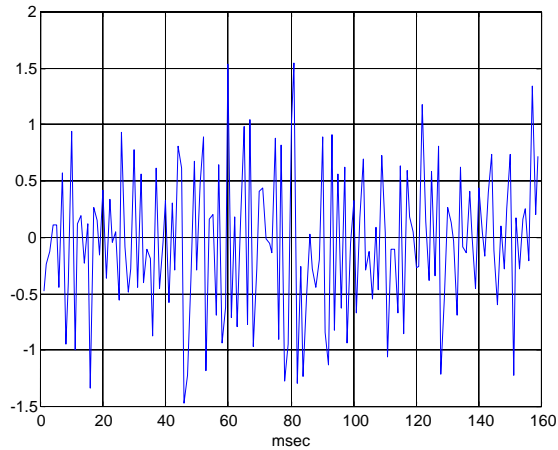


Figure 6. Decoded data, Doppler shift 25 Hz, PLL turned OFF, SNIR -31 dB.

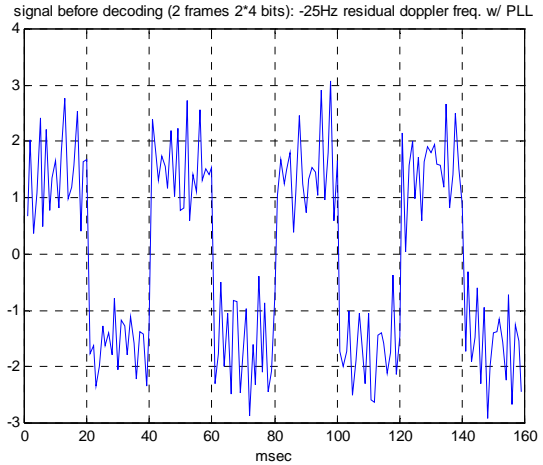


Figure 8. Decoded data, Doppler shift -25 Hz, PLL turned ON, SNIR -33 dB, with additional filtering.

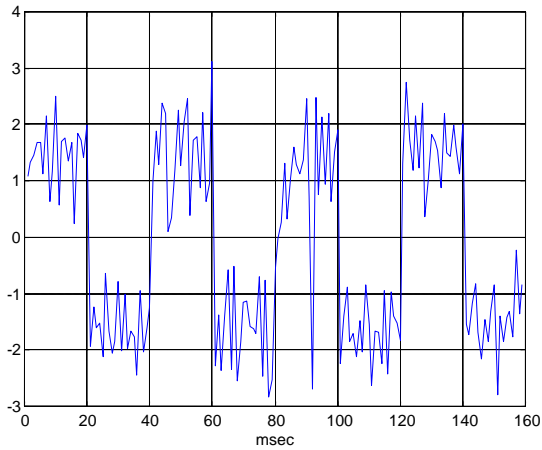


Figure 7. Decoded data sequence {+1, -1, +1, -1, ...}; Doppler shift 25 Hz, PLL turned ON, SNIR -31 dB.

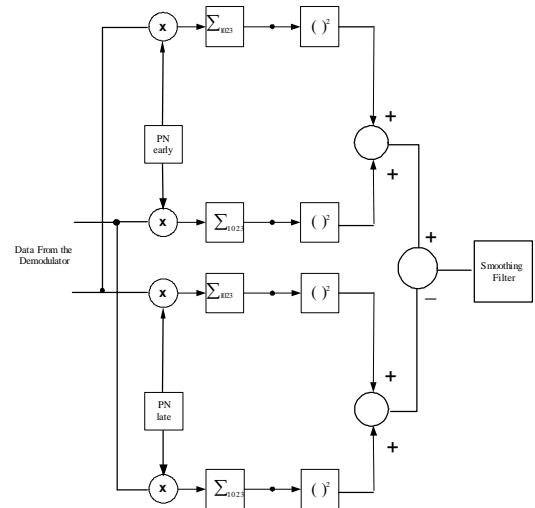


Figure 9. Digital DLL block diagram.

In the case when the received signal experiences lower power levels due to harsh propagation conditions, additional filtering can dramatically improve the integrity of the data. In our S/W approach we can dynamically add or remove digital filters in the processing chain based upon the value of the measured SNIR for a specific satellite. Figure 8 illustrates the decoded data for -33dB SNIR when additional software Band Pass Filter (BPF), tuned on the IF frequency with 2 MHz bandwidth, is added. With a signal power level 2 dB less (-33 dB), it can be seen that the variance on the data is smaller than in the -31 dB SNIR case.

3.4. DIGITAL DLL

The role of the digital DLL is to detect the subframe boundary at less than four samples per chip resolution in order to improve the user position accuracy. The block diagram of the digital DLL is shown in Figure 9. It can be seen that there are two identical branches executing the same function, except for the fact that the PN sequence in one branch is advanced half a chip, while in the other branch it is delayed half a chip. The data flow is as follows: I and Q data from the demodulator is correlated with the early and late versions of the PN sequence for a specific satellite. It is then squared and subtracted as described in following formula and finally filtered by a smoothing filter.

$$(I_e^2 + Q_e^2) - (I_l^2 + Q_l^2)$$

where : I_e, I_l, Q_e, Q_l are the early and late I and Q.

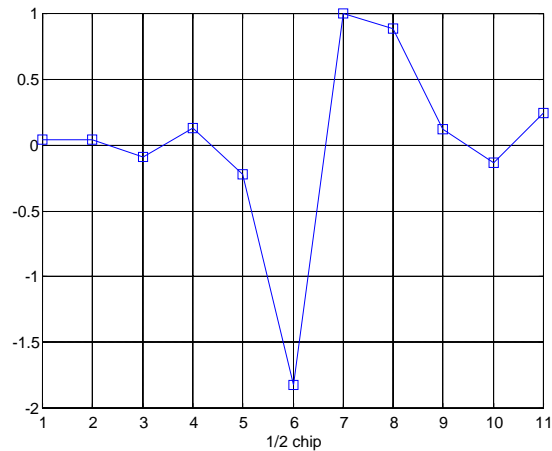


Figure 10. Output of the digital DLL.

The filtered output is shown in Figure 10. It can be easily seen that the curve exhibits a minimum and a maximum corresponding to the early and late correlations, respectively. The x axis intercept represents the subframe boundary, related to the two neighboring chips. This more accurate frame boundary provides a more precise time positioning for the incoming sub frames, and ultimately provides a better user position estimate.

4. RESULTS

To determine the user position in three dimensions (x_u , y_u , z_u), a pseudorange measurement needs to be performed for four satellites. In our approach we assume the user position is approximately known so we can iteratively use a first order approximation for solving the nonlinear system of equations. After the first iteration, the new position is used as initial approximate position and so on. After a few iterations, the final value of the position is obtained.

We implemented all baseband processing for the GPS receiver on the Sandbridge Sandblaster SB9600 multithreaded platform [4] [5]. The performance for different position accuracies is shown in Figure 11. We characterized the receiver performance for various update periods. For high accuracy with fast response (i.e. 5 meters and 100msec update response), about 1200MHz of processing is required. For more typical conditions (i.e. 75 meters and 500msec update response), only 400MHz of processing is required. This equates to a 50% and 16% utilization of the SB9600 platform.

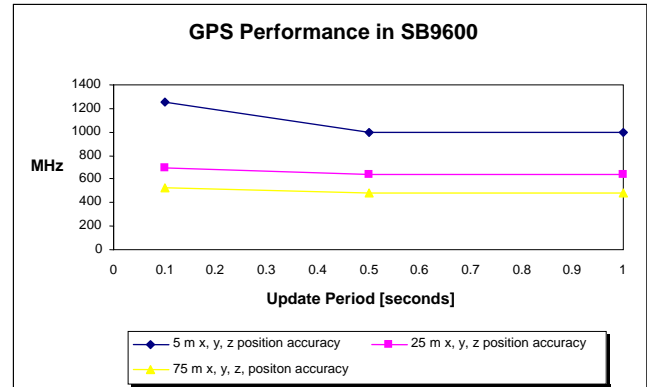


Figure 11. GPS receiver performance

5. CONCLUSIONS

We have presented a software GPS design that provides for dynamic trade-offs in performance versus accuracy. The design is implemented completely in software and will run in real time on the SB9600 platform.

6. REFERENCES

- [1] Elliot D. Kaplan, "Understanding GPS Principles and Applications", *Artech House Inc. 1996*.
- [2] Global Positioning System Standard Positioning Service Signal Specification, GPS NAVSTAR 2nd Edition, June 2, 1995
- [3] E. O. Bringham, "The Fast Fourier Transform And Its Applications", *Prentice-Hall Inc. 1988*.
- [4] J. Glossner, S. Dorward, S. Jinturkar, M. Moudgill, E. Hokenek, M. Schulte, and S. Vassiliadis, "Sandbridge Software Tools", in Proceedings of the 3rd International Workshop on Systems, Architectures, Modeling, and Simulation (SAMOS'03), July 21-23, 2003, pp. 142-147, Samos, Greece.
- [5] J. Glossner, D. Iancu, J. Lu, E. Hokenek, and M. Moudgill, "A Software Defined Communications Baseband Design", *IEEE Communications Magazine*, Vol. 41, No. 1, pp. 120-128, Jan., 2003.