US006606346B2

(12) **United States Patent**
Abraham et al.

(10) **Patent No.: US 6,606,346 B2**
(45) **Date of Patent: Aug. 12, 2003**

(54) **METHOD AND APPARATUS FOR COMPUTING SIGNAL CORRELATION**

(75) Inventors: **Charles Abraham**, San Jose, CA (US); **Donald L. Fuchs**, Wyckoff, NJ (US)

(73) Assignee: **Global Locate, Inc.**, San Jose, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **09/861,086**

(22) Filed: **May 18, 2001**

(65) **Prior Publication Data**

US 2002/0172266 A1 Nov. 21, 2002

(51) **Int. Cl.**[7] **H04L 27/30**

(52) **U.S. Cl.** **375/142**; 375/150; 375/343; 375/367; 370/515

(58) **Field of Search** 375/142, 150, 375/343, 364, 367; 370/335, 342, 441, 479, 515

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,527,278 A	*	7/1985	Deconche et al.	327/113
5,090,028 A	*	2/1992	Crebouw	375/343
5,426,665 A	*	6/1995	Cleverly et al.	375/150
5,579,338 A	*	11/1996	Kojima	329/302
5,663,734 A		9/1997	Krasner	342/357
5,727,018 A	*	3/1998	Wolf et al.	370/515
5,901,171 A		5/1999	Kohli et al.	375/200
5,931,893 A	*	8/1999	Dent et al.	708/422
5,966,402 A	*	10/1999	Yamamoto	375/150
6,005,899 A	*	12/1999	Khayrallah	375/343
6,289,041 B1	*	9/2001	Krasner	375/152
6,370,208 B1	*	4/2002	Kuo et al.	375/150

OTHER PUBLICATIONS

R. Wolfert, S. Chen, S. Kohli, D. Leimer, "Rapid Direct P(Y)–Code Acquisition In a Hostile Environment" Software Technology and Systems, Sensors Directorate, Air Force Research Lab. pp 353–360.

S. Lyusin, I. Khazanov, S. Likhovid, "Fast Acquisition by Matched Filter Technique for GPS/GLONASS Receivers", Magellan Corp. Moscow Dev. Center. pp 307–315.

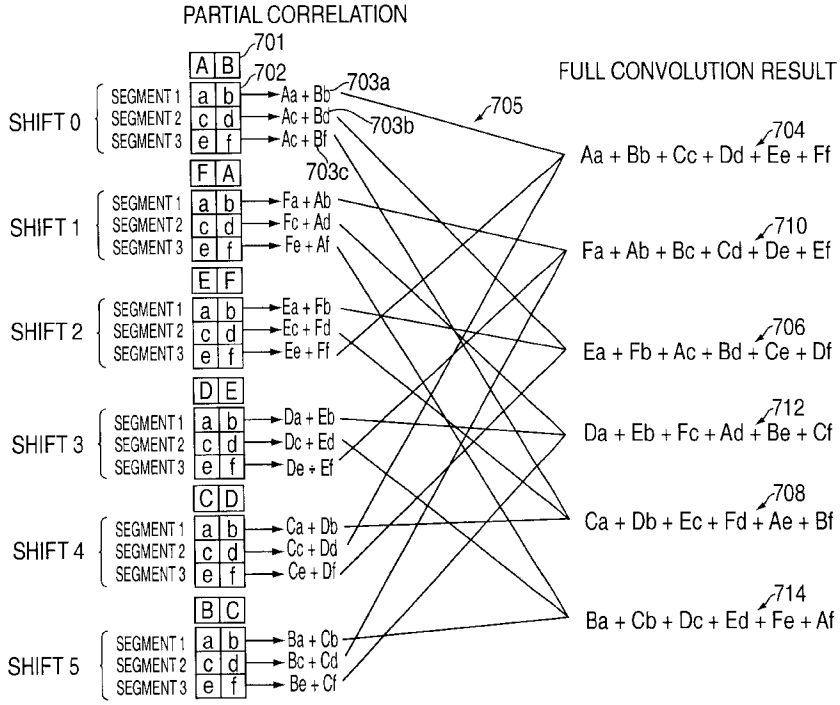
* cited by examiner

Primary Examiner—Stephen Chin
Assistant Examiner—Dac V. Ha
(74) *Attorney, Agent, or Firm*—Moser, Patterson & Sheridan, LLP

(57) **ABSTRACT**

A method and apparatus for computing a convolution between an input GPS signal and a C/A code reference by generating the convolution result in real time without storing unprocessed signal samples. The apparatus comprises a vector multiplier running at high speed to achieve the same result as a vector multiplier sized to process an entire epoch.

23 Claims, 9 Drawing Sheets



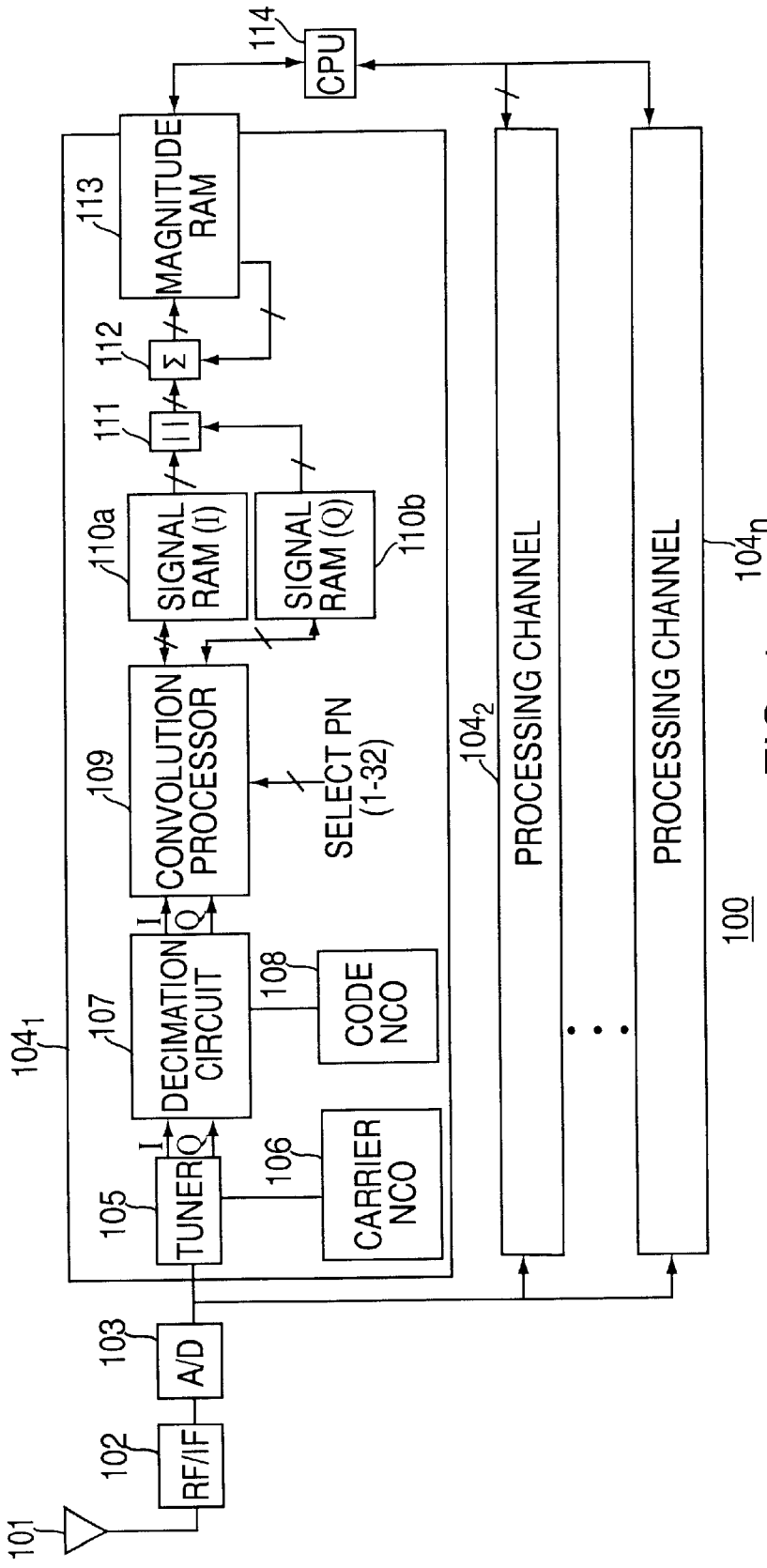

101
102
RF/IF
103
A/D
105
TUNER
I
Q
106
CARRIER NCO
107
DECIMATION CIRCUIT
108
CODE NCO
109
CONVOLUTION PROCESSOR
SELECT PN (1-32)
110a
SIGNAL RAM (I)
110b
SIGNAL RAM (Q)
111
112
Σ
113
MAGNITUDE RAM
114
CPU
104_1
104_2
PROCESSING CHANNEL
104_n
PROCESSING CHANNEL
100

FIG. 1

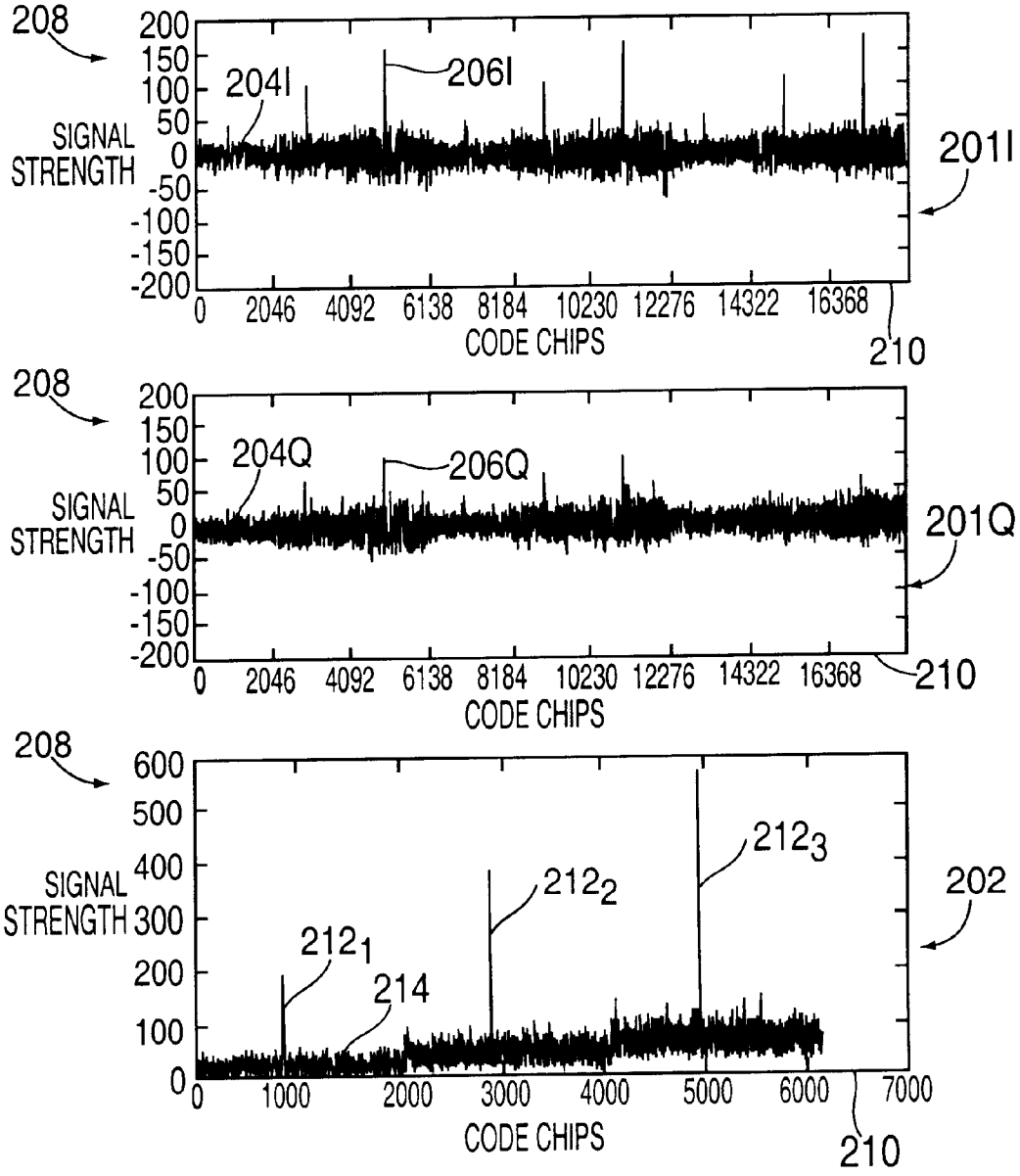

208
SIGNAL STRENGTH
200
150
100
50
0
-50
-100
-150
-200
2041
2061
2011
0 2046 4092 6138 8184 10230 12276 14322 16368
CODE CHIPS
210
208
SIGNAL STRENGTH
204Q
206Q
201Q
0 2046 4092 6138 8184 10230 12276 14322 16368
CODE CHIPS
210
208
SIGNAL STRENGTH
600
500
400
300
200
100
0
212_1
214
212_2
212_3
202
0 1000 2000 3000 4000 5000 6000 7000
CODE CHIPS
210

FIG. 2

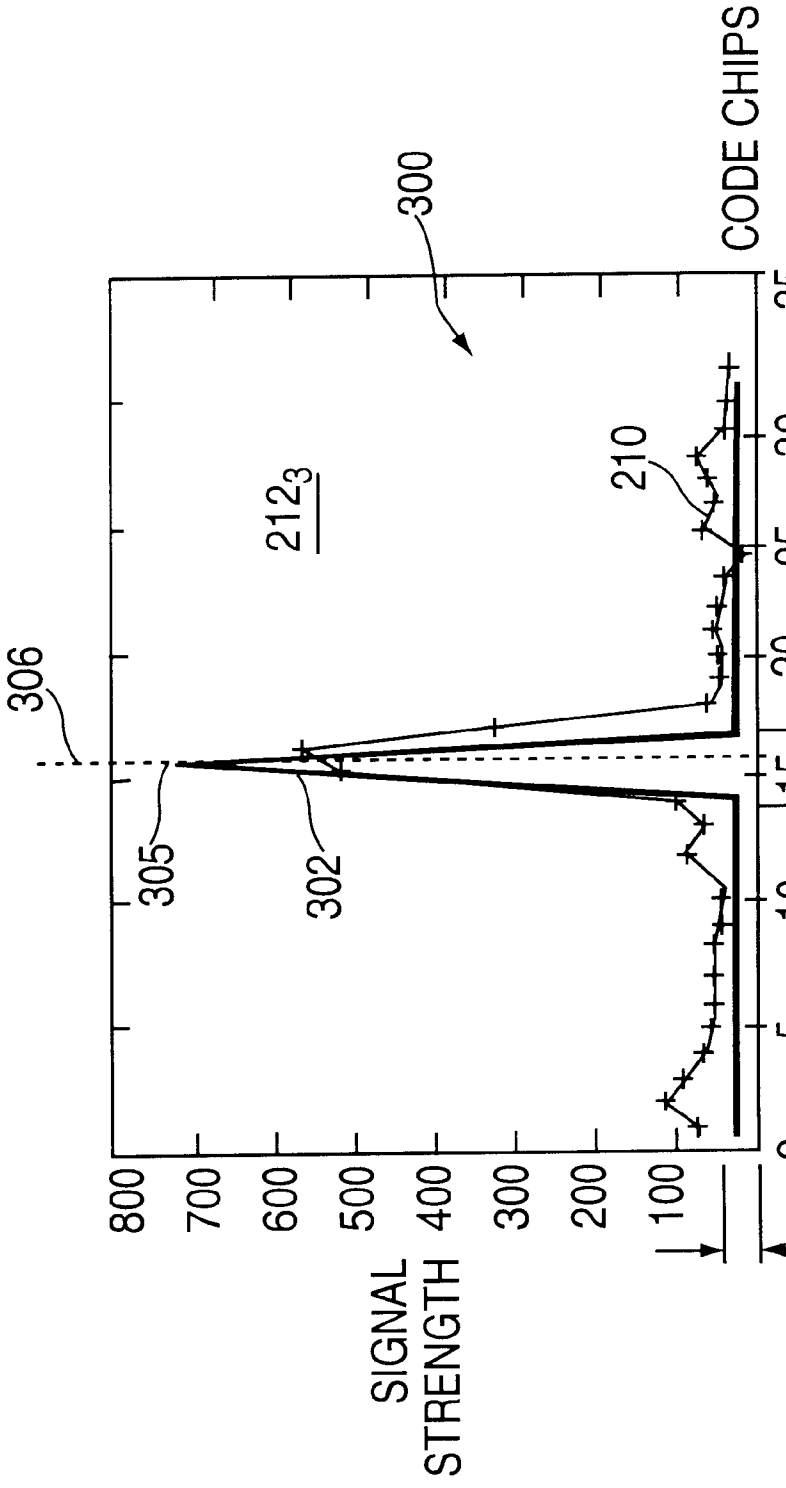

SIGNAL STRENGTH
800
700
600
500
400
300
200
100
CODE CHIPS
0
5
10
15
20
25
30
35
300
212_3
210
306
305
302
303
304

FIG. 3

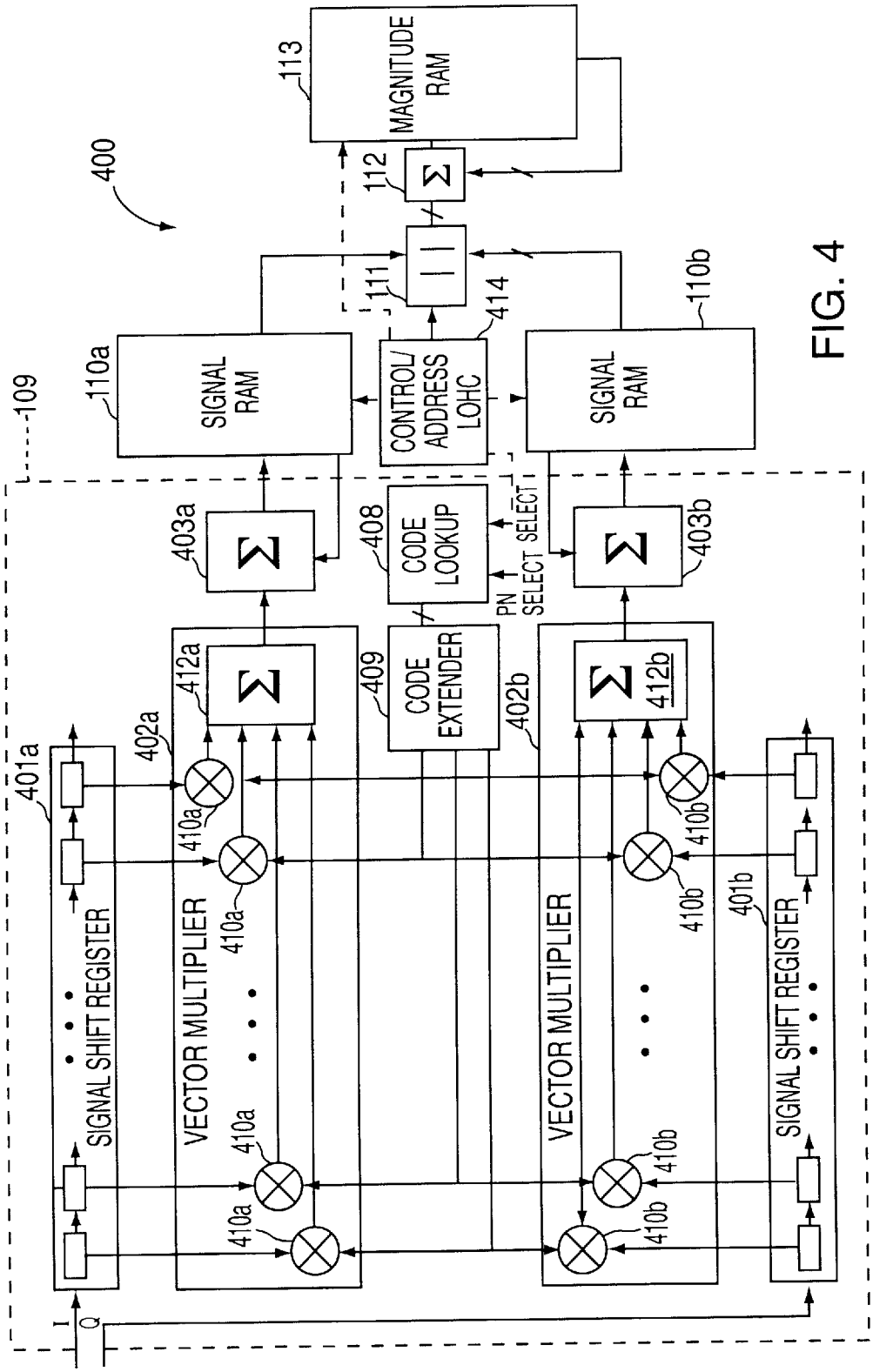

400
113
MAGNITUDE RAM
112
Σ
111
414
110a
SIGNAL RAM
CONTROL/ ADDRESS LOHC
110b
SIGNAL RAM
109
403a
Σ
408
CODE LOOKUP
PN SELECT SELECT
403b
Σ
412a
Σ
409
CODE EXTENDER
402b
Σ 412b
402a
401a
410a
410b
SIGNAL SHIFT REGISTER
VECTOR MULTIPLIER
VECTOR MULTIPLIER
401b
SIGNAL SHIFT REGISTER
I
Q

FIG. 4

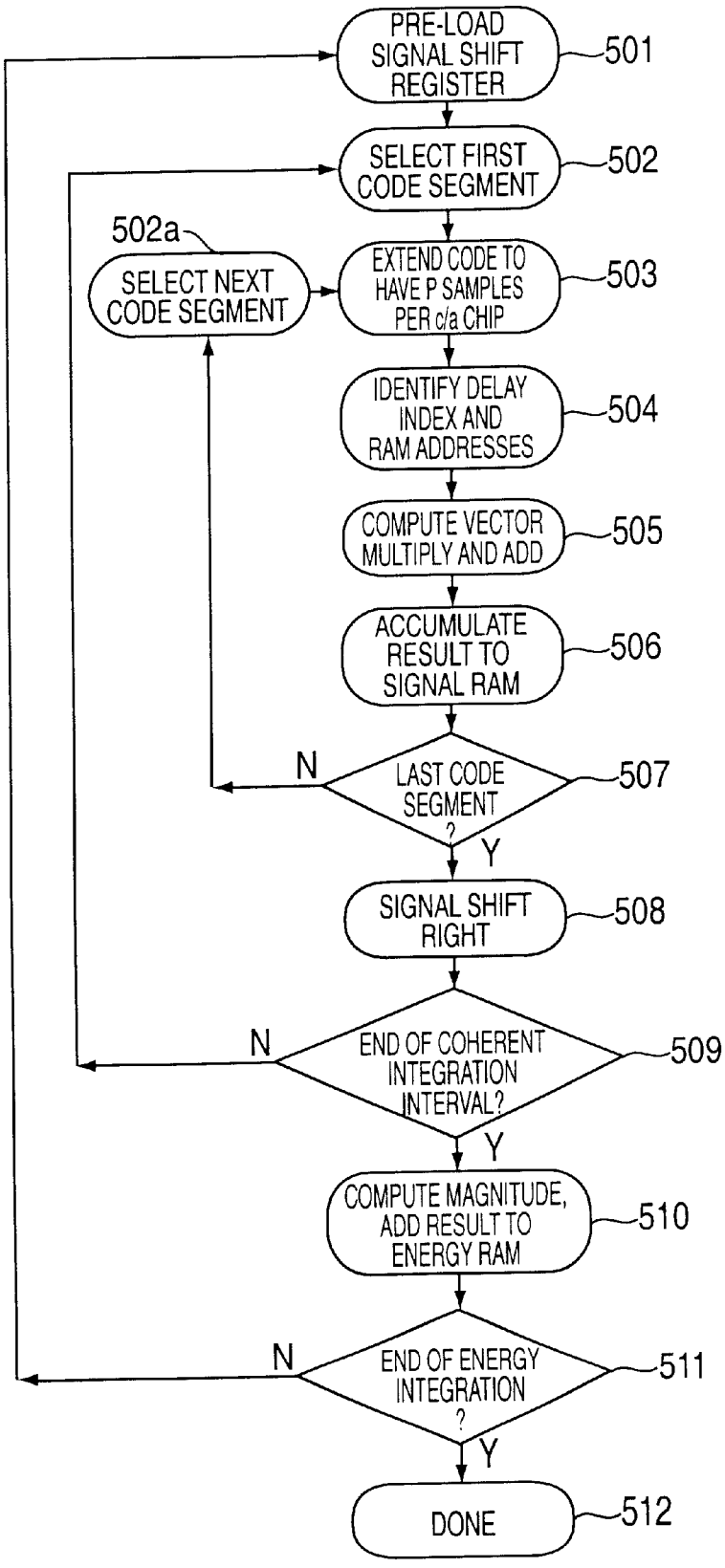

PRE-LOAD SIGNAL SHIFT REGISTER
501
SELECT FIRST CODE SEGMENT
502
502a
SELECT NEXT CODE SEGMENT
EXTEND CODE TO HAVE P SAMPLES PER c/a CHIP
503
IDENTIFY DELAY INDEX AND RAM ADDRESSES
504
COMPUTE VECTOR MULTIPLY AND ADD
505
ACCUMULATE RESULT TO SIGNAL RAM
506
N
LAST CODE SEGMENT ?
507
Y
SIGNAL SHIFT RIGHT
508
N
END OF COHERENT INTEGRATION INTERVAL?
509
Y
COMPUTE MAGNITUDE, ADD RESULT TO ENERGY RAM
510
N
END OF ENERGY INTEGRATION ?
511
Y
DONE
512

FIG. 5

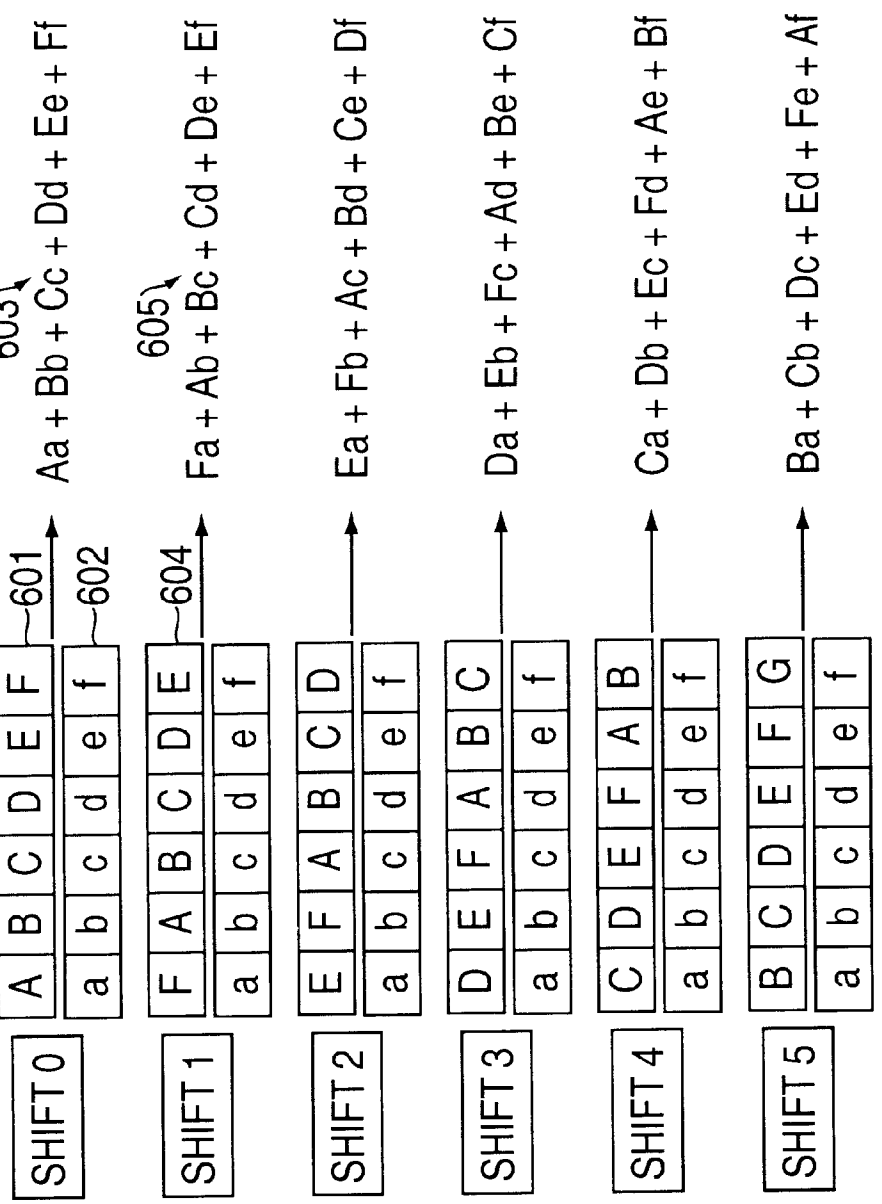


FIG. 6
(PRIOR ART)

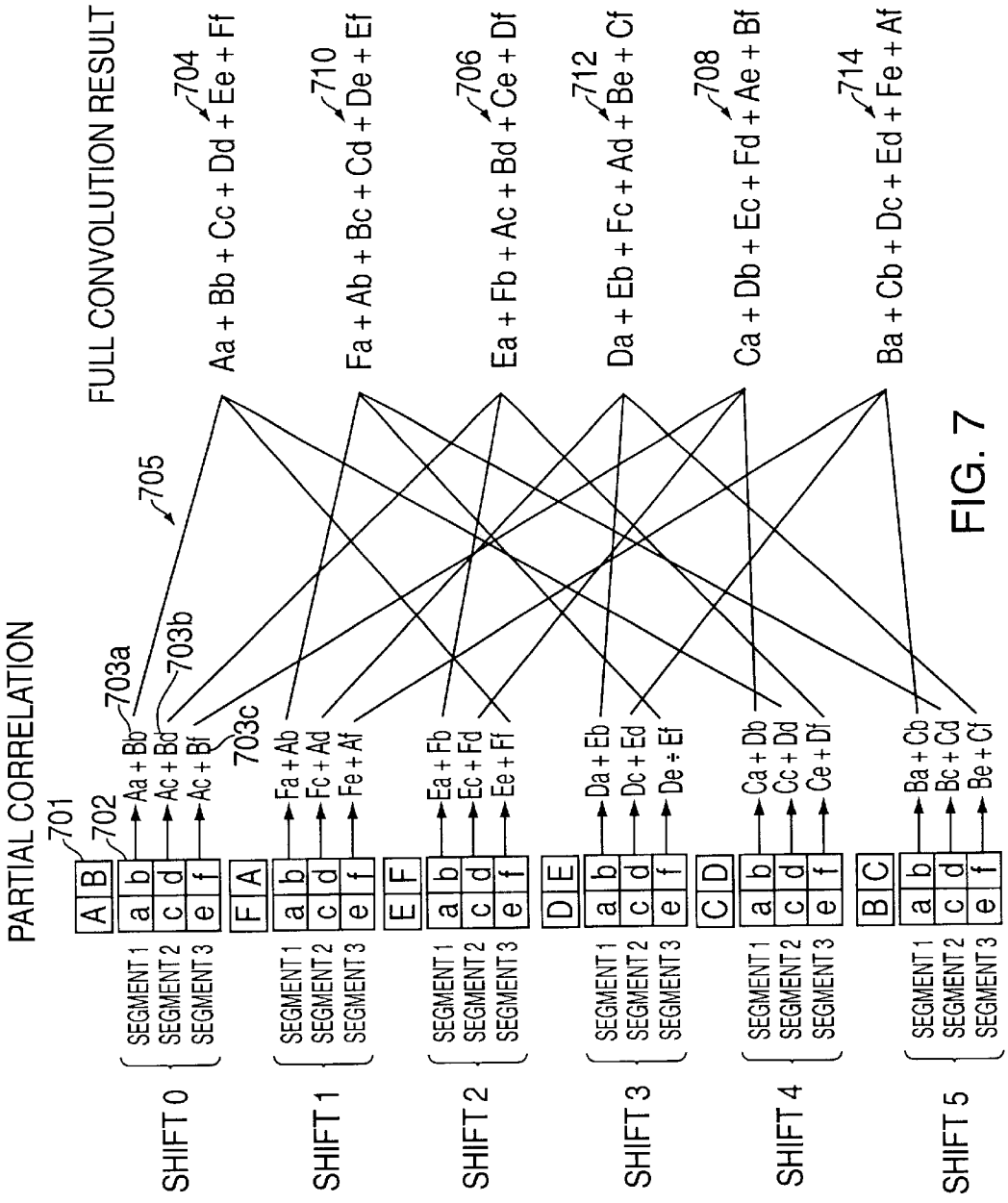

PARTIAL CORRELATION
FULL CONVOLUTION RESULT
701
702
703a
703b
703c
704
705
706
708
710
712
714
SHIFT 0
SHIFT 1
SHIFT 2
SHIFT 3
SHIFT 4
SHIFT 5
SEGMENT 1
SEGMENT 2
SEGMENT 3
Aa + Bb + Cc + Dd + Ee + Ff
Fa + Ab + Bc + Cd + De + Ef
Ea + Fb + Ac + Bd + Ce + Df
Da + Eb + Fc + Ad + Be + Cf
Ca + Db + Ec + Fd + Ae + Bf
Ba + Cb + Dc + Ed + Fe + Af

FIG. 7

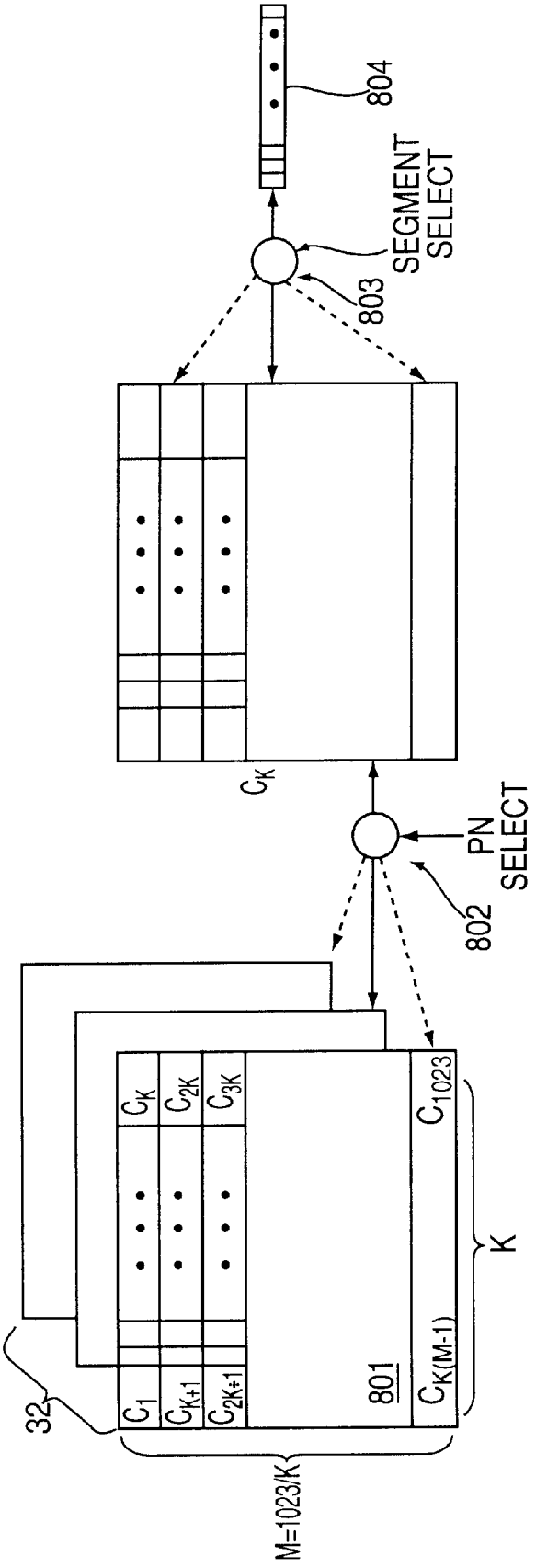

804
SEGMENT SELECT
803
C_K
PN SELECT
802
C_K
C_{2K}
C_{3K}
C_{1023}
C_1
C_{K+1}
C_{2K+1}
801
$C_{K(M-1)}$
K
32
M=1023/K

FIG. 8

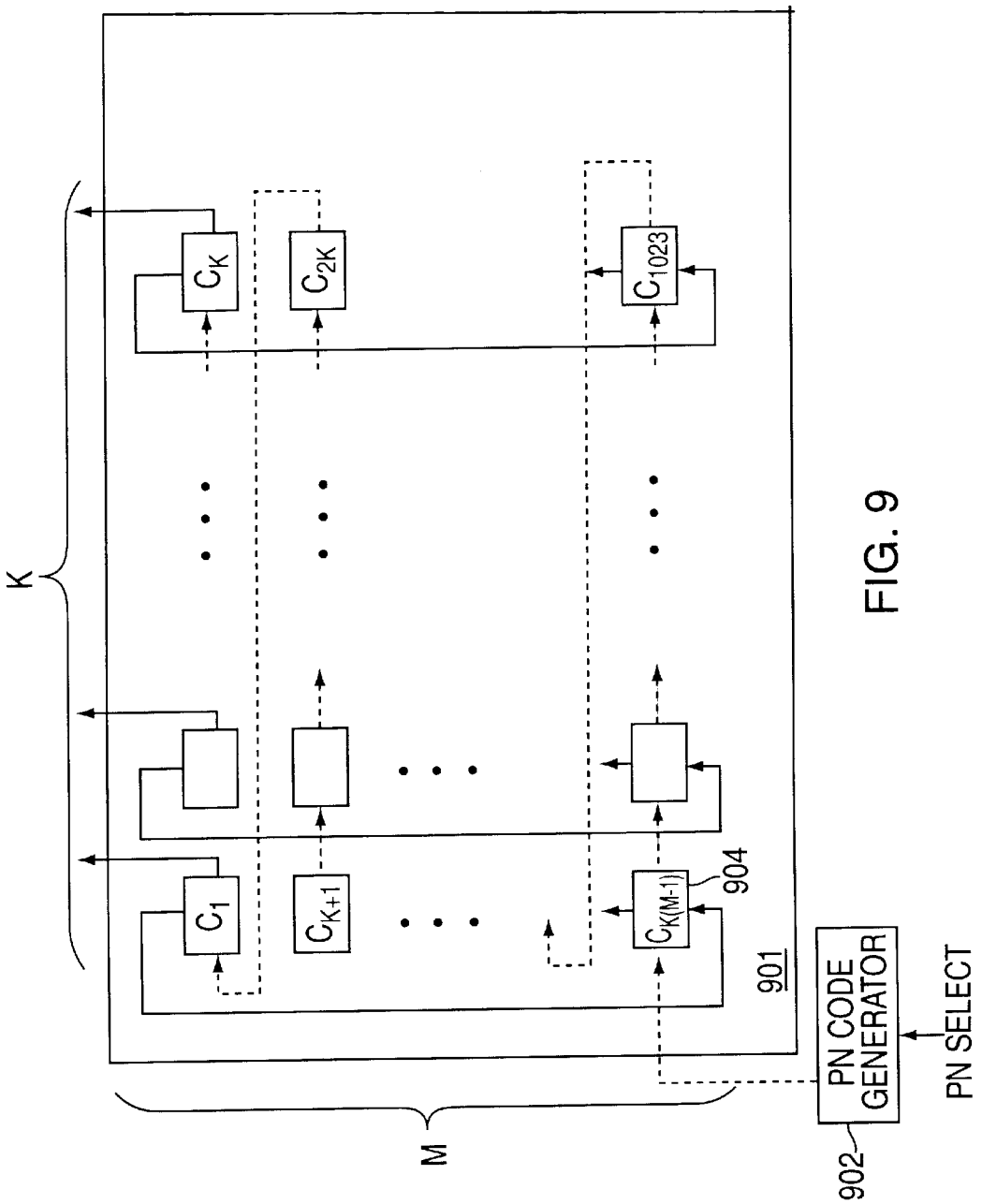

K
M
C_K
C_{2K}
C_{1023}
C_1
C_{K+1}
$C_{K(M-1)}$
904
901
PN CODE GENERATOR
902
PN SELECT

FIG. 9

METHOD AND APPARATUS FOR COMPUTING SIGNAL CORRELATION

BACKGROUND OF THE INVENTION

1. Field of the Invention

The present invention relates to signal correlators for digital signal receivers and, more particularly, the invention relates to a method and apparatus for performing signal correlation in, for example, a global positioning system (GPS) receiver.

2. Description of the Background Art

The process of measuring a global positioning system (GPS) signal begins with a procedure to search for the GPS signal in the presence of noise by attempting a series of correlations of the incoming signal against a known pseudo-random noise (PRN) code. The search process can be lengthy, as both the exact frequency of the signal and the time-of-arrival delay are unknown. To find the signal, receivers traditionally conduct a two dimensional search, checking each delay possibility at every possible frequency. To test for the presence of a signal at a particular frequency and delay, the receiver is tuned to the frequency, and the incoming signal is correlated with the known PRN code delayed by an amount corresponding to the time of arrival. If no signal is detected, the search continues to the next delay possibility, and after all delay possibilities are checked, continues to the next frequency possibility. Each individual correlation is performed over one or more milliseconds in order to allow sufficient signal averaging to distinguish the signal from the noise. Because many thousand frequency and delay possibilities are checked, the overall acquisition process can take tens of seconds.

Recently, new applications of GPS technology in wireless devices have emerged, for example, the use of GPS in cellular phones to provide emergency location capability. In these applications, rapid signal acquisition in just a few seconds is required. Furthermore, these applications require a GPS receiver to operate in harsh signal environments and indoors, where GPS signal levels are greatly attenuated. Detecting attenuated signals requires each correlation to be performed over a relatively long period of time. For example integration may be performed over a few seconds, as opposed to the 1–10 millisecond period used in traditional GPS receivers. The two dimensional sequential search process employed by traditional receivers becomes impractical at such long integration times, because the overall search time increases by a factor of 100 or more.

To accelerate the search process, GPS designers add additional correlators to the receiver so that multiple time of arrival possibilities can be checked simultaneously. Typically, each correlator that is added requires a separate code mixer and signal accumulator. For a given sensitivity level, this decreases search times in proportion to the number of correlators. To achieve the sensitivity and acquisition time demanded in cellular phone applications, the design might have to incorporate thousands of correlators. This addition is typically prohibitively complex and expensive for a consumer class device.

For example, U.S. Pat. No. 5,901,171, issued May 4, 1999, describes a triple multiplexing technique that allows a single time shared processing block to be used to perform up to 20 simultaneous correlations on each of 12 channels. This offers an improvement in performance relative to single correlator designs since blocks of 20 delay possibilities are checked simultaneously. A full signal search over a full range of delay uncertainties requires using the block of 20 correlators approximately 100 times in succession to check 2046 delays. Thus, if an acquisition must be performed in a few seconds, the integration time is limited to tens of milliseconds. This is insufficient to achieve the sensitivity needed for indoor GPS applications.

To further improve the search process, other GPS receiver architectures include processing capable of generating a convolution between the incoming signal and the known PRN code. This is equivalent to providing a complete set of correlators spanning all time delay possibilities over a full C/A code epoch (1023 chips), and U.S. Pat. No. 5,663,734, issued Sep. 2, 1997, describe fast Fourier transform (FFT) based software techniques to efficiently generate the necessary correlation results using software algorithms. This approach is not suitable for all applications, because a programmable digital signal processor (DSP) is needed to run the software FFT, and a large memory is needed to store unprocessed signal samples. Furthermore, this approach can have a large processing delay due to the software computations and the fact that software processing starts only after a complete snapshot of the signal is stored. In many applications, a real time processing solution is preferred, preferably one that does not involve extensive software processing. Lyusin et al., "Fast Acquisition by Matched Filter Technique for GPS/GLONASS Receivers", pp 307–315 describes hardware approaches to performing the convolution in real time using a matched filter with 1023 taps. The matched filter consists of shift registers large enough to hold a full C/A code epoch, as well as a width 1023 vector multiplier and adder unit that generates the inner product between a full epoch of the signal and the C/A code.

This circuit is complex relative to the constraints of low cost consumer devices such as cellular phones. Other matched filter approaches, such as utilized in military class receivers for P-code acquisition, also incorporate large vector multipliers.

Thus, there is a need for an improved, simple and low cost GPS processing block capable of processing an entire epoch of signal and C/A code. Such a device must be built from hardware of relative simplicity, yet be capable of generating a full convolution, or many parallel correlations, preferably without a large vector multiplier.

SUMMARY OF THE INVENTION

The invention provides a method and apparatus for computing a full convolution between an input signal (e.g., a GPS signal) and a pseudorandom noise (PRN) code reference by generating the convolution result in real time without storing unprocessed signal samples, and without extensive software processing. The apparatus comprises a vector multiplier running at high speed to achieve the same result as a vector multiplier sized to process an entire epoch. The invention can be implemented in an integrated circuit that fits the complexity constraints of a consumer class device such as a cellular phone. The design includes the necessary logic to enable long term averaging of convolution results to ensure high sensitivity. This invention is capable of correlating signals for use in deriving a position location from highly attenuated signals, including signals received indoors.

The complete apparatus consists of a conventional GPS tuner, a decimation circuit, a convolution processor, and RAM blocks that accumulate convolution results. The convolution processor runs at a high clock rate on the order of

100 MHz and higher enabling the computation of a full convolution by repeated use of a small block of circuitry. Specifically, each point of the convolution is decomposed into a series of partial correlations, each of which is generated using a vector multiplier that is sized to process only a portion of an epoch. The apparatus organizes the partial correlations by subdividing the C/A code into a non-overlapping set of code segments. Each partial correlation uses only one code segment at a time, allowing the C/A code to be stored and retrieved efficiently, using a simple lookup table.

The processor begins by decimating input IF samples to create a signal stream at a desired sample rate, where the rate is precisely matched to the timing of the incoming signal. If the desired sample rate is Pf_o (P samples per C/A chip) then the sampling rate is set so that exactly 1023×P samples are taken in each signal epoch. The processor correlates the signal clocking signals through shift registers sized to hold P×K input samples, where K is a factor of 1023. At each signal shift, a series of M partial correlation operations are performed with M chosen such that M×K=1023. Each partial correlation consists of taking the inner product of the contents of the signal shift registers with a block of reference samples created by extending a length K segment of the C/A code to P×K samples. Partial correlation results are accumulated in memory. By accumulating partial correlation results, the processor generates complete correlation results for many correlation points, up to the full convolution.

DESCRIPTION OF DRAWINGS

The invention can be readily understood by considering the following detailed description in conjunction with the accompanying drawings, where:

FIG. **1** shows a block diagram of a GPS receiver comprising the present invention;

FIG. **2** shows an example of waveforms produced by the invention;

FIG. **3** shows details of an accumulated magnitude waveform of FIG. **2**;

FIG. **4** shows a detailed block diagram of the convolution processor and the convolution results processing circuits;

FIG. **5** depicts a flow diagram of a method of operation of the invention;

FIG. **6** graphically illustrates a simplified example of computing a full convolution in the traditional manner;

FIG. **7** graphically illustrates how the full convolution of FIG. **6** is performed using the invention;

FIG. **8** illustrates an embodiment of a code lookup apparatus suitable for use in the invention;

FIG. **9** illustrates an embodiment of a two-dimensional code shift register suitable for use in an alternate embodiment of the invention.

DETAILED DESCRIPTION

FIG. **1** depicts a block diagram of a global positioning system (GPS) receiver **100** incorporating the present invention. The use of a GPS receiver as the platform within which the invention is incorporated forms one application of the invention. Other platforms that require signal correlation may find use for the present invention.

Signals (such as GPS signals) are received by an antenna **101**. A radio-frequency-to-intermediate-frequency converter (RF/IF converter) **102** filters, amplifies, and frequency shifts the signal for digitization by an analog-to-digital converter (A/D) **103**. The elements **101**, **102** and **103** are substantially similar to those elements used in a conventional GPS receiver.

The output of the A/D **103** is coupled to a set of processing channels $\mathbf{104}_1$, $\mathbf{104}_2$, . . . $\mathbf{104}_n$ (where n is an integer) implemented in digital logic. Each processing channel $\mathbf{104}_n$ may be used to process the signal from a particular GPS satellite. The signal in a particular channel is tuned digitally by a tuner **105**, driven by a numerically controlled oscillator (NCO) **106**. The tuner **105** serves two purposes. First, the IF frequency component remaining after RF/IF conversion is removed. Second, the satellite Doppler frequency shift resulting from satellite motion, user motion, and reference frequency errors is removed. The output from the tuner is a baseband signal consisting of an in-phase component (I) and a quadrature component (Q). The steps of **105** and **106** are substantially similar to those used in conventional GPS receiver designs.

A decimation circuit **107** processes the output of the **105**. The output of the decimation circuit **107** is a series of complex signal samples with I and Q components, output at a rate precisely timed to match the timing of the input signal. In one embodiment of the invention, the decimation operation is a simple pre-summer that sums all the incoming signal samples over the period of an output sample. A numerically controlled oscillator (NCO) **108** is used to time the sampling process. For example, if P=2, the code NCO **108** is set to generate a frequency of $(2\times f_s)$, where f_s is f_o (the GPS signal's C/A code chipping rate), adjusted for Doppler shift. The NCO adjusts for Doppler shift based on external input from firmware commands. Because the Doppler shift is different for each satellite, a separate code NCO **108** and decimation circuit **107** is required for each channel $\mathbf{104}_n$. It should be noted that there is no requirement that the incoming sample rate be an integer multiple of the f_s, as the code NCO **108** is capable of generating an arbitrary frequency. If the decimation circuit **107** is a pre-summer, the number of samples summed will typically toggle between two values, so that over the long term, the correct sample timing is maintained. For example, if the incoming sample rate is 10 MHz, and the desired sample rate is 2.046 MHz, the pre-summer will add either 4 or 5 samples, so that the desired sample rate is maintained on average.

The decimation circuit **107** may also include a quantizer (not shown) at its output to reduce the number of bits in the signal components before further processing. In one embodiment of the invention, 2-bit quantization is used.

The signal samples from decimation circuit **107** are coupled to a convolution processor **109**. The convolution processor **109** generates results that are stored in signal random access memories (RAMs) **110***a* and **110***b*. Specifically, these RAMs **110***a* and **110***b* hold a complex vector that makes up all or part of the full convolution between the input signal and a reference PN code (e.g. a GPS C/A code). The convolution result will have a peak at points corresponding to high correlation between the signal and reference (the PN code). As shall be discussed in detail below, the relative location of these peaks for various satellite signals is used to ultimately compute position information.

The convolution processor **109** and signal RAMs **110***a* and **110***b* accumulate convolution results for multiple epochs of the GPS signal, which repeats at nominal 1 millisecond intervals. For example, if 10 milliseconds of the signal are processed, the values in RAM **110***a* and **110***b* are the sum of 10 correlation results each generated over one epoch. All the

individual correlations should have a similar characteristic, since the timing of the decimation operation ensures that samples are taken at the same relative moment within each epoch. Accumulating similar results from individual correlations improves the signal to noise ratio, enhancing the ability of the receiver to detect weak signals. This processing may be referred to as coherent integration and, as will be discussed, can be combined with magnitude integration to yield correlation results averaged over a time period of up to several seconds.

The length of time over which coherent integration interval is performed is limited by several factors, including uncompensated Doppler shift, GPS signal navigation data bits, and phase shifts induced by motion of the receiver **100**. These factors introduce slow, but seemingly random phase variations into the signals. Over many tens of milliseconds, these phase changes cause destructive interference that defeats the purpose of coherent integration. Therefore, to achieve long averaging intervals, the receiver **100** performs a secondary step of magnitude accumulation. Specifically, the signals stored in the signal RAMs **110***a* and **110***b* are periodically output to a complex normalizer **111** that generates a complex magnitude value of the complex convolution vector. The complex magnitude values are accumulated by an adder **112** and stored in magnitude RAM **113**. Each time the complex magnitude of the signal is computed, the signal RAMs **110***a* and **110***b* are cleared to allow another coherent integration to occur. The process continues until the desired number of magnitude accumulations is completed. For example, if the coherent averaging interval is 10 milliseconds, and 200 magnitude accumulations are desired, the total process will run over 2 seconds.

After convolution processing, the magnitude RAM **113** contains a vector containing the complex magnitude of the convolution result, integrated to improve signal-to-noise ratio. As shall be discussed below, this vector is further processed by software algorithms that are executed by the CPU **114** to produce pseudorange data that is used to yield the position of the receiver. It should be noted that the CPU computational load for these steps is quite modest compared to a conventional GPS receiver or an FFT based correlator. In this implementation, the computationally intensive tasks of correlation and integration are completed prior to software processing.

FIG. **2** depicts waveforms **201**I, **201**Q and **202** generated by the components of FIG. **1**. The waveforms **201**I, **201**Q and **202** are plots of signal strength (axis **208**) versus code chips (axis **210**). The waveforms depict the output of the convolution processor **109** during coherent integration and magnitude integration. For clarity, only 9 milliseconds of signal processing time is shown consisting of 3 magnitude accumulations each based on 3 coherent integrations. In the example, P=2, so there are 2046 signal samples per coherent integration. Waveforms **201**I and **201**Q are the output from the convolution processor **109** where **201**I is the I-component of the output and **201**Q is the Q-component. Each block of 2046 samples is a full convolution result, generated in real time by the convolution processor **109** from the 2046 signal samples processed during the interval. The convolution result contains noise except in the vicinity of a single peak (such as indicated by reference numbers **206**I and **206**Q) corresponding to the time delay of the signal. The signal repeats every epoch, so the peak reappears each 2046 samples. Over the first three cycles, correlation results are accumulated in the RAM **110***a* and **110***b* by summing values at corresponding delays from each epoch. (For example, the values at output time 4 are summed with the values at output time 2050 and 4096.) The correlation peak always appears at the same delay offset and the size of the peak increases over the accumulation, roughly tripling over 3 epochs. The level of the noise also increases, but rises only as the square root of 3 because the noise correlation is uncorrelated from epoch to epoch. The signal to noise ratio improves through the accumulation process, increasing by roughly the square root of 3. Waveform **201**Q illustrates the same signal accumulation process occurring in the quadrature channel.

Beginning with the 4^{th} cycle of the signal, the signal RAMs **110***a* and **110***b* are cleared to zero, and the signal accumulation process begins again. Waveforms **201**I and **201**Q show the correlations accumulating and dumping 3 times over 9 signal epochs.

At the end of the coherent averaging interval the accumulated signal's magnitude is computed and summed into the magnitude RAM **113**. The signal in the magnitude RAM **113** is shown as waveform **202**. In the example, the waveform **202** updates three times corresponding to the completion of each coherent integration. The peaks are identified by reference numbers $\mathbf{212}_1$, $\mathbf{212}_2$, $\mathbf{212}_3$ and noise is identified by reference number **214**. As can be seen, the signal-to-noise ratio increases with each magnitude accumulation, further enhancing the ability of the system to identify the peak corresponding to the time of arrival.

It should be noted that in the example, the complex phase of the signal varied over the 9 epochs. In particular, the signal was initially present in both I and Q channels, but by the final epoch, had rotated so that the signal was strong in the I channel and nearly absent in the Q channel. As mentioned above, imperfect Doppler shift tuning and other effects cause this rotation. Over many epochs, the phase would rotate through many cycles, resulting in cancellation of the signal when accumulated. For this reason, the inventive receiver accumulates coherently over only a short interval, relying on magnitude (non-coherent) accumulation for long term averaging. Magnitude values are independent of phase, and may be successfully integrated over several seconds.

FIG. **3** illustrates the accumulated magnitude waveform **202** in greater detail. The plot **300** shows the magnitude of the convolution in the vicinity of a peak $\mathbf{212}_3$ corresponding to the time delay of the signal. Points on the code chip axis **210** are spaced at an interval equal to the CIA code chip length divided by P, where P is the ratio of the signal sampling rate to f_o, the C/A code chipping rate. In the example, P=2, so the points are spaced at ½ chip intervals, or approximately 500 ns. (This spacing in time corresponds to a range difference of 150 meters). In order to achieve pseudorange measurements on the order of ten meters or better, the convolution results are further processed, typically in the Cpu **114**, to produce the position information. There are numerous interpolation techniques that can be used to estimate the true time delay, using the discrete correlation values provided by the convolution process. One embodiment uses a least squares estimation technique to identify parameters of a signal that best fits the noisy measured data. The ideal response of a signal is the magnitude of the autocorrelation of the signal. This waveform can easily be shown to have the form of a raised triangle **302**. The width **303** of the triangle base is exactly 2 C/A code chips, or 4 points on the convolution result (for the P=2 case). The height **304** of the base of the triangle is the magnitude of the noise in the convolution for time delays not corresponding to the signal. The magnitude of this noise can be estimated from the data or pre-calculated based on design

parameters, such as the amplifier noise figure, cable and filter loss and system temperature. The peak **305** of the triangle and the center **306** of the triangle are unknowns corresponding to the signal magnitude and time delay. The least squares method can be used to estimate these two parameters so as to fit the noisy data points to a triangle with a given peak and center. FIG. **4** depicts a detailed block diagram of the convolution processor **109** (as well as the convolution results processing circuits **400**), in particular details showing how a full convolution is generated by repeated use of a small block of circuitry. Operation of the circuits can be best understood with simultaneous reference to FIG. **4**, a flow diagram of FIG. **5** representing the operation of the processor **109** of FIG. **4**, and by comparison of the simple examples of FIG. **6** and FIG. **7**.

Signals from the decimation circuit **107** are coupled to shift registers **401***a* and **401***b*, handling I and Q components, respectively. Each shift register **401***a* and **401***b* is of length P×K, where P is the desired number of samples per C/A code chip, and K is chosen as a design parameter. As will be explained K is a factor of **1023**. To simplify the discussion, the remainder of the discussion focuses on one particular embodiment with P=2 (samples spaced ½ chip apart) and K=33. This means of advancing the signal through the shift register eliminates the need for circuitry to double-buffer the signal, reducing the cost and complexity of implementation.

Signals advance through shift registers **401***a* and **401***b* at the rate of $2f_o$, as timed by the code NCO **108**. The signals remain in place in the shift registers for many clock cycles, so that a series of partial correlation operations can be performed. Specifically, a total of M partial correlations are performed, where M=1023/K or 31 in this example. Each partial correlation consists of a fast vector multiply and add operation between the contents of each signal shift register and a segment of the code containing P×K (e.g., 66) code samples. The fast vector multiplication and addition occurs in circuits **402***a* and **402***b*. Circuits **402***a* and **402***b* respectively comprise multipliers **410***a* and **410***b* and summers **412***a* and **412***b*. The operation consists of multiplying each of the 66 signal samples in the signal register **401***a* or **401***b* by 66 code samples (formed by extending 33 code samples with the code extender **409**), and summing the results in summer **412***a* and **412***b*. The operation occurs separately and simultaneously in the I and Q channels. Mathematically, this operation is referred to as an inner product, defined as

$$\sum_{i=1}^{P\times K} <signal_i> <code\ c_i>$$

The output of the vector multiply and add may be re-quantized to keep the numbers in a small range so as to avoid overflowing RAMs **404***a* and **404***b*. For simplicity, the quantizer is not shown. In one embodiment, the re-quantization is to 2 bits of resolution.

The results of the vector multiply and add are accumulated by adders **403***a* and **403***b* and processed by the convolution results processing circuits **400**. Circuits **400** comprise signal RAM **110***a*, **110***b*, complex normalizer **111**, adder **112** and magnitude RAM **113**. stored in signal RAMs **111***a* and **404***b*. The accumulation process consists of reading from RAM **110***a* and **110***b* the current values for a particular time delay, adding the just computed partial correlations, and writing the sums back to RAMs **110***a* and **110***b*. By properly combining partial correlations that correspond to a particular time delay, the full correlation for that delay is computed. As described previously, the process continues for as many epochs of the signal as desired to enhance signal to noise ratio. Thus, the adders **403***a* and **403***b* serve two purposes: the combining of partial correlations within an epoch; and the accumulation of correlations across several epochs.

The outputs from signal RAMs **110***a* and **110***b* are combined in complex normalizer **405** to form the magnitude of the signal. The I and Q waveforms in these RAMs **110***a* and **110***b* can be viewed as the real and imaginary part of a complex waveform. Forming the magnitude consists of squaring each component, summing the results, and taking the square root of the result. There are several approximations to the magnitude that can be used to simplify circuitry. In one embodiment, the complex magnitude is approximated by taking the scalar magnitude of I and Q independently and determining which is larger. The magnitude can be approximated by taking the larger magnitude and adding it to the one half of the smaller magnitude.

The results of the magnitude operation may be scaled to keep the values in a small range so as to avoid overflowing RAM **113**. For simplicity, a scaler is not shown. In one embodiment, the scaling consists of shifting the result by 3 bits (i.e., divide by 8).

It would also be possible to accumulate signal powers rather than signal magnitudes. In this case, the operation in **405** would be power estimation, typically computed by taking the sum of the squares of I and Q. In this case, the pseudorange determination algorithms described in reference to FIG. **3** would have to be slightly modified to perform a fit against a power waveform as opposed to a magnitude waveform. Alternatively, additional nonlinear operations could be used to generate values representative of the magnitude or power of I and Q.

The output from complex normalizer **111** is accumulated by the adder **112** into magnitude RAM **113**. The accumulation process consists of reading from RAM **113** the current magnitude value for a particular time delay, adding in the just computed magnitude result, and writing the sum back to the RAM **113**. As discussed previously, the magnitude accumulation continues for as many cycles as required to achieve signal to noise ratio enhancement.

The vector multipliers **402***a* and **402***b* perform M partial correlations for each shift of the signal. A code lookup circuit **408** generates the reference code samples for each partial correlation. The lookup is controlled by two lookup indexes. First, the code must be selected from 1 of 32 codes. This selection is constant through the convolution process and is established when the processing channel is configured to correlate for a particular satellite signal. The second index is a segment index between 1 and M. Each C/A code consists of 1023 chips, which are divided into M non-overlapping segments each consisting of K adjacent code chips. The lookup index identifies which code segment is needed. The output from the code lookup circuit is K chips comprising the segment. The selection process is controlled by Control/Address Logic **414**.

The code extender **409** takes as its input K chips of a segment and extends the segment into K×P code samples. The extension operation consists of converting each code chip into P identical code samples. The output from the code extender **409** forms the reference code input to vector multipliers **402***a–b*. In the example, the output from the code extender is 66 samples made up of 33 unique values, each replicated twice.

The architecture shown in FIG. **4** requires a clock that is substantially faster than the C/A code rate f_o. For example, if two samples per C/A code chip are used (P=2) and K and

M are to be 33 and 31 respectively, achieving the full convolution requires performing 31 partial correlations for each shift of the signal shift register, which advances at rate $2\times f_o$. Typically, at least two clock cycles are required to read and write RAMs **110***a* and **110***b*. Assuming two clock cycles, the minimum clocking rate required to achieve the full convolution is:

$$f_{clk}=2\times31\times2\times f_o=2\times31\times2\times1.023 \text{ MHz}\approx127 \text{ MHz}$$

This rate is easily achievable in modern integrated circuit logic.

It should be noted that the invention could also be used to compute a subset of the full convolution. In this case, fewer than M partial correlations are performed for each shift of the signal shift register. In this case, the total range of delays will be less than the P×1023 making up a full convolution. In particular if M_2 partial correlations are performed, then M_2 times K times P delay values are generated. The clocking rate to the processor is reduced by the ratio of M_2 to M. Furthermore, the size of the RAMs is reduced by this ratio as well. Thus, this alternative may be useful in systems that do not have the computation or memory resources to process the full convolution.

Other choices for K and M result allows further design tradeoffs to be made, however, since the prime factors of 1023 are 3, 11, and 31, the choices for K and M are limited. Reducing K is desirable since this reduces the size of the shift registers **401***a* and **401***b* and the complexity of the vector multipliers **402***a* and **402***b*, but requires a larger M and therefore a large clocking rate. The choices for K are 3, 11, 31, 33, 93. These choices would require clocking rates of 1.39 GHz, 380 MHz, 135 MHz, 127 MHz, and 45 MHz respectively (always assuming P=2 and 2 clock cycles per partial correlation.) Based on the technology available at the time of the demonstration, the K=33 choice was made for one embodiment. With future technologies, the choice of K=11 and a clock rate of 380 MHz may become viable and would result in a further reduction of the logic complexity. Thus, the architecture has the desirable attribute of supporting optimized tradeoffs between speed and logic complexity.

The sequencing of code segments is controlled by control logic **414**. This control logic also identifies the correct addresses for the RAMs **110***a*, **110***b* and **113**. As will be discussed below, the partial correlations are generated in a non-sequential order, thus the generation of RAM addresses is non-trivial.

The operation of the circuits of FIG. **4** can also be understood by reference to the flow diagram of FIG. **5**. Operation begins at step **501** with pre-loading of the signal shift registers **401***a* and **401***b*. At this point, convolution processing can begin. At step **502**, a code segment is accessed for the particular partial correlation. At step **503**, the code segment is extended by the code extender to have P samples per C/A chip. Next, at step **504**, the delay index and corresponding RAM addresses are computed. The delay index indicates which point of the full convolution will be updated by the partial correlation. As will be apparent from the example discussed in conjunction with FIG. **7**, the delay index jumps around in a non-linear, but deterministic manner. The address computation is a function of the number of signal shifts and the code segment.

At step **505**, the partial correlation is computed using the vector multipliers **402***a* and **402***b*. At step **506**, the result is accumulated into the signal RAMs at the location indicated by the delay index. Next at step **507**, a check is made to determine whether the processing has reached the end of the coherent integration interval. If not, the method returns back to step **502***a*, and repeats for the above steps for the next code segment.

If, at step **507**, the check indicates that partial correlations are complete for all code segments (e.g., 31 partial correlations), the method proceeds to step **508**. At step **508**, the signal registers **401***a* and **401***b* are shifted by one sample.

The process then moves to step **509**, where a check is performed to see if the last shift encountered the end of the coherent integration interval. If not, the process cycles back to the start at step **502**. If the check indicates the end of the coherent integration interval, then the method continues to step **510**, where the signal magnitude is computed by complex normalizer **111**. The result is added using adder **112** and stored in the magnitude RAM **113**. Next, at step **511**, a check is made to determine if all magnitude accumulations have been performed. If so, the method completes at step **512**. If not, processing continues by performing the next partial correlation at step **501**.

FIG. **6** and FIG. **7** illustrate, through a simplified example, how the invention utilizes partial correlations to accumulate a full convolution result. For clarity, these diagrams illustrate convolution of a very short length 6 code, as opposed to the length 1023 C/A code of a GPS signal. To further simplify the example, one sample per code chip is used, i.e. P=1. FIG. **6** illustrates convolution through a standard matched filtering approach, and FIG. **7** illustrates the identical convolution through the method of combining of partial correlations. The details of FIG. **7** are helpful in understanding overall operation of the invention. Both methods generate identical convolution results.

FIG. **6** shows the operation of a conventional matched filter for a length 6 signal. Operation begins at a moment in time indicated as shift 0. At this moment, 6 consecutive signal samples comprising an entire cycle of the signal are in the signal shift register **601**. Individual samples are labeled with uppercase indices A, B, C, D, E, and F. Code samples for the entire length 6 code are held in reference register **602** and are labeled with lowercase indices a, b, c, d, e, and f. At the time of shift 0, a vector multiplication and add is performed to generate the correlation result for shift 0. Each signal sample is multiplied by a corresponding code sample and the results are summed to yield correlation result **603**.

Next, the signal shift register **604** is advanced by one sample, as indicated by shift 1. The signal is periodic, so the new sample introduced at the left side of the register is identical to that shifted out to the right. The shifted contents of the register **604** are now samples with indices F, A, B, C, D, and E. The code is not shifted. The vector multiplication and addition now yields a correlation result **605** for shift 1. This process of shifting continues for 5 additional shifts, at which point all 6 correlation results making up the full convolution are available.

FIG. **7** illustrates how the same convolution result can be obtained through the method of partial correlations. As described, the invention requires that the code be factored into M segments of length K. In the simplified example of FIG. **7**, the length 6 code was factored into 3 segments of length 2, i.e. K=2 and M=3. Operation begins at a moment in time indicated at shift 0. At this moment, two signal samples are held in the signal shift register **701**. The signal samples are labeled with uppercase indices A and B. The 6 samples of the code are contained in 3 segments each of length 2. The first code segment **702** contains 2 code samples labeled with lowercase indices a and b. The signal is held in place for 3 partial correlation operations, resulting in partial correlation results **703***a*, **703***b* and **703***c*. The first partial correlation result is created by a vector multiplication and addition between the contents of the signal register and the

first code segment (segment 1). The second and third results are created by vector multiplications of the signal register with the second and third code segments respectively. Note that the signal register is held in place for a sufficient time for all three-vector multiplications to be performed, and that the code is not shifted during this time, rather different code segments are selected.

The partial correlation results are accumulated into the memory according to the signal paths **705**. For example, at shift 0, the partial correlation from the first code segment sums into the correlation result **704**. The partial correlation from the second segment sums into the correlation result **706** for shift 2. The partial correlation from the third segment contributes to the correlation result **708** for shift 4.

After three partial correlations, the signal is shifted. At this stage, indicated as shift 1, the signal register contains samples F and A. Again, three partial correlations are generated with the same three code segments as before. The results from these partial correlations contribute to correlation results **710**, **712**, **714** respectively for shifts 1, 3, and 5. The process continues for 4 additional signal shifts, at which time the full convolution result is available. As can be seen, the operation requires generating a total of 18 partial correlations that contribute to the 6 full results comprising the convolution.

The architecture described by FIG. **7** illustrates two important properties of the invention. First, it is apparent that the full convolution was produced for a length 6 code using only a shift register and vector multiplication and addition unit of length 2. This requires less circuitry than the FIG. **6** where these elements are of length 6. Second, in FIG. **7**, the code samples are accessed in fixed segments that are the same for each shift, and each segment is a separate non-overlapping section of the code. Thus, a simple lookup or register scheme can be used to provide the code to the vector multipliers, as will be discussed further in reference to FIG. **8** and FIG. **9**. These schemes require less circuitry than other architectures that might, for example, require large blocks of code bits to be made available in a more complex set of permutations. The invention also eliminates the need to provide code generation circuitry.

FIG. **8** shows a block diagram of one embodiment of a code lookup circuit **408** suitable for the invention. Table **801** contains stored values for all 1023 bits of each of 32 codes, for example in read-only memory (ROM) or hard-wired logic. The table **801** is organized as 32 sub-tables, one for each code. Each sub-table is further organized as M segments of length K where K×M=1023, and K and M are chosen as described previously. Multiplexer **802** selects a particular code based on a select value. The output of multiplexer **802** is a particular sub-table for the desired. Multiplexer **803** selects a particular segment based on a segment select value between 1 and M. The output of **803** is a particular code segment **804**, of length K, which contains code bits provided to code extender **409**.

It should be noted that multiplexer **803** must be high speed in order to allow the code segment to be changed each partial correlation, i.e. every two clock cycles. For this reason, it is necessary that all code bits be pre-stored in table **801**, as opposed to being generated on the fly in the traditional manner of a code generator.

The circuits of FIG. **8** are intended to be illustrative. In practice, there are many different circuit designs that are functionally equivalent. In particular, the process of logic synthesis used in modern ASIC design will lead to a certain pattern of gates that achieves a behavior equivalent to that described above but not necessarily using multiplexers in the manner described.

FIG. **9** shows a block diagram of an alternate embodiment of a code lookup circuit **408** suitable for the invention. The 1023 code bits corresponding to a particular code are held in 1023 dual-directional shift registers **901**, organized as M rows of length K. The shift registers operate in two modes: a running mode, and a loading mode.

In the running mode, each register **901** is configured to shift its sample to the register above it in the next row, except for the top row of registers that shifts to the bottom row of registers. The shift directions for running mode are indicated by solid arrows within **901**. By clocking all the registers, rows of code bits will circulate, such that at any one time the top row contains one of M code segments of length K. This top row of bits is provided to code extender **409**. The registers circulate rapidly, so that a different code segment is made available for each partial correlation.

In the loading mode, each register is configured to shift its sample to the register next in its row, except for the last column of registers, which shift to the first column of registers in the row above. The shift directions for loading mode are indicated by dotted arrows within **901**. The left hand lower shift register **904** is connected to code generator **902**. The code generator is a traditional code generator, capable of sequentially creating the 1023 code bits of a particular code based on a select value. When the code lookup circuit is configured for a particular, the registers are placed in the loading mode, and the generator is used to generate the bits of the code, which then clock through the registers. After all bits have been clocked through, the code will reside in the registers as M segments of length K. The circuit is then ready for use in the running mode.

Although various embodiments which incorporate the teachings of the present invention have been shown and described in detail herein, those skilled in the art can readily devise many other varied embodiments that still incorporate these teachings.

What is claimed is:

1. A method for computing correlations of a digital signal with a pseudorandom reference code, where said digital signal comprises a repeating code, the method comprising:

a) dividing a pseudorandom reference code into a plurality of code segments;

b) selecting a code segment;

c) forming an inner product between said selected code segment and a portion of said repeating code of said digital signal to produce a partial correlation;

d) repeating steps b) and c) to produce a plurality of partial correlations;

e) summing the plurality of partial correlations as each partial correlation is produced to form a plurality of correlations.

2. The method of claim **1** further comprising:

shifting said set of digital signal samples into a shift register, where, for each shift of the shift register, performing one or more said partial correlations.

3. The method of claim **1** where a size of said set of digital signal samples and a size of said set of pseudorandom reference code is an integer times a factor of 1023.

4. The method of claim **1** wherein said pseudorandom reference code is a C/A code of a global positioning system receiver.

5. The method of claim **4** wherein said set of pseudorandom reference code is generated by extending a code segment by replicating a C/A code bit to an integer number of samples.

6. The method of claim **5** further comprising:

generating said code segments from a lookup table by selecting one of a fixed number of non-overlapping segments of the C/A code.

7. The method of claim 6 where said code segments are generated from a set of circulating shift registers.

8. The method of claim 1 wherein the inner product is formed by multiplying each of the digital signal samples with a respective chip of the selected code segment and summing each multiplication result with other multiplication results to produce the partial correlation.

9. The method of claim 1 further comprising:

dividing the digital signal into in-phase and quadrature components; and separately performing steps a) through e) on the in-phase and quadrature components.

10. The method of claim 9 further comprising:

producing an energy signal representing an amount of energy in the plurality of correlations;

integrating the energy signal over a pre-defined period.

11. A receiver of global positioning system (GPS) signals comprising:

an RF/IF converter for filtering and frequency translating received GPS signals having a repeating code to form an IF signal;

an analog to digital converter for digitizing the IF signal;

a tuner for removing Doppler shift from the digitized signal and producing an in-phase (I) signal and a quadrature (Q) signal;

a decimation circuit for subsampling the I and Q signals;

a convolution processor for producing I and Q partial correlations by multiplying selected segments of a C/A reference code with portions of said repeating code of each of said subsampled I and Q signals;

a first accumulator for accumulating said I partial correlations to form a plurality of correlations between said I signal and the C/A reference code; and

a second accumulator for accumulating said Q partial correlations to form a plurality of correlations between said Q signal and the C/A reference code.

12. The receiver of claim 11 further comprising:

a signal normalizer for generating normalizer values representative of the magnitude or power of the I and Q signals;

magnitude accumulator for summing the normalizer values.

13. The receiver of claim 11 wherein convolution processor comprises:

a code generator for producing each C/A reference code comprising a code look up circuit and a code extender.

14. The receiver of claim 11 wherein convolution processor comprises:

a first shift register for storing a segment of said subsampled I signal; and

a second shift register for storing a segment of said subsampled Q signal.

15. The receiver of claim 11 wherein convolution processor comprises:

a first vector multiplier for producing I partial correlations of said subsampled I signal; and

a second vector multiplier for producing Q partial correlations of said subsampled Q vector.

16. The receiver of claim 11 wherein said first and second accumulators each comprise:

an adder; and

a memory.

17. The receiver of claim 11 further comprising:

a plurality of processing channels, where each channel produces a convolution for a different GPS signal.

18. The receiver of claim 17 further comprising:

a computer for computing a position location using a plurality of convolutions.

19. A receiver of global positioning system (GPS) signals comprising:

an RE/IF converter for filtering and frequency translating received GPS signals having a repeating code to form an IF signal;

an analog to digital converter for digitizing the IF signal;

a tuner for removing Doppler shift from the digitized signal and producing an in-phase (I) signal and a quadrature (Q) signal;

a decimation circuit for subsampling the I and Q signals;

a convolution processor for producing I and Q partial correlations by multiplying selected segments of a C/A reference code with portions of said repeating code of each of said subsampled I and Q signals, and further comprising a code generator for producing each C/A reference code comprising a code look up circuit and a code extender, a first shift register for storing a segment of said subsampled I signal and a second shift register for storing a segment of said subsampled Q signal, a first vector multiplier for producing I partial correlations of said subsampled I signal and a second vector multiplier for producing Q partial correlations of said subsampled Q signal;

a first accumulator for accumulating said I partial correlations to form a plurality of correlations between said I signal and the C/A reference code;

a second accumulator for accumulating said Q partial correlations to form a plurality of correlations between said Q signal and the C/A reference code;

a signal normalizer for generating normalizer values representative of the magnitude or power of the I and Q signals; and

magnitude accumulator for summing the normalizer values.

20. The receiver of claim 19 wherein said first and second accumulators each comprise:

an adder; and

a memory.

21. The receiver of claim 19 further comprising:

a plurality of processing channels, where each channel produces a convolution for a different GPS signal.

22. The receiver of claim 19 further comprising:

a computer for computing a position location using a plurality of convolutions.

23. A method of computing correlations between a pseudorandom reference code and a digital signal having a repeating code, the method comprising:

a) dividing a pseudorandom reference code into a plurality of code segments;

b) selecting a portion of the repeating code of the digital signal;

c) selecting a code segment;

d) forming an inner product between said selected code segment and the selected portion of said repeating code to produce a partial correlation;

e) repeating steps c) and d) to produce a plurality of partial correlations;

f) repeating steps b) through e) for a plurality of portions of the repeating code;

g) accumulating the plurality of partial correlations as each partial correlation is produced to form a correlation; and

h) forming a plurality of correlations.

* * * * *