

Keywords: GPS, Global Positioning System, Galileo, receiver, LNA, mixer, ADC, RF front end, low noise amplifier, USB dongle, PCI Express, software baseband, soft baseband, downconvert, base band, L1, C/A code,

## APPLICATION NOTE 4237

# Universal GPS Receiver Lets You Use a Laptop PC for Soft Baseband Processing

By: David Weber, Strategic Applications Engineer  
Roger Bremer, Strategic Applications Engineer

Jun 11, 2008

*Abstract: This application note investigates the basic theory of software-based receivers for L1-band civilian GPS applications, including a discussion of signal acquisition and tracking, and the need for bit synchronization in the receiver. It then details a low-cost method for adding GPS capability to laptop PCs through a USB dongle with the MAX2769 universal GPS receiver.*

A similar version of this article was featured in [Maxim's Engineering Journal](#), vol. 63 (PDF, 1.3MB).

## Overview

Communication and navigation engineers are increasingly interested in using software techniques for global positioning systems (GPS).<sup>1,2</sup> Thanks to very-large-scale integration (VLSI) development, powerful CPUs and DSPs can now detect and decode GPS signals in real time using software. The resulting software-based GPS receivers offer considerable flexibility in modifying settings to accommodate new applications without redesigning hardware, choosing an IF frequency, or implementing future upgrades.

With the [MAX2769](#) GPS receiver RF front-end, a simple USB dongle or PCI Express® (PCIe®) Mini Card format can be used to add low-cost GPS capability to laptop PCs. The MAX2769 transfers raw recovered data to the PC host, which executes the baseband decoding in software, thus eliminating the cost of the baseband ASIC typically required in stand-alone GPS systems. In short, the MAX2769 enables designers to implement an adapter capable of handling GPS and Galileo signals using a single chip.

This article provides an operational overview of GPS systems and, then, details Maxim's software-based GPS receiver solution.

## GPS Basics

A GPS system consists of 24 space satellites or space vehicles (each identified by a unique pseudorandom-noise code), a ground-control station, and user equipment (receivers). For civilian GPS and Galileo applications, the satellites communicate over the L1 band located at 1.57542GHz.<sup>3</sup> A GPS receiver requires line-of-sight visibility of at least four satellites to establish a reliable position. The acquisition and tracking of



[Click here](#) for an overview of the wireless components used in a typical radio transceiver.

signals is very complex because each one varies with time and receiver location.

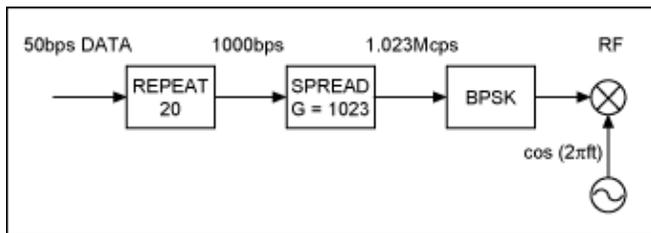


Figure 1. The signal generation block for the civilian GPS signal uses a spread-spectrum approach for sending the signal.

## GPS Signaling

The GPS system is actually a simple spread-spectrum communication system.<sup>4</sup> **Figure 1** shows the signal generation block for civilian applications. First, the 50bps navigation message is repeated 20 times to produce a 1000bps bit stream. The repeated signal is then spread by a unique coarse/acquisition (C/A) code with a length of 1023 chips—a chip is the rate at which the pseudorandom-noise (PRN) code is applied. The result is a baseband signal of 1.023Mcps. Hence, the 43dB processing gain (G) of the GPS system permits it to resolve a signal well below the thermal-noise level.

Each satellite is assigned a unique C/A code, also called a Gold code.<sup>5</sup> Because the Gold code exhibits excellent auto- and cross-correlation properties, it is widely used in CDMA communication systems, such as WCDMA and cdma2000®. The baseband signal is modulated with binary phase-shift keying (BPSK), and upconverted to the L1 band for transmission.

## Analysis of Signal-Acquisition Methods

Because GPS is a CDMA communications system, the receiver must synchronize the PRN code as a prerequisite to demodulating the data. Code synchronization is usually achieved in two steps: code acquisition for coarse-code alignment, and code-phase tracking for fine alignment.<sup>6</sup> More explicitly, a GPS receiver must first determine whether it has line-of-sight visibility to certain satellites or not. As we know, each satellite is distinguished by a unique C/A code. When the satellite is visible, acquisition determines the signal's frequency and code phase, which in turn establishes the corresponding demodulation parameters. The received-signal frequency varies due to the Doppler effect,<sup>7</sup> which causes the frequency to deviate from its nominal value by 5kHz to 10kHz, depending on the speed of the satellite with respect to the receiver.

In the receiver, the GPS signal is first downconverted to in-phase and quadrature (I and Q) components. A pair of I/Q correlators then correlate the I and Q baseband signals with the locally generated PRN sequence. After integrating over the duration of one bit, the I-Q correlator outputs are summed to provide an output-decision variable.

Whenever the decision variable exceeds a certain threshold value, the system assumes that the corresponding acquisition was successful and proceeds to tracking mode. Otherwise, the relative phase of the locally generated PRN sequence and the oscillator frequency are adjusted to update the decision variable, and the above process is repeated.

While the simple logic structure of this serial-search method makes it feasible for implementation in an ASIC, it is not practical for software implementations because the search space is huge. Assuming that the system tolerates a 500Hz carrier-frequency offset and the Doppler frequency is 10kHz, the search space for a software implementation is roughly  $2 \times (10,000/500) \times 1023 = 40,920$ . Obviously, a serial-search acquisition would be difficult in software.

A simpler acquisition method to implement with software is frequency-domain, parallel code-phase acquisition. This method combines the Doppler-frequency and code-phase searches into one search, which, after a fast Fourier transform (FFT) of the PRN code, reflects all code-phase information into the frequency domain. We then need only to search the space over the Doppler-frequency offset, thereby implementing a fast and effective software search.

The system implements this search by first multiplying the incoming signal with the locally generated sine and cosine carrier waves (the I and Q signal components, respectively). The I and Q components are then combined as a complex input to an FFT block. The result of this Fourier transform is multiplied with the conjugate of a PRN code's FFT transform (the PRN generator generates a code with zero-code phase). In practice, the FFT operation and generation of PRN code can be tabulated to reduce computation complexity.

Finally, the product of the incoming signal and local code, which represents the correction between the incoming and carrier frequencies, is applied to an inverse Fourier transform whose squared output feeds back to the decision logic. The FFT-based frequency domain has proven to be a low consumer of computational resources. For the example mentioned earlier, the complexity of acquisition is roughly  $20,000/500 = 40$  FFT operations.

The serial-search method has the simple logic and control architecture necessary for a convenient ASIC implementation. However, the huge search space required for this method imposes additional complexity on the software algorithm. The serial-search method, therefore, is not a good choice for software GPS receivers. In contrast, the low complexity of the parallel-code acquisition method makes it ideal for software implementation. Its logic architecture, however, is far more complex than that of the serial-search method, making it difficult to implement in an ASIC.

## Tracking Refines Alignment

Acquisition establishes a coarse alignment of the GPS signal's frequency and code-phase parameters. The purpose of tracking, therefore, is to refine this alignment so that the system can demodulate the data with exact code-phase and frequency information. Tracking includes code-phase tracking and carrier-frequency tracking.

Code tracking is accomplished with the delay-lock loop (DLL) shown in **Figure 2**. The DLL circuit multiplies the incoming signal by three local replicas of the PRN code (positioned in time at  $\pm 0.5$  chip); these replicas represent early, prompt, and late arrivals with respect to the incoming signal. After integration, each of these signals represents a correlation between the incoming signal and a local replica. The one with the highest correlation value is then selected and retained (**Figure 3**).

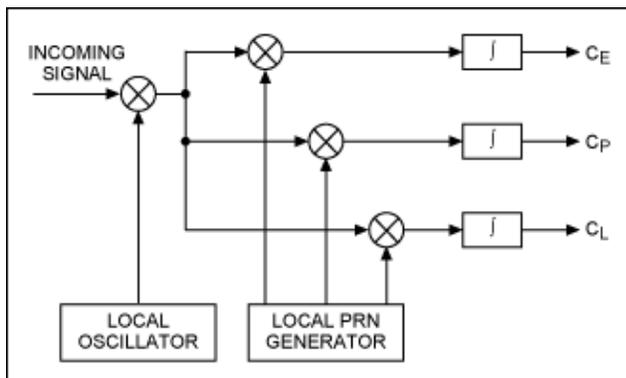


Figure 2. A delay-locked loop is used in the code-tracking phase to help refine the alignment to better demodulate the data with exact code-phase and frequency information.

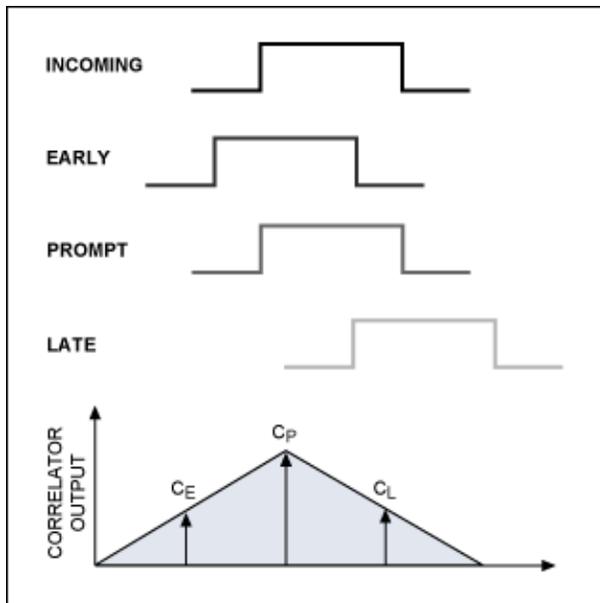


Figure 3. The DLL circuit multiplies the incoming signal by three local replicas of the PRN code (positioned in time at  $\pm 0.5$  chip), which represent early, prompt, and late arrivals with respect to the incoming signal. The one with the highest correlation value is then selected and retained.

Carrier-frequency tracking is carried out by a phase-lock loop (PLL) or Costas loop.<sup>8</sup> The purpose of carrier tracking is to tune the locally generated frequency to the exact frequency of the incoming signal.

After acquisition and tracking have established the initial synchronization, the system can decode the navigation bits. Data demodulation begins by despreading the 1.023Mcps input signal to a 1000bps bit stream. Bit synchronization is then invoked to recover the 50bps information from the 1000bps stream. For bit synchronization, we first need to identify the beginning of a bit in time. This is accomplished by finding the zero-crossing edge (at zero volts), which indicates the beginning of a bit. When that edge is known, we can partition the 1000bps input stream at 20ms intervals, knowing that the duration of a navigation data message (50 bits) is 20ms. Finally, the bit samples in a 20ms interval are summed and averaged to decode the navigation data.

## Software-Based GPS Receivers

Whereas traditional GPS receivers implement acquisition, tracking, and bit-synchronization operations in an ASIC, software-based GPS receivers offer added flexibility by implementing these functions in software. By simplifying the hardware architecture, software makes the receiver smaller, less costly, and more power efficient. The software code can be written in C/C++, MATLAB®, and other languages, and ported into all operating systems (embedded OS, PC, Linux, and DSP platforms). Thus, software GPS receivers offer the greatest flexibility for mobile handsets, PDAs, and similar applications.

For laptop computers, designers can design a USB dongle to work with any laptop with USB ports. For newer laptops with a PCIe Mini Card connector, they can put an RF front-end on the PCIe Mini Card and embed the card in the PC (**Figures 4a, 4b**). The PCIe Mini Card interface includes a USB port, so the design of the front-end adapter is similar for both USB and PCIe Mini Cards. The main difference is in some of the power-management logic required to support PCIe and the differing DC voltages (3.3V for PCIe and 5V for external USB ports).

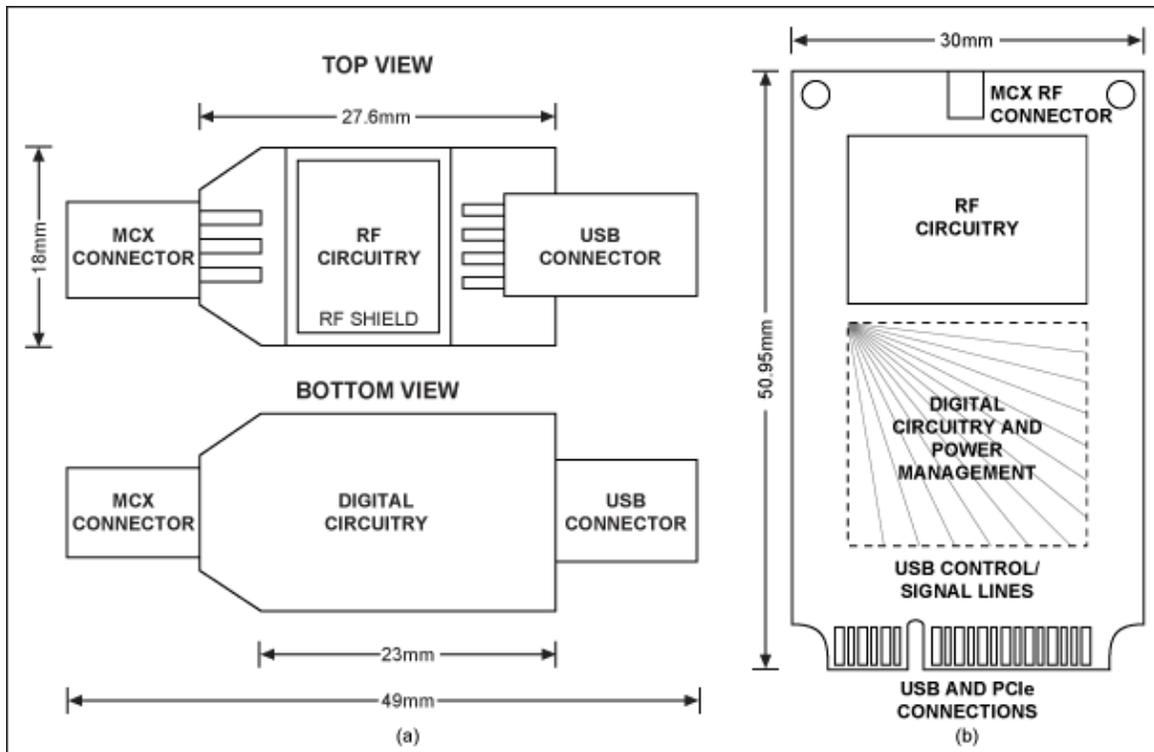


Figure 4. Typical adapter configurations for the USB dongle (a), and the PCIe Mini Card (b) show the simple, low-cost design for either option.

**Figure 5b** shows the block diagram of a USB dongle, demonstrating how simple the solution can be. This circuit employs a MAX2769 universal GPS receiver, a counter, and a USB interface controller to capture the signal, convert it to digital, and send it to the host PC. Software then performs all the baseband functions and displays the location on the PC screen. The notebook PC thus becomes a powerful GPS device that can support navigation and a wealth of location-based services.

The GPS front-end streams digitized IF data into the notebook through an industry-standard USB 2.0 interface. The software baseband program uses the input to calculate a position fix and, subsequently, to perform tracking. (One possible source of the software is Geotate. For more information, go to: <https://www.u-blox.com/en/product-resources>.)

To provide a generic interface, the software can create a virtual COM port so that it can link to a wealth of existing navigation and location applications. Most GPS software package interfaces conform to NMEA 0183 and will typically run on Microsoft's Windows® XP and Windows Vista® operating systems. Additionally, the software should be able to handle all available assistance data, whether supplied through industry-standard protocols or proprietary customer interfaces.

The CPU inside most of today's laptop computers has plenty of performance to meet the real-time decoding needs of software GPS receivers. On a 1GHz Pentium® M system, the average processor load when tracking will be about 6%; on a 2.18GHz Core™ Duo processor, the processor load will typically be below 5% when performing updates every second. With further algorithm development, it is possible to reduce the CPU usage to under 2%.

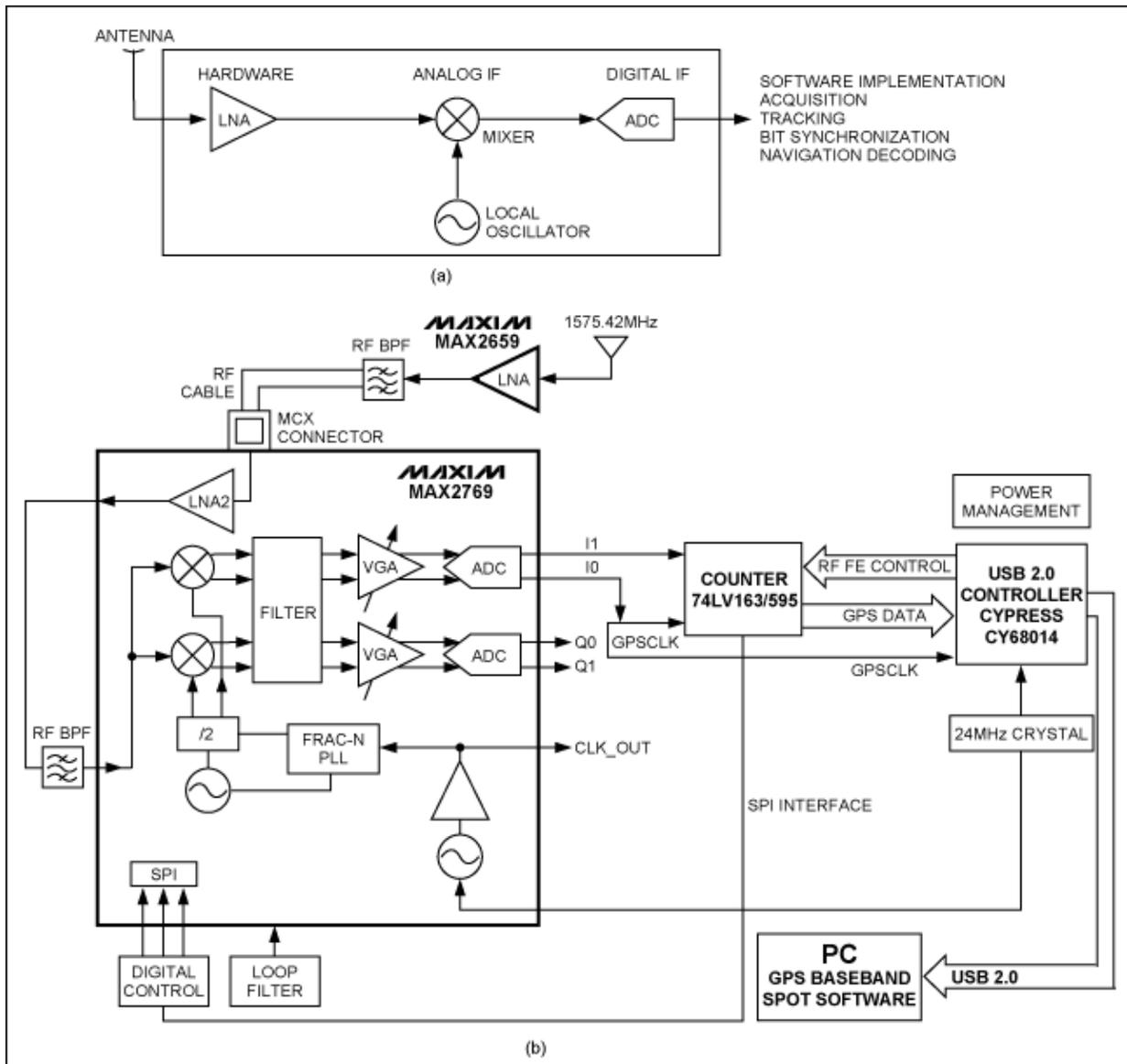


Figure 5. In a software GPS receiver the captured RF signal must be amplified, mixed down in frequency, and then digitized (a). In an actual receiver, a low-noise amplifier boosts the RF signal; then, the MAX2769 mixes it down and digitizes the signal. Finally, data travels through the counter and USB interface controller and is inserted into the USB protocol for transmission to the host PC (b).

## Circuit Operation and Performance

The RF front-end of a software-based GPS receiver first amplifies the weak incoming signal with a low-noise amplifier (LNA), and then downconverts the signal to a low intermediate frequency (IF) of approximately 4MHz (Figure 5a). This downconversion is accomplished by mixing the input RF signal with the local oscillator signal using one or two mixers. The resulting analog IF signal is converted to a digital IF signal by the analog-to-digital converter (ADC).

The MAX2769 integrates all these functions (LNA, mixer, and ADC), thus significantly reducing the development time for applications. The device offers a choice of two LNAs: one LNA features a very-low, 0.9dB noise figure, 19dB of gain, and -1dBm IP3, for use with passive antennas; the other LNA has a 1.5dB

noise figure with slightly lower gain and power consumption, and a slightly higher IP3, for use with an active antenna. Current consumption at 2.8V is minimal, ranging from 13mA to 18mA depending on the configuration.

There is a provision for external filtering at RF after the amplifier. The signal is then downconverted directly using the integrated 20-bit, sigma-delta, fractional-N frequency synthesizer together with a 15-bit integer divider to achieve virtually any desired IF between zero and 12MHz. A wide selection of possible IF filtering choices accommodates different schemes, such as those of Galileo.

The overall gain from RF input to IF output can be tuned or automatically controlled over a 60dB to 115dB range. The output can be chosen as analog, CMOS, or limited differential. The internal ADC has a selectable output of one to three bits. The integrated reference oscillator enables operation with either a crystal or a temperature-compensated crystal oscillator (TCXO), and any input reference frequency from 8MHz to 44MHz can be used.

A simple reference design for a USB dongle is available based on the MAX2769 with a Cypress Semiconductor USB controller, which operates from a 24MHz reference (Figure 5b). The dongle employs a single MAX8510 LDO to regulate the DC supply. A 3-wire (SPI™) digital bus is used to program the registers of the MAX2769. Alternatively, the system can be operated in one of eight hard-wired preconfigured states without SPI control.

Also on the chip are circuits to bias an active antenna and then turn off that antenna during shutdown mode for USB compliance. The MAX2769 can detect the presence or absence of a current draw on the antenna supply and automatically switch between LNA1 and LNA2. This flexibility is useful in applications that support the insertion of a more sensitive external active antenna in place of an integrated passive antenna. The designer simply wires the external antenna port to LNA2 and the internal port to LNA1. Upon insertion of the external antenna, the MAX2769 senses the current draw and automatically switches from LNA1 to LNA2.

The MAX2769 multiband GPS receiver provides a high-performance, compact solution for laptops, mobile handsets, PDAs, and automotive applications. A total nominal voltage gain of up to 115dB and a module noise figure of 1.4dB provide -143dBm acquisition and -154dBm tracking sensitivity using a commercially available GPS software package.

## Conclusion

Software GPS techniques provide a high level of flexibility and simplicity for many potential applications. To support these possibilities, the MAX2769 GPS receiver RF front-end provides flexibility in frequency planning for software GPS receivers, as well as traditional hardware implementations. Of course, every solution has its pros and cons—software GPS receivers require a high-performance processor and moderate amounts of memory. However, with software development and the proper selection of clocking and data update rates, memory usage can be minimized.

## References

1. Kaplan, E., *Understanding GPS: Principles and Applications*, 2nd ed. (Norwood, MA: Artech House Publishers, 1996).
2. Bao-Yen Tsui, J., *Fundamentals of Global Positioning System Receivers: A Software Approach*, 2nd ed. (New York: John Wiley & Sons, Inc., 2004).
3. For the decoding of navigation messages and position calculations, refer to Bao-Yen Tsui, *Fundamentals of Global Positioning System Receivers*.
4. Viterbi, A., *CDMA: Principles of Spread Spectrum Communications* (Reading, MA: Addison Wesley Longman Publishing Co., Inc., 1995).
5. Gold, R., "Co-optimal binary sequences for spread spectrum multiplexing," *IEEE Transactions on*

*Information Theory*, Vol. 13:4 (October 1967). pp. 619–621.

6. Ziemer, R.E., and Peterson, R.L., *Digital Communications and Spread Spectrum Systems* (New York: Macmillan Publishing Company, 1985).
7. Proakis, J. G., *Digital Communications*, 4th ed. (New York: McGraw-Hill, 2000).
8. Ibid.

A similar article appeared in the December 2006 issue of *Microwaves & RF*.

cdma2000 is a registered certification mark and registered service mark of the Telecommunications Industry Association.

MATLAB is a registered trademark of The MathWorks Inc.

PCI Express is a registered service mark of PCI-SIG Corporation.

PCIe is a registered service mark of PCI-SIG Corporation.

Pentium is a registered trademark of Intel Corporation.

Windows is a registered trademark and registered service mark of Microsoft Corporation.

Windows Vista is a registered trademark and registered service mark of Microsoft Corporation.

#### Related Parts

[MAX2769](#)

Universal GPS Receiver

[Free Samples](#)

---

#### More Information

For Technical Support: <http://www.maximintegrated.com/support>

For Samples: <http://www.maximintegrated.com/samples>

Other Questions and Comments: <http://www.maximintegrated.com/contact>

---

Application Note 4237: <http://www.maximintegrated.com/an4237>

APPLICATION NOTE 4237, AN4237, AN 4237, APP4237, Appnote4237, Appnote 4237

Copyright © by Maxim Integrated Products

Additional Legal Notices: <http://www.maximintegrated.com/legal>