INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE INGENIERÍA MECÁNICA Y ELÉCTRICA

UNIDAD PROFESIONAL "ADOLFO LOPEZ MATEOS"

SECCIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

"IMPLEMENTACIÓN DE LOS ALGORITMOS DE SEGUIMIENTO DE LA SEÑAL GPS SOBRE DISPOSITIVOS LÓGICOS PROGRAMABLES (FPGA)"

T E S I S

QUE PARA OBTENER EL GRADO DE MAESTRO EN CIENCIAS EN INGENIERÍA DE TELECOMUNICACIONES

P R E S E N T A:

ING. CESAR DANIEL PÉREZ MONTOYA

DIRECTORES DE TESIS:

M. en C. MIGUEL SÁNCHEZ MERAZ
DR. AMADEO JOSÉ ARGÜELLES CRUZ

México, DF. Junio del 2009

INSTITUTO POLITÉCNICO NACIONAL

SECRETARÍA DE INVESTIGACIÓN Y POSGRADO

SIP-14

ACTA DE REVISIÓN DE TESIS

En la Ciudad de México, D. F. siendo las **17:00** horas del día **26** del mes de **JUNIO** del **2009** se reunieron los miembros de la Comisión Revisora de Tesis designada por el Colegio de Profesores de Estudios de Posgrado e Investigación de E.S.I.M.E. para examinar la tesis titulada:

"IMPLEMENTACIÓN DE LOS ALGORITMOS DE SEGUIMIENTO DE LA SEÑAL GPS SOBRE DISPOSITIVOS LÓGICOS PROGRAMABLES (FPGA)"

Presentada por el alumno:

PÉREZ	**MONTOYA**	**CÉSAR DANIEL**
Apellido paterno	Apellido materno	Nombre(s)

Con registro:

B	0	7	1	4	8	1

aspirante de:

MAESTRO EN CIENCIAS EN INGENIERÍA DE TELECOMUNICACIONES

Después de intercambiar opiniones los miembros de la Comisión manifestaron ***SU APROBACIÓN DE LA TESIS***, en virtud de que satisface los requisitos señalados por las disposiciones reglamentarias vigentes.

LA COMISIÓN REVISORA

Los Directores de tesis

M. EN C. MIGUEL SÁNCHEZ MERAZ

DR. AMADEO JOSÉ ARGÜELLES CRUZ

Presidente

DR. VLADIMIR KAZAKOV

Segundo Vocal

DR. AMADEO JOSÉ ARGÜELLES CRUZ

Tercer Vocal

DR. SALVADOR ALVAREZ BALLESTEROS

Secretario

DRA. MARTHA CECILIA GALÁZ LARIOS

Suplente

M. EN C. MARCO ANTONIO ACEVEDO MOSQUEDA

El Presidente del Colegio

DR. JAIME ROBLES GARCIA

SECCION DE ESTUDIOS DE POSGRADO E INVESTIGACION

INSTITUTO POLITÉCNICO NACIONAL

SECRETARÍA DE INVESTIGACIÓN Y POSGRADO

CARTA CESIÓN DE DERECHOS

En la Ciudad de México el día 29 del mes Junio del año 2009, el que suscribe Cesar Daniel Pérez Montoya alumno del Programa de Maestría en Ciencias en Ingeniería de Telecomunicaciones con número de registro B071481, adscrito a la Escuela Superior de Ingeniería Mecánica y Eléctrica, manifiesta que es autor intelectual del presente trabajo de Tesis bajo la dirección del M. en C. Miguel Sánchez Meraz y el Dr. Amadeo José Argüelles Cruz y cede los derechos del trabajo intitulado Implementación de los Algoritmos de Seguimiento de la señal GPS sobre Dispositivos Lógicos Programables (FPGA), al Instituto Politécnico Nacional para su difusión, con fines académicos y de investigación.

Los usuarios de la información no deben reproducir el contenido textual, gráficas o datos del trabajo sin el permiso expreso del autor y/o director del trabajo. Este puede ser obtenido escribiendo a la siguiente dirección cperezm0609@ipn.mx. Si el permiso se otorga, el usuario deberá dar el agradecimiento correspondiente y citar la fuente del mismo.

Ing. Cesar Daniel Pérez Montoya

Nombre y firma

Resumen

En este trabajo se presenta el estudio e implementación del algoritmo de seguimiento de un receptor de señales GPS basado en una arquitectura de radio definido por software como una aportación para que en un futuro se cuente con un receptor GPS basado en software y que tenga como dispositivo principal un FPGA.

La implementación del algoritmo se divide en cuatro partes principales: estudio e implementación del lazo de seguimiento de código, estudio e implementación del lazo de seguimiento de portadora, juntar estos dos lazos de seguimiento para que al final tengamos uno solo que a la salida nos de cómo resultado los datos de navegación y por último implementar el lazo de seguimiento completo en lenguaje C para poder pasarlo al FPGA.

La implementación del algoritmo sobre dispositivos lógicos programables FPGA es para poder tener un receptor con características cognitivas el cual nos va a permitir simular otros sistemas de navegación sin necesidad de cambiar dispositivos de hardware con solo actualizar algunos parámetros de configuración dentro de la aplicación de software desarrollada.

Abstract

This thesis presents the study and implementation of the tracking algorithm of a GPS receiver based in architecture of software defined radio as a contribution to in a future have a GPS receiver based on software that has as a primary device FPGA.

The implementation of the algorithm is divided into four main parts: study and implementation of the code tracking loop, study and implementation of the carrier tracking loop, together these two tracking loops to the end we have only one that at the exit us of as resulted the data of navigation, and finally implement the tracking loop in C language in to pass the FPGA.

The implementation of the algorithm on FPGA programmable logic devices is to take a receiver with cognitive features which will allow us to simulate other navigation system without changing hardware devices just to update some settings within the software application developed.

Contenido

Lista de Figuras

Figuras del capítulo 1

Figuras del capítulo 2

Figuras del capítulo 3

Figuras del capítulo 4

Figuras del capítulo 5

Figuras del capítulo 6

Lista de Tablas

Tablas del capítulo 2

Tablas del capítulo 4

Tablas del capítulo 6

Abreviaturas

ADC: Convertidor Analógico a Digital.

ASIC: Circuito Integrado de Aplicación Específica.

BPSK: Modulación por Desplazamiento de Fase Binaria.

C/A: Código de Adquisición Aproximado.

CDMA: Acceso Múltiple por División de Código.

DLL: Lazo de Bloqueo de Retraso. Usado para sincronizar la réplica de código generada localmente.

DSSS: Espectro Ensanchado por Secuencia Directa.

EDK: Kit de Desarrollo Embebido.

FEC: Corrección Anticipada de Errores.

FLL: Lazo de Bloqueo de Frecuencia.

FPGA: Arreglo de Compuertas de Campo Programables.

GNSS: Sistema Global de Navegación por Satélite.

GPS: Sistema de Posicionamiento Global.

HOW: Palabra Handover. 17 bits, versión truncada de TOW.

IF: Frecuencia Intermedia.

IP: Propiedad Intelectual.

ISE: Entorno de Software Integrado.

LFSR: Registro de Corrimiento con Retroalimentación Lineal.

LO: Oscilador Local.

L1: Banda de frecuencia GPS: $f_{L1} = 1575.42$ MHz

L2: Banda de frecuencia GPS: $f_{L2} = 1227.60$ MHz

LSB: Bits menos significativos.

MSB: Bits más significativos.

NCO: Oscilador Controlado Numéricamente.

PLL: Lazo de Bloqueo de Fase. Usado para sincronizar la réplica de portadora generada localmente.

PPS: Servicio de Localización Precisa.

PRN: Ruido Pseudo – Aleatorio.

P(Y): Código de Precisión.

RF: Radio Frecuencia.

SDR: Radio Definido por Software.

SPS: Servicio de Localización Estándar. Calcula la posición basándose en la señales de código C/A.

SVN: Número de Vehículo en el Espacio.

TLM: Palabra telemetría. 8 bits de preámbulo usados para sincronizar el mensaje de navegación.

TOW: Tiempo de Semana. La semana GPS comienza el sábado a medianoche.

UHF: Ultra Alta Frecuencia.

USB: Bus Serial Universal.

VCO: Oscilador Controlado por Voltaje.

XPS: Plataforma de Estudio de Xilinx

Antecedentes

Así como muchas de las innovaciones en alta tecnología de los últimos 60 años fueron originalmente desarrolladas para la milicia, GPS no fue la excepción. En 1973 el Departamento de Defensa de Estados Unidos, con una inversión de 12 mil millones de dólares (MMD), empezó a desarrollar el proyecto Navstar GPS para proveer información precisa de localización para aeronaves, navíos, submarinos, tanques de guerra, etc. No fue sino hasta 1983 que Navstar GPS expandiera sus señales para uso civil; esto permitió a la aviación y a otros medios de transportación una mejor precisión en sus sistemas de navegación.

La constelación Navstar GPS compuesta de 24 satélites envía dos tipos de señales a la tierra que difieren en niveles de precisión. El primer tipo conocido como Servicio de Localización Precisa (PPS – *Precise Positioning Service*) es una señal encriptada para usos militares, la cual fue diseñada para niveles de aproximación de 15 a 30 metros, pero en la práctica se han logrado precisiones en el orden de 10 metros. La segunda señal conocida como Servicio de Localización Estándar (SPS – *Standard Positioning Service*) es una señal estándar para uso civil, tiene una precisión de 100 a 150 metros. La señal SPS es degradada a propósito por el Departamento de Defensa utilizando una técnica conocida como Disponibilidad Selectiva (SA – *Selective Availability*).

Los receptores GPS han estado disponibles en el mercado por más de 20 años, sin embargo como sucede con otros tipos de radio receptores, estos estaban basados fundamentalmente en una electrónica de tipo analógica y se utilizaban microprocesadores únicamente para el cálculo de la posición [22]. Esto resultaba en receptores voluminosos y

con mucho consumo de potencia. En 1989 con el lanzamiento del satélite GPS de fase II todo el procesamiento de señales analógicas fue remplazado por microprocesadores y algunos circuitos integrados adicionales.

En la actualidad los receptores GPS están basados en el uso de ASIC's para el procesamiento de señales y microprocesadores rápidos para ejecutar cálculos. Los ASIC resultan ser muy efectivos para el procesamiento de señales sin embargo resulta muy costoso el rediseño de sistemas basados en este tipo de dispositivos ya que no pueden ser reprogramados como un microprocesador.

Por aproximadamente una década se han desarrollado receptores GPS basados en software, sin embargo solo hasta hace poco se ha podido disponer de la capacidad de procesamiento en microprocesadores o en computadoras personales que hacen posible implementar un receptor completo que pueda operar en tiempo real.

Justificación

Un receptor de señales GPS basado en software ofrecerá la posibilidad de aplicar rediseños de una forma directa y con costos bajos. Este receptor, al contrario de un receptor ordinario, realiza todas sus operaciones en un microprocesador programable, haciendo que el sistema sea altamente flexible. Esta flexibilidad permitirá verificar la efectividad de diferentes algoritmos y ajustar la operación del receptor en el caso de que sucedan cambios en la señal GPS que se recibe. Esta flexibilidad del receptor resultará de mucho valor ya que a principios de la década siguiente estarán disponibles nuevas señales GPS (L2 y L5), así como Galileo, el nuevo sistema de navegación satelital de Europa, que será compatible con GPS.

Debido a lo ya mencionado, esta tesis tiene como finalidad aportar con el estudio e investigación del algoritmo de seguimiento de la señal (seguimiento de código y portadora) ya que es una parte fundamental dentro de la arquitectura de un receptor de señales GPS basado en software para que en conjunto con los demás algoritmos (adquisición, extracción de los datos de navegación, pseudorango y cálculo de posición) en un futuro se pueda tener un receptor de señales GPS basado en una arquitectura de radio definido por software.

Objetivo

Desarrollar e implementar el bloque de seguimiento (seguimiento de código y seguimiento de portadora) de la señal para un receptor GNSS de señales GPS, basado en una arquitectura de Radio Definido por Software, utilizando procesadores programables FPGA.

Objetivos particulares:

- Investigar el funcionamiento del bloque de seguimiento de la señal GPS.
- Investigar e implementar en Matlab y en lenguaje C el lazo de seguimiento de código.
- Investigar e implementar en Matlab y en lenguaje C el lazo de seguimiento de portadora.
- Implementar el lazo de seguimiento completo (seguimiento de código y portadora) en un procesador programable usando el kit de desarrollo XtremeDSP Pro.

Capítulo 1

Introducción

En este capítulo daremos una descripción general del sistema GPS y la relación que podrían tener estos con el Radio Definido por Software. Aquí estudiamos una visión general de los receptores de señales GPS implementados en una arquitectura de Radio Definido por Software. Además, se menciona la organización de la tesis.

1.1 El Sistema de Posicionamiento Global (GPS).

La habilidad del GPS para determinar con precisión la localización de un usuario utilizando un receptor GPS en cualquier lugar bajo cualquier condición climática, atrajo a millones de usuarios. El Sistema de Posicionamiento Global (GPS – *Global Positioning System*) es un sistema de navegación basado en satélites de comunicación que fue desarrollado por el Departamento de Defensa a principios de los 70's. Inicialmente fue desarrollado para cumplir las necesidades de la Milicia de los Estados Unidos. Sin embargo, mas tarde fue permitido su uso a personas civiles, y es ahora un sistema de uso dual que puede ser utilizado tanto por militares como civiles.

FIGURA 1.1. Constelaciones GPS.

El GPS envía continuamente información de posición y tiempo en cualquier parte del mundo bajo cualquier condición climática. El GPS es un sistema que ha sido utilizado por un número ilimitado de usuarios. Esto es, los usuarios solo pueden recibir las señales satelitales del GPS.

El GPS consiste de una constelación de 24 satélites operacionales y 3 satélites de respaldo. Esta constelación inicio operaciones en 1990. Para llevar a cabo la cobertura continua mundial, los satélites del sistema GPS son agrupados en órbitas de 4 satélites, así forman un total de 6 órbitas, como se muestra en la figura 1.1 Con esta constelación geométrica, de 4 a 10 satélites serán visibles en cualquier parte del mundo, pero solo 4 satélites son necesarios para determinar la información de la localización. Las órbitas de los satélites del GPS se asemejan a una forma circular (una forma elíptica con un máximo de excentricidad de 0.01) con una inclinación de 55º del ecuador. El semieje mayor de una órbita GPS es alrededor de 26,560 Km (por ejemplo, la altura del satélite es de 20,000 Km sobre la superficie terrestre). El periodo correspondiente de la órbita GPS es de 12 hrs. siderales, aproximadamente 11 hrs. 58 min [41].

Cuando se determina la posición, el receptor que se utiliza para ello localiza automáticamente como mínimo tres satélites de la red, de los que recibe unas señales

indicando la posición y el reloj de cada uno de ellos. Con base en estas señales, el aparato sincroniza el reloj del GPS y calcula el retraso de las señales, es decir, la distancia al satélite. Por triangulación calcula la posición en que éste se encuentra. La triangulación en el caso del GPS, a diferencia del caso 2D que consiste en averiguar el ángulo respecto de puntos conocidos, se basa en determinar la distancia de cada satélite respecto al punto de medición. Conocidas las distancias, se determina fácilmente la propia posición relativa respecto a los tres o cuatro satélites. Conociendo además las coordenadas o posición de cada uno de ellos por la señal que emiten, se obtiene la posición absoluta o coordenada reales del punto de medición. También se consigue una exactitud extrema en el reloj del GPS, similar a la de los relojes atómicos que llevan a bordo cada uno de los satélites.

La antigua Unión Soviética tenía un sistema similar llamado GLONASS, ahora gestionado por la Federación Rusa.

Actualmente la Unión Europea está desarrollando su propio sistema de posicionamiento por satélite, denominado Galileo.

1.1.1 Historia del GPS.

En 1957, la Unión Soviética lanzó al espacio el satélite **Sputnik I**, que era monitorizado mediante la observación del Efecto Doppler de la señal que transmitía. Debido a este hecho, se comenzó a pensar que, de igual modo, la posición de un observador podría ser establecida mediante el estudio de la frecuencia Doppler de una señal transmitida por un satélite cuya órbita estuviera determinada con precisión.

La Armada estadounidense rápidamente aplicó esta tecnología, para proveer a los sistemas de navegación de sus flotas de observaciones de posiciones actualizadas y precisas. Así surgió el sistema **TRANSIT**, que quedó operativo en 1964 y hacia 1967 estuvo disponible, además, para uso comercial.

Las actualizaciones de posición, en ese entonces, se encontraban disponibles cada 40 minutos y el observador debía permanecer casi estático para poder obtener información adecuada.

Posteriormente, en esa misma década y gracias al desarrollo de los relojes atómicos, se diseñó una constelación de satélites, portando cada uno de ellos uno de estos relojes y estando todos sincronizados con base en una referencia de tiempo determinada.

En 1953 se combinaron los programas de la Armada y el de la Fuerza Aérea de los Estados Unidos (este último consiste en una técnica de transmisión codificada que proveía datos precisos usando una señal modulada con un código de ruido pseudo-aleatorio (PRN – *Pseudo-Random Noise*)), en lo que se conoció como **NAVSTAR GPS**.

Entre 1978 y 1985 se desarrollaron y lanzaron once satélites prototipo experimentales NAVSTAR, a los que siguieron otras generaciones de satélites, hasta completar la constelación actual, a la que se declaró con "capacidad operacional inicial" en diciembre de 1993 y con "capacidad operacional total" en abril de 1995.

1.1.2 Características técnicas y prestaciones.

El Sistema de Posicionamiento Global (GPS) lo componen:

1. Sistema de satélites: Está formado por 24 unidades con trayectorias sincronizadas para cubrir toda la superficie del globo terráqueo. Más concretamente, repartidos en 6 planos orbitales de 4 satélites cada uno. La energía eléctrica que requieren para su funcionamiento la adquieren a partir de dos paneles compuestos de celdas solares adosados a sus costados.
2. Estaciones terrestres: Envían información de control a los satélites para controlar las órbitas y realizar el mantenimiento de toda la constelación.
3. Terminales receptores: Indican la posición en la que están, conocidas también como Unidades GPS.

FIGURA 1.2. Operadora de satélites controlando la constelación NAVSTAR – GPS, en la base aérea de Schriever.

1.1.2.1 Segmento espacial.

- Satélites en la constelación: 24 (4 x 6 órbitas)
  - Altitud: 20,200 km
  - Período: 11 hrs 56 min.
  - Inclinación: 55º (respecto al ecuador terrestre)
  - Vida útil: 7.5 años
- Segmento de control (estaciones terrestres)
  - Estación principal: 1
  - Antena de tierra: 4
  - Estación monitora (de seguimiento): 5
- Señal de RF
  - Frecuencia portadora:
    - ✓ Civil – 1575.42 MHz (L1). Utiliza el Código de Adquisición Aproximado (C/A).
    - ✓ Militar – 1227,60 MHz (L2). Utiliza el Código de Precisión (P) cifrado.
      - Nivel de potencia de la señal: -160 dBW (en superficie tierra).
      - Polarización: circular dextrógira

- Exactitud
  - Posición: aproximadamente 15 m (el 95%)
  - Hora: 1 ns
- Cobertura: mundial
- Capacidad de usuarios: ilimitada
- Sistema de coordenadas:
  - Sistema Geodésico Mundial 1984 (WGS84)
  - Centrado en la Tierra, fijo.
- Integridad: tiempo de notificación 16 minutos o mayor. NO ES SUFICIENTE PARA LA AVIACIÓN CIVIL.
- Disponibilidad: 24 satélites

FIGURA 1.3. Lanzamiento de satélites para la constelación NAVSTAR – GPS mediante un cohete delta.

1.1.3 Evolución del sistema GPS.

El GPS está evolucionando hacia un sistema más sólido (GPS III), con una mayor disponibilidad y que reduzca la complejidad de las aumentaciones GPS. Algunas de las mejoras previstas comprenden:

- Incorporación de una nueva señal en L2 para uso civil.
- Adición de una tercera señal civil (L5): 1176.45 MHz
- Protección y disponibilidad de una de las dos nuevas señales para servicio de Seguridad Para la Vida (SOL).
- Mejora en la estructura de las señales.
- Incremento en la potencia de señal (L5 tendrá un nivel de potencia de -154 dB).
- Mejora en la precisión (1 – 5 m).
- Aumento en el número de estaciones monitorizadas: 12 (el doble)
- Permitir mejor interoperabilidad con la frecuencia L1 de Galileo

El programa GPS III persigue el objetivo de garantizar que el GPS pueda satisfacer requisitos militares y civiles para los próximos 30 años. Este programa se está desarrollando para utilizar un enfoque en 3 etapas (una de las etapas de transición es el GPS II); muy flexible, permite cambios futuros y reduce riesgos. El desarrollo de satélites GPS III comenzó en 2005, y el primero de ellos estará disponible para su lanzamiento en 2012, con el objetivo de lograr la transición completa de GPS III en 2017 [44]. Los desafíos son los siguientes:

- Representar los requisitos de usuarios, tanto civiles como militares, en cuanto a GPS.
- Limitar los requisitos GPS III dentro de los objetivos operacionales.
- Proporcionar flexibilidad que permita cambios futuros para satisfacer requisitos de los usuarios hasta 2030.
- Proporcionar solidez para la creciente dependencia en la determinación de posición y de hora precisa como servicio internacional.

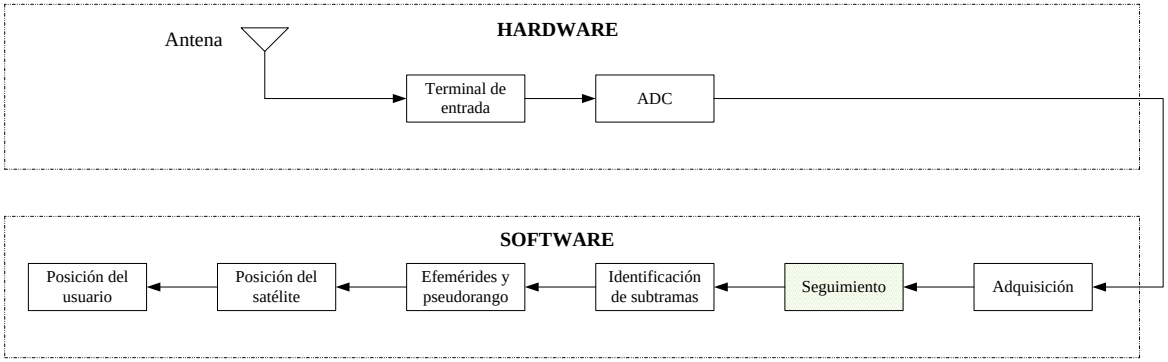

FIGURA 1.4. Un receptor GPS basado en software básico.

1.2 Receptor GPS básico.

El receptor GPS básico discutido en esta tesis es el mostrado en la figura 1.4. Las señales transmitidas de los satélites GPS son recibidas por la antena. A través de la cadena de radiofrecuencia (RF) la señal de entrada es amplificada a una amplitud adecuada y la frecuencia es convertida a una frecuencia de salida deseada. Un convertidor analógico a digital (ADC) es usado para digitalizar la señal de salida. La antena, señales de RF, y el ADC es el hardware usado en el receptor [26].

Después de que las señales son digitalizadas, el software es usado para su procesamiento, y es por eso que a esta tesis se le da un enfoque de software. La adquisición significa encontrar la señal de un cierto satélite.

El lazo de seguimiento (marcado en la figura 1.4) es usado para encontrar la transición de la fase de los datos de navegación, para mayor detalles ver capitulo 4, aquí cabe mencionar que el seguimiento es la parte central de esta tesis aunque en algunos capítulos se hable en general sobre los receptores GPS basados en software no perdamos de vista que el objetivo es el estudio e implementación del bloque de seguimiento del receptor de señales GPS. En un receptor convencional, la adquisición y el seguimiento son realizados por hardware. De la transición de fase de los datos de navegación; las subtramas y los datos de navegación pueden ser obtenidos. Los datos de efemérides y pseudorango

pueden ser obtenidos de los datos de navegación. Los datos de efemérides son usados para obtener la posición satelital. Finalmente, la posición del usuario puede ser calculada por la posición satelital y el pseudorango. Tanto el hardware usado para coleccionar datos digitalizados y el software usado para encontrar la posición del usuario serán discutidos en algunos capítulos de esta tesis.

1.3 Criterios de presentación.

Hay dos criterios posibles de implementación. Uno, es una manera simple de seguir el diagrama de flujo de la señal mostrado en la figura 1.4. En este criterio se comienza con la estructura de la señal del sistema GPS y los métodos para procesar la señal para obtener la información necesaria. Esta información será usada para calcular las posiciones de los satélites y los pseudorangos. Usando la posición de los satélites y los pseudorangos, puede ser encontrada la posición del usuario. En este criterio, la discusión será simple, de un objetivo a otro. Sin embargo, la desventaja de este criterio es que los lectores no tendrán una idea clara ya que estos pasos son necesarios. Entenderían el concepto de la operación del GPS solo después de leer un libro completo sobre GPS.

El otro criterio es comenzar con el concepto básico del GPS. Este criterio empezara con el concepto básico de encontrar la posición del usuario de la posición satelital. La descripción de la constelación satelital será presentada. La información detallada de la órbita del satélite está contenida en los datos GPS. Así que para obtener estos datos, la señal GPS debe ser seguida. El código C/A de las señales GPS será presentado. Cada satélite tiene un único código C/A. Un receptor puede realizar la adquisición sobre el código C/A para encontrar la señal. Una vez que el código C/A de un cierto satélite es encontrado, la señal puede ser seguida (seguimiento de código y portadora), ver capitulo 4. Si se realiza un seguimiento correcto, esto ayuda a obtener los datos de navegación. De estos datos, la posición del satélite puede ser encontrada. El pseudorango puede ser obtenido comparando

el tiempo que un cierto dato llega al receptor. La posición del usuario puede ser calculado de la posición satelital y el pseudorango de varios satélites.

Esta tesis toma este segundo criterio para presentar el material debido a que da una clara idea de la función del GPS desde el comienzo; no olvidando que en esta tesis además de que se da esta descripción detallada del receptor GPS el tema principal es el seguimiento de la señal GPS.

1.4 Radio Definido por Software.

Esta tesis usa el concepto de Radio Definido por Software para presentar el tema principal del trabajo de investigación. La idea de radio por software es de usar un convertidor analógico a digital (ADC) para cambiar la señal de entrada en datos digitales en la fase más temprana posible en el receptor. En otras palabras, la señal de entrada es digitalizada lo más cerca a la antena como sea posible. Una vez que la señal es digitalizada, un procesamiento digital de la señal será usado para obtener la información necesaria. El objetivo principal del radio por software es minimizar el uso de hardware en un radio.

El propósito principal de usar el concepto de radio definido por software para presentar este tema es de ilustrar la idea del seguimiento y la adquisición de la señal GPS. Aunque usando hardware para realizar la adquisición y seguimiento de la señal puede también describir la función de un receptor GPS, esto parece que usando software puede proporcionar una clara idea de la adquisición y seguimiento de señales. Además, un enfoque en software proporcionara un mejor entendimiento de la función del receptor debido a que algunos de los cálculos pueden ser ilustrados con programas.

1.5 Ventajas potenciales del Radio Definido por Software.

Un aspecto importante del uso de un enfoque en software para construir un receptor GPS es que este enfoque puede drásticamente desviarse del enfoque en hardware convencional. Por ejemplo, el usuario puede tomar un instante de los datos y procesarlos para encontrar la ubicación y de manera continua realizar el seguimiento de la señal. Teóricamente, 30 segundos de datos son suficientes para encontrar la ubicación del usuario. Esto es especialmente útil cuando los datos no pueden ser coleccionados de una manera continúa. El enfoque de software es muy flexible. Este puede procesar datos tomados de varios tipos de hardware. Por ejemplo, un sistema puede tomar datos complejos conocidos como canales en fase y en cuadratura (I y Q). Otro sistema puede tomar datos reales de un canal. El dato puede ser fácilmente cambiado de una forma a otra. Uno puede también generar programas para procesar señales complejas de programas de procesamiento para señales reales o viceversa con algunas modificaciones simples. Un programa puede ser usado para procesar señales digitalizadas con varias frecuencias de muestreo. Por lo tanto, un enfoque en software puede casi ser considerado como un componente de hardware independiente.

Nuevos algoritmos pueden ser fácilmente desarrollados sin cambiar el diseño de hardware. Esto es especialmente útil para solucionar algunos nuevos problemas que se llegasen a presentar o probar nuevos algoritmos desarrollados para una causa en común.

1.6 Organización de la tesis.

Esta tesis contiene 6 capítulos. El capítulo 2 describe las propiedades de la señal transmitida por los satélites GPS y recibida por la antena del receptor GPS. Para comenzar el diseño de un receptor GPS de una sola frecuencia basado en software es necesario

conocer las características de la señal y datos transmitidos de los satélites GPS. Primero, será presentada una visión general de las propiedades más importantes de las señales GPS. En seguida hablaremos de los tipos de modulación ocupados en navegación satelital. Después pasaremos a una descripción de la creación de las señales GPS de los satélites. Finalmente, las propiedades de las secuencias dispersas, se describirán las secuencias PRN y los datos de navegación.

El capítulo 3 discute la estructura básica de un receptor GPS. La estructura descrita es la más comúnmente usada. La señal llega a la antena, y la señal de radio frecuencia (RF) es filtrada y es convertida a una frecuencia intermedia (IF) por la terminal de entrada de RF. Después de la conversión de bajada, la señal analógica es muestreada por un convertidor A/D proporcionando una señal digital a las partes restantes del receptor. El procesamiento de la señal es llevado a cabo en canales – uno para cada señal seguida. El último paso en el receptor GPS es aplicar los algoritmos para el cálculo de posición.

El capítulo 4 da la parte teórica del tema de nuestra tesis. Aquí se describirán los métodos de seguimiento de la señal GPS. Primero, se explica un esquema de demodulación para poder extraer los datos de navegación. Posteriormente hablamos un poco sobre los PLL de segundo orden que son usados en el seguimiento de portadora. Por último, se explican los métodos de seguimiento de la señal GPS: seguimiento de portadora y seguimiento de código y la decodificación de los datos de navegación.

El capítulo 5 presenta la arquitectura de un receptor GPS basado en software. Explicaremos la aplicación encargada de grabar las tramas GPS sobre la computadora y el uso de software de procesamiento de señales para poder llevar el algoritmo implementado (lazo de seguimiento) sobre el microprocesador de un FPGA. Y por último en este capítulo se darán los detalles del kit de desarrollo XtremeDSP Pro de Nallatech.

En el capítulo 6 se presenta la implementación de la arquitectura mostrada en el capítulo 5 y se muestran los resultados obtenidos de la implementación del algoritmo de seguimiento de la señal GPS realizado en Matlab y en lenguaje C. Por último, daremos los detalles de la implementación del algoritmo de seguimiento de señales GPS sobre el microprocesador PowerPC 405 de nuestro kit de desarrollo XtremDSP Pro.

En esta tesis se hace uso de la herramienta de simulación Matlab para la implementación del algoritmo de seguimiento de la señal GPS, así que todas las figuras que de alguna manera tengan que ver con alguna simulación fueron implementadas con ayuda de esta herramienta y también se hace uso de lenguaje C ya que nos ayuda a la implementación del algoritmo en un microprocesador reprogramable dentro de un FPGA de la familia Virtex-II Pro.

Capítulo 2

Señales GPS

En este capítulo describiremos las propiedades de la señal transmitida por los satélites GPS y recibida por la antena del receptor GPS. Para comenzar el diseño de un receptor GPS de una sola frecuencia basado en software es necesario conocer las características de la señal y datos transmitidos de los satélites GPS. Primero, será presentada una visión general de las propiedades más importantes de las señales GPS. En seguida hablaremos de los tipos de modulación ocupados en navegación satelital. Después pasaremos a una descripción de la creación de las señales de los satélites GPS. Finalmente, las propiedades de las secuencias dispersas; se describirán las secuencias PRN y los datos de navegación.

2.1 Señales y Datos.

Las señales GPS son transmitidas en dos radiofrecuencias en la banda UHF. La banda UHF cubre la banda de frecuencias de 500 MHz a 3 GHz. Estas frecuencias son conocidas como L1 y L2 y son derivadas de una frecuencia en común, $f_0 = 10.23\ MHz$:

$$f_{L1} = 154 f_0 = 1575.42\ MHz \tag{2.1}$$

$$f_{L2} = 120 f_0 = 1227.60\ MHz \tag{2.2}$$

Las señales están compuestas de las siguientes tres partes:

Portadora: La onda portadora con frecuencia f_{L1}o f_{L2},

Datos de navegación: Los datos de navegación contienen información con respecto a las órbitas satelitales. Esta información es actualizada de todos los satélites de las estaciones terrestres en el Segmento de Control GPS. Los datos de navegación tienen una tasa de bit de 50 bps.

Secuencia dispersa: Cada satélite tiene dos únicas secuencias dispersas o códigos. El primero es el código de adquisición aproximado (C/A), y el otro es el código de precisión encriptado (P(Y)). El código C/A es una secuencia de 1023 chips[1]. El código es repetido cada milisegundo dando una tasa de 1.023 MHz. El código P es un código más largo ($\approx 2.35 \times 10^4$ chips) con una tasa de 10.23 MHz. Este se repite cada semana comenzando a principios de cada semana GPS la cual empieza la noche del sábado para domingo. El código C/A es solamente modulado sobre la portadora L1 mientras el código P(Y) es modulado tanto con la portadora L1 como con la portadora L2.

2.2 Modulación para Navegación Satelital.

Modulación por desplazamiento de fase binaria (BPSK – Binary Phase Shift Keying) es un esquema de señalización digital simple en el cual una portadora RF es transmitida "como es" o con un cambio de fase de 180º sobre intervalos sucesivos en el tiempo dependiendo de si un 1 o un 0 digital se transmitió [23]. Una señal BPSK, es ilustrada en la figura 2.1, puede ser pensada como el producto de dos formas de onda en el tiempo – la portadora RF no modulada y la forma de onda de los datos toman un valor de +1 o -1 para cada intervalo de $T_b = 1/R_b$ segundos, donde R_b es la tasa de los datos en bits por segundos.

[1] Un chip corresponde a un bit. Es simplemente llamado chip para enfatizar que este no lleva información alguna.

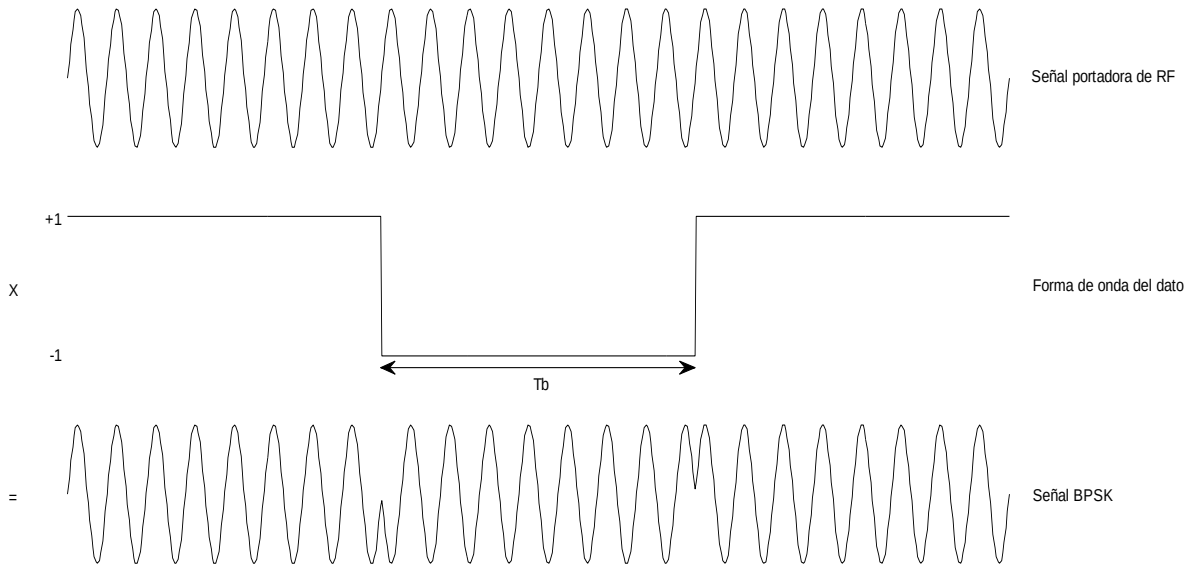

FIGURA 2.1. Modulación BPSK

La amplitud de la forma de onda de los datos para el *k*-ésimo intervalo de T_b segundos puede ser generada del *k*-ésimo bit de los datos que se transmite usando el mapeo [0, 1] $\rightarrow$ [-1, +1] o [0, 1] $\rightarrow$ [+1, -1]. En muchos sistemas, se emplea la corrección anticipada de errores (FEC – *Forward Error Correction*), por donde los bits redundantes (más bits de la información original) son transmitidos a través del canal, y que permite al receptor detectar y corregir algunos errores que pueden ser introducidos por ruido, interferencia, o desvanecimiento. Cuando la FEC es empleada, un método habitual es remplazar $T_b$con T_s y $R_b$con R_s para distinguir los símbolos de los datos (actualmente transmitidos) de los bits de los datos (que contienen la información antes del FEC). La forma de onda de los datos sola es considerada una señal en banda base, lo que significa que su contenido de frecuencias se concentra alrededor de 0 Hz en lugar de una frecuencia portadora. La modulación por una frecuencia portadora de RF centra el contenido de frecuencias por la frecuencia portadora, creando lo que se conoce como señal pasa banda.

Espectro Ensanchado por Secuencia Directa (DSSS – Direct Sequence Spread Spectrum) es una extensión de BPSK u otra modulación de desplazamiento de fase usada por GPS y otros sistemas de navegación por satélite. Como se muestra en la figura 2.2, la señalización DSSS agrega un tercer componente, llamado como una forma de onda *ensanchada* o PRN, la cual es similar a la forma de onda de los datos pero a una tasa de

símbolo mucho más alta. Esta forma de onda PRN es completamente conocida, al menos en los receptores. La forma de onda PRN algunas veces es periódica, y la secuencia finita de bits usada para generar la forma de onda PRN sobre un periodo es llamada como una *secuencia PRN* o *código PRN*. Una excelente visión general de los códigos PRN, incluyendo su generación, características, y familias de código con buenas propiedades son proporcionadas en [16]. El intervalo mínimo de tiempo entre las transiciones en la forma de onda PRN es comúnmente llamado como *periodo de chip*, T_c, la porción de la forma de onda PRN sobre un periodo de chip es conocida como *chip* o *símbolo extendido*, y el recíproco del periodo de chip es conocido como *tasa de chip*, R_c. El parámetro de tiempo independiente para la forma de onda PRN es algunas veces expresado en unidades de chips y llamado como *fase de código*.

La señal que se acaba de describir es llamada *espectro ensanchado*, por el mayor ancho de banda ocupado por la señal después de la modulación por la alta tasa de la forma de onda PRN. En general, el ancho de banda es proporcional a la tasa de chip. Hay tres razones primarias del porqué las formas de onda DSSS son utilizadas para navegación satelital.

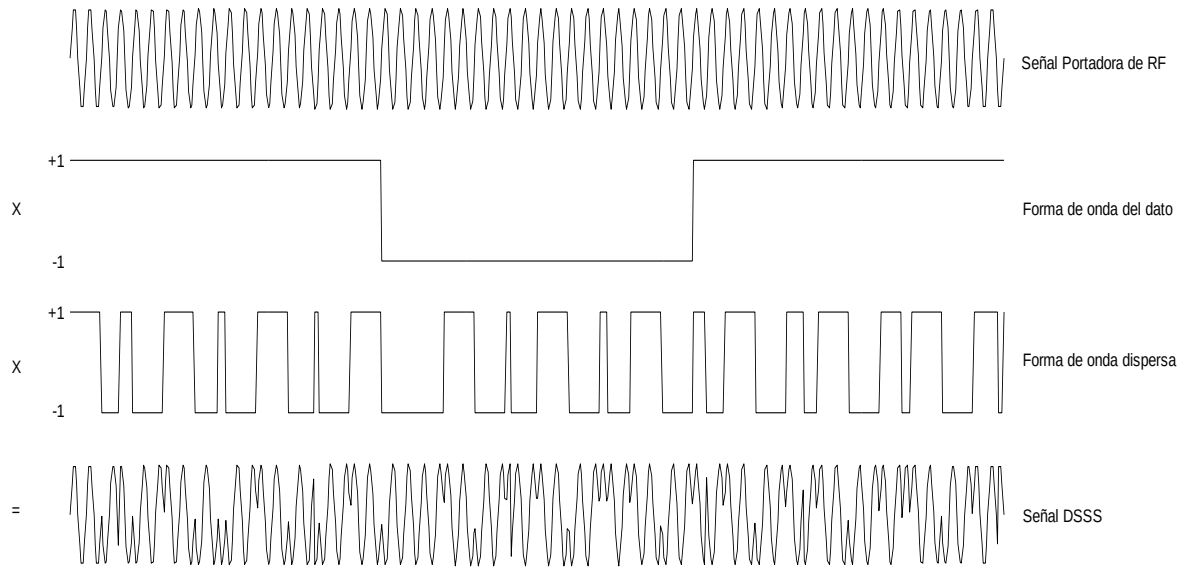

FIGURA 2.2. Modulación DSSS

Primera y las más importante, la frecuente inversión de fases en la señal introducida por la secuencia PRN permite precisar el alcance en el receptor. Segunda, el uso de diferentes secuencias PRN bien diseñadas permite configurar múltiples satélites para transmitir señales simultáneamente y en la misma frecuencia.

Un receptor puede distinguir entre estas señales, en función de sus diferentes códigos. Por esta razón, la transmisión de múltiples señales DSSS con diferentes secuencias PRN sobre una frecuencia portadora en común es llamada como Acceso Múltiple por División de Código (CDMA – *Code Division Multiple Access*). Finalmente, DSSS proporciona un importante rechazo de la interferencia en banda angosta.

2.3 Estructura de la señal GPS.

A continuación se hace una descripción detallada de la generación de señales GPS. La figura 2.3 es un diagrama a bloques describiendo la generación de la señal; [16]. El diagrama a bloques debe ser leído de izquierda a derecha. A la izquierda, la señal de reloj principal es suministrada a los demás bloques. La señal de reloj tiene una frecuencia de 10.23 MHz. Actualmente la frecuencia exacta es 10.22999999543 MHz ajustada para efectos relativistas dando una frecuencia de 10.23 MHz para el usuario. Cuando se multiplica por 154 y 120, se generan las señales portadoras L1 y L2, respectivamente. En la esquina inferior izquierda es usado un limitador para estabilizar la señal de reloj antes de suministrarlo a los generador de código P(Y) y C/A.

En la parte inferior el generador de datos genera los datos de navegación. El generador de códigos y el generador de datos son sincronizados a través de la señal X_1 suministrada por el generador de códigos P(Y). Después de la generación de código, los códigos se combinan con los datos de navegación a través de adiciones de modulo 2.

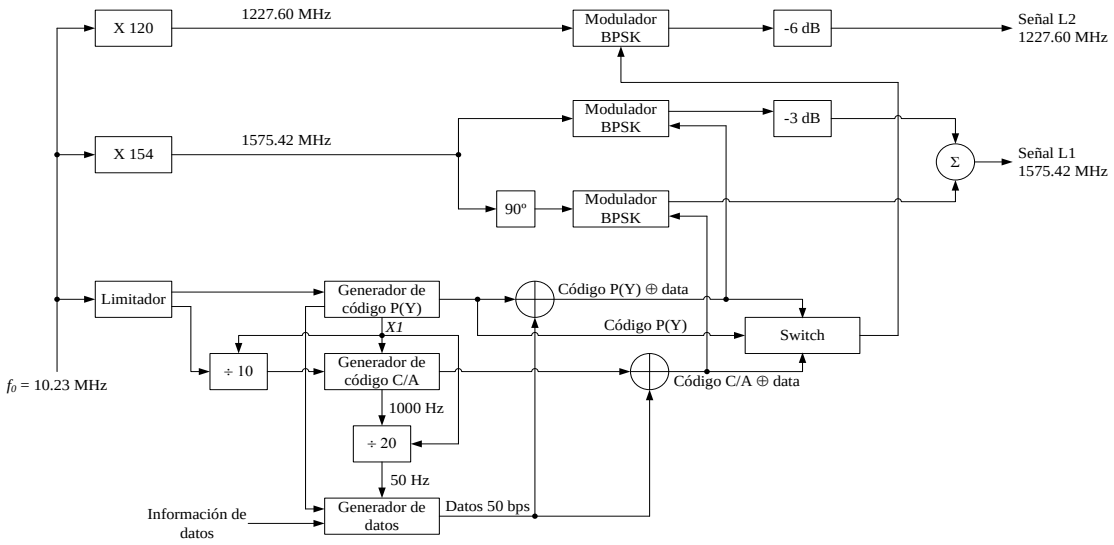

FIGURA 2.3. Generación de señales GPS de los satélites.

La operación OR exclusiva es usada sobre secuencias binarias representadas por 0's y 1's, y sus propiedades son mostradas en la tabla 2.1. Si la secuencia binaria estuvo representada por la representación polar de no retorno a cero, por ejemplo 1's y -1's, la multiplicación ordinaria puede ser usada.

TABLA 2.1

Salida de la operación OR exclusiva

Entrada	Entrada	Salida
0	0	0
0	1	1
1	0	1
1	1	0

TABLA 2.2

Salida de una multiplicación ordinaria

Entrada	Entrada	Salida
-1	-1	1
-1	1	-1
1	-1	-1
1	1	1

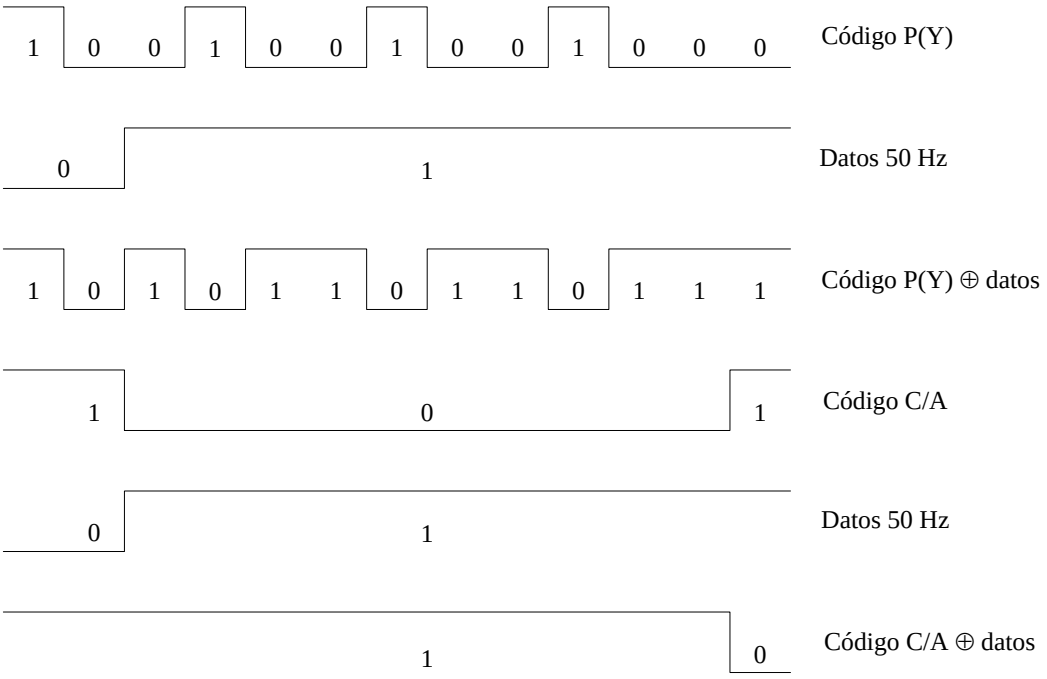

FIGURA 2.4. Código GPS mezclado con los datos.

Las propiedades correspondientes de la multiplicación con dos secuencias binarias de no retorno a cero son mostradas en la tabla 2.2. Las señales *código C/A* ⊕ *datos* y *código P(Y)* ⊕ *datos* mostradas en la figura 2.4 son suministradas a los dos moduladores para la frecuencia L1. Aquí las señales son moduladas con la señal portadora usando el método de Modulación por desplazamiento de fase binaria (BPSK). Notar que los dos códigos son modulados en fase y en cuadratura entre sí en L1. Esto es, hay un cambio de fase de 90º entre los dos códigos, esto se muestra en la figura 2.5. Después la parte P(Y) es atenuada 3 dB, estas dos señales son sumadas para formar la señal L1 resultante. El así llamado SPS se basa solo en las señales de código C/A.

De ello se desprende que la señal transmitida por satélite *k* pueda ser descrita como:

$$s^k(t) = \sqrt{2P_C}\left(C^k(t) \oplus D^k(t)\right)\cos(2\pi f_{L1}t) + \sqrt{2P_{PL1}}\left(P^k(t) \oplus D^k(t)\right)\sin(2\pi f_{L1}t) + \sqrt{2P_{PL2}}\left(P^k(t) \oplus D^k(t)\right)\sin(2\pi f_{L2}t), \quad (2.3)$$

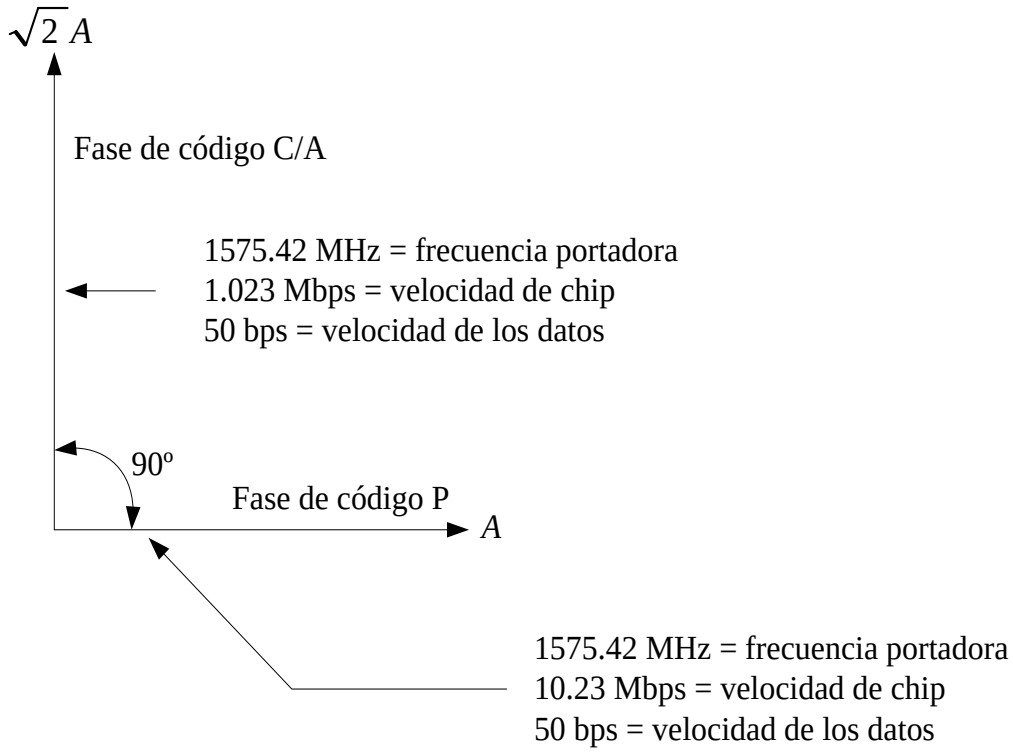

FIGURA 2.5. Estructura de la señal GPS para L1.

donde P_C, P_{PL1} y P_{PL2} son las potencias de las señales con código C/A o P, C^k es la secuencia de código C/A asignado al número de satélite k, P^k es la secuencia de código P asignado al número de satélite k, D^k es la secuencia de datos de navegación, f_{L1}y f_{L2} son las frecuencias portadoras de L1 y L2, respectivamente.

La figura 2.6 muestra las tres partes que forman la señal sobre la frecuencia L1. El código C/A se repite automáticamente cada ms, y un bit de navegación dura 20 ms. De ahí *para cada bit de navegación, la señal contiene 20 códigos C/A completos.* $D(t)$ es el flujo de bits discretos de los datos de navegación.

La figura 2.7 muestra el código Gold C, el dato de navegación D, la señal sumada en modulo dos $C \oplus D$, y la portadora. La señal final es creada por la modulación BPSK donde la portadora instantáneamente hace un cambio de fase de 180º en el momento del cambio de chip. Cuando la transición de un bit del dato de navegación ocurre (alrededor de un tercio del borde derecho) la fase de la señal resultante es también desplazado 180º. El espectro C/A GPS es ilustrado en la figura 2.8.

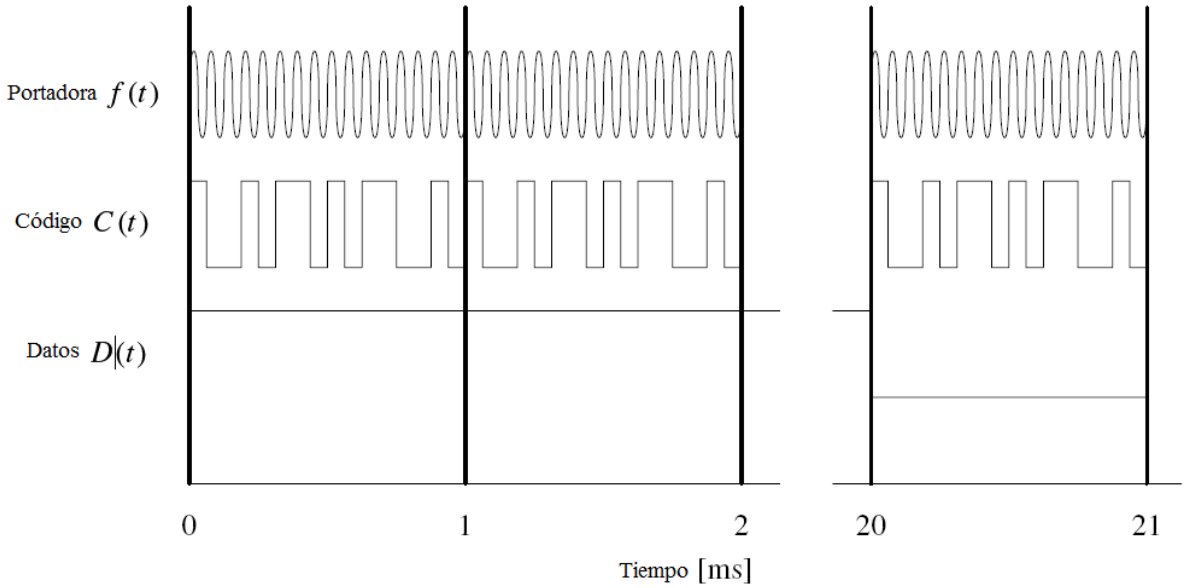

FIGURA 2.6. Estructura general de la señal GPS: $f(t)$ es la onda portadora y $C(t)$ es la secuencia discreta de código C/A.

En resumen: Para GPS la longitud de código es 1023 chips, tiene una tasa de chip de 1.023 MHz (periodo de tiempo de 1 ms), tasa de datos de 50 Hz (20 periodos de código por bit de datos), ~ 90% de la potencia de la señal dentro de ~ 2 MHz de ancho de banda.

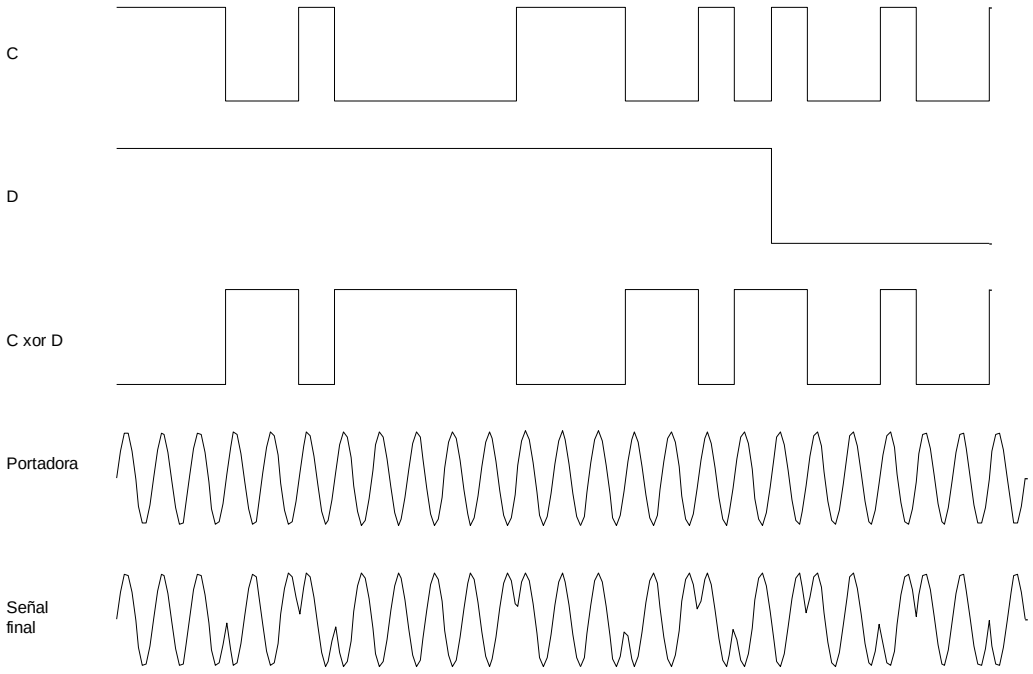

FIGURA 2.7. El efecto de la modulación BPSK de la onda portadora L1 con el código C/A y el dato de navegación para un satélite.

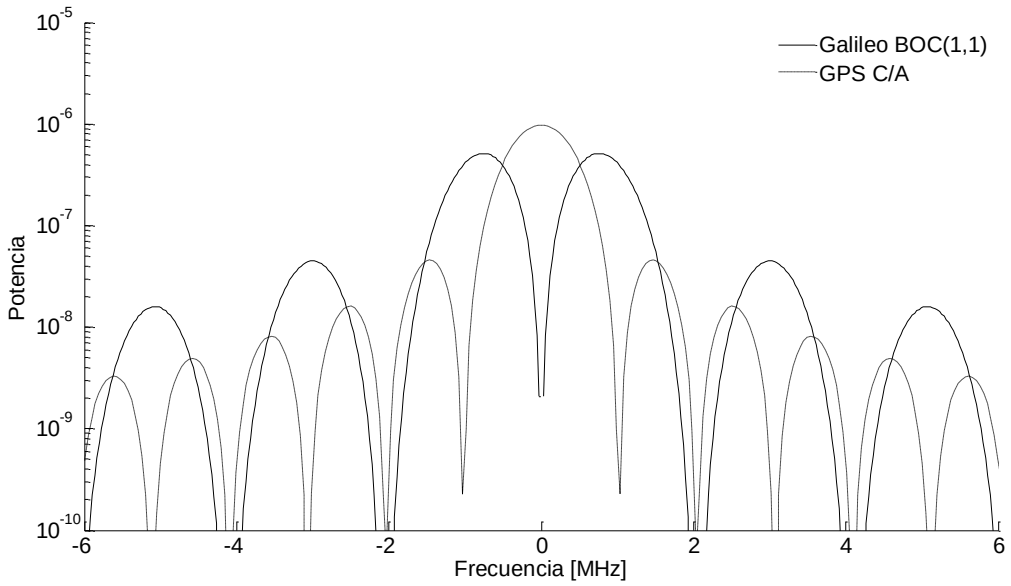

FIGURA 2.8. Espectro GPS C/A L1. Frecuencia central 1575.42 MHz.

2.4 Código C/A.

En esta sección describiremos la secuencia dispersa usada en GPS. Nos limitamos a las secuencias de código C/A. Las secuencias dispersas usadas como códigos C/A en GPS pertenecen a la única familia de secuencias. Son algunas veces llamados como *códigos Gold*, como Robert Gold los describió en 1967; ver [9]. Son también llamadas como secuencias ruido pseudo-aleatorio; o simplemente secuencias PRN, por sus características.

2.4.1 Secuencias Gold.

Los códigos PRN transmitidos por el satélite GPS son secuencias deterministicas; el nombre de PRN (Ruido Pseudo-Aleatorio) proviene de que la señal parece ruido (de hecho una señal aleatoria larga funciona “casi” igual de bien) y lo de “pseudo” es porque no es realmente aleatorio, sino que se obtiene de una formula. Eso es así por una cuestión de pura eficiencia y compacidad. Pero, como digo, una señal aleatoria, en promedio, es casi igual de buena. Cada código C/A es generado usando un Registro de Corrimiento con

Retroalimentación Lineal (LFSR – *Linear Feedback Shift Register*). Este genera una secuencia máxima de longitud $N = 2^n - 1$ elementos.

Un código Gold es la suma de dos secuencias de longitud máxima. El código C/A para GPS usa $n = 10$. La secuencia $p(t)$ se repite cada ms así que la longitud de chip es $1\ \text{ms}/1023 = 977.5\ \text{ns} \approx 1\mu s$, que corresponde a una longitud métrica de 300 m cuando la propagación pasa a través del vacío o aire. Para más detalles sobre la generación de los códigos Gold referirse a la sección 2.4.3. La función de autocorrelación para este código C/A es:

$$r_p(\tau) = \frac{1}{NT_c}\int_0^{NT_c} p(t)p(t+\tau)dt \tag{2.4}$$

En la secuencia se tienen 512 unos y 511 ceros, y estos parecen ser distribuidos aleatoriamente. Hasta ahora la cadena de chips que se genera es del todo deterministica. La secuencia es pseudo-aleatoria, no aleatoria. Fuera del intervalo de correlación la función de autocorrelación de $p(t)$ es $-1/N$. Para el código C/A el término constante es $-1/N = -1/1023$, qué es mostrado en la figura 2.9; en la figura la muestra izquierda tiene un valor de correlación $r_p(0) = 1$; todos los otros valores de correlación son $\frac{63}{1023}$, $\frac{-1}{1023}$, o $\frac{-65}{1023}$. La función de autocorrelación puede ser expresada como la suma de este término constante y unas series infinitas de la función triangulo $r_X(\tau)$. Estas series infinitas son obtenidas por la convolución de $r_X(\tau)$ con unas series infinitas de funciones impulso que son cambiadas en fase por mNT_c:

$$r_P(\tau) = -\frac{1}{N} + \frac{N+1}{N} r_X(\tau) * \sum_{m=-\infty}^{\infty} \delta(\tau + mNT_c) \tag{2.5}$$

Donde $*$ denota la convolución. El espectro de potencia de esta secuencia PRN periódica es derivado de la transformada de Fourier de (2.5):

$$S_P(\omega) = \frac{1}{N^2}\left(\delta(\omega) + \sum_{\substack{m=-\infty \\ m\neq 0}}^{\infty} (N+1)\,\text{sinc}^2\left(\frac{m\pi}{N}\right)\delta\left(\omega + \frac{m2\pi}{NT_c}\right)\right) \tag{2.6}$$

Para $m = \pm1, \pm2, \pm3, \ldots$; ver [16].

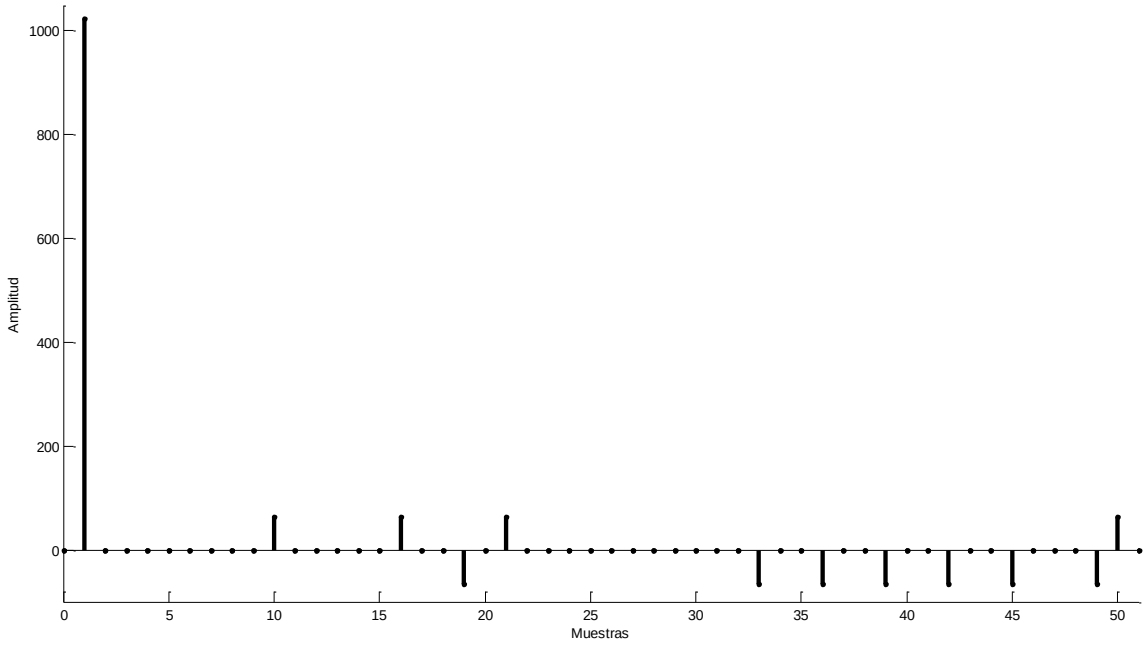

FIGURA 2.9. Gráfica de una función de autocorrelación para una secuencia Gold.

2.4.2 Generación de Secuencias Gold – Una visión general.

La generación de códigos Gold es esquematizado en la figura 2.10. El generador de código C/A contiene dos registros de corrimiento conocidos como G_1 y G_2. Cada uno de estos registros de corrimiento tiene 10 celdas generando secuencias con una longitud de 1023. Las dos secuencias resultantes con longitud de chip 1023 son sumadas en modulo dos para generar un código C/A con una longitud de chip 1023, solo si el polinomio es capaz de generar código de la longitud máxima.

Cada periodo de 1023, los registros de corrimiento son restaurados con todos unos, haciendo al código empezar de nuevo. El registro G_1 siempre tiene una configuración de retroalimentación con el polinomio:

$$f(x) = 1 + x^3 + x^{10} \tag{2.7}$$

significa que el estado tres y el diez son retroalimentados a la entrada. De una manera similar, el registro G_1 tiene el polinomio:

$$f(x) = 1 + x^2 + x^3 + x^6 + x^8 + x^9 + x^{10} \tag{2.8}$$

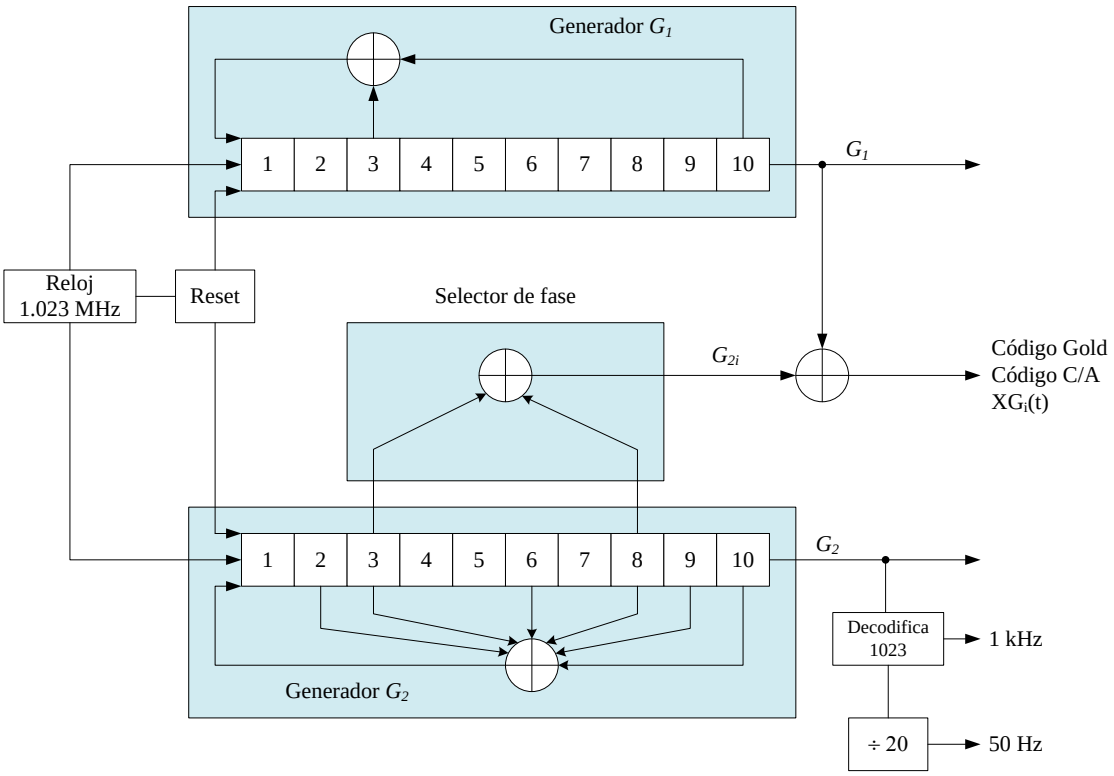

FIGURA 2.10. Generador de código C/A.

Para hacer diferentes códigos C/A para los satélites, la salida de los dos registros de corrimiento son combinadas en una manera muy especial. El registro G_1 siempre proporciona su salida, pero el registro G_2 proporciona dos de sus estados a un sumador de modulo dos para generar su salida. La selección de estados para la suma en modulo dos es llamado selección de fase. La tabla 2.3 muestra la combinación de la sección de fase para cada código C/A. Esta también muestra los primeros 10 chips de cada código en representación octal.

Como la generación de los códigos C/A son de inmensa importancia, bosquejamos en detalle el principio de operación del generador de código C/A en la siguiente sección.

2.4.3 Generación de Secuencias Gold – Detalles.

Un registro de corrimiento es un dispositivo capaz de contractar varios bits de información. Cuando un pulso de reloj es aplicado al registro, el contenido de cada celda se recorre un bit a la derecha. El contenido de la ultima celda es “leída” como salida. Las

propiedades especiales de tal registro de corrimiento dependen de cómo la información es "leída" en la celda 1.

Para un LFSR, la entrada de la celda 1 es determinada por el estado de las otras celdas. Por ejemplo, la suma binaria de las celdas 3 y 10 en un registro de 10 celdas pudiera ser la entrada. Si las celdas 3 y 10 tienen diferentes estados (una es 1 y la otra es 0), un 1 será leído en la celda 1 en el siguiente pulso de reloj. Si las celdas 3 y 10 tienen el mismo estado, 0 será leído en la celda 1. Si comenzamos con 1 en cada celda, después de 12 pulsos de reloj el contenido será 0010001110. El siguiente pulso de reloj tomará el 1 en la celda 3 y el 0 en la celda 10 y plasmaran su suma (1) en la celda 1. Mientras tanto, todos los demás bits se desplazaran a la celda de la derecha, y el 0 en la celda 10 viene a ser el siguiente bit a la salida. Una manera de escribir y denotar este diseño particular es por el modulo dos polinomial $f(x) = 1 + x^3 + x^{10}$. Esta representación polinomial es particularmente útil porque si $1/f(x) = h_0 + h_1x + h_2x^2 + h_3x^3 + \cdots$, entonces los coeficientes $h_0, h_1, h_2, \ldots$ forman la secuencia de salida binaria. El código C/A es generado por dos LFSR de 10 bits con una longitud máxima de $2^{10} - 1$. Uno es el registro $1 + x^3 + x^{10}$ ya descrito y llamado como G_1. El otro $f(x) = 1 + x^2 + x^3 + x^6 + x^8 + x^9 + x^{10}$. Las celdas 2, 3, 6, 8, 9 y 10 son utilizadas y sumadas binariamente para tener la nueva entrada en la celda 1. En este caso, la salida no proviene de la celda 10 pero de un segundo conjunto de intervenciones. Varios pares de estas segundas intervenciones son sumas binarias. Los diferentes pares producen la misma secuencia con diferentes retardos o desplazamientos (tal como dice la propiedad "desplazarse y sumar": una suma chip a chip de una secuencia del registro de longitud máxima y algún desplazamiento de esta es la misma secuencia excepto por un desplazamiento). La versión retardada de la secuencia G_2 es la suma binaria de la salida de la salida de G_1. Que viene a ser el código C/A. Los registros de corrimiento G_1 y G_2 son puestos en el estado de todos unos en sincronismo con la parte temporal del código X_1 usado en la generación del código P. Los diversos pares de intervenciones de G_2 (retardos) son usados para generar el conjunto completo de 36 únicos códigos C/A PRN. Estos son códigos Gold, [9, 7], y dos tienen una muy baja correlación cruzada (son casi ortogonales).

TABLA 2.3. Asignación de la fase de código C/A. La selección de los diferentes estados de la fase de código genera los diferentes códigos C/A para los satélites GPS.

Número de Satélite (ID)	**Número de Señal GPS PRN**	**Selección de fase de código en G_2**	**Retardo de código en chips**	**Primeros 10 chips en octal**
1	1	$2 \oplus 6$	5	1440
2	2	$3 \oplus 7$	6	1620
3	3	$4 \oplus 8$	7	1710
4	4	$5 \oplus 9$	8	1744
5	5	$1 \oplus 9$	17	1133
6	6	$2 \oplus 10$	18	1455
7	7	$1 \oplus 8$	139	1131
8	8	$2 \oplus 9$	140	1454
9	9	$3 \oplus 10$	141	1626
10	10	$2 \oplus 3$	251	1504
11	11	$3 \oplus 4$	252	1642
12	12	$5 \oplus 6$	254	1750
13	13	$6 \oplus 7$	255	1764
14	14	$7 \oplus 8$	256	1772
15	15	$8 \oplus 9$	257	1775
16	16	$9 \oplus 10$	258	1776
17	17	$1 \oplus 4$	469	1156
18	18	$2 \oplus 5$	470	1467
19	19	$3 \oplus 6$	471	1633
20	20	$4 \oplus 7$	472	1715
21	21	$5 \oplus 8$	473	1746
22	22	$6 \oplus 9$	474	1763
23	23	$1 \oplus 3$	509	1063
24	24	$4 \oplus 6$	512	1706
25	25	$5 \oplus 7$	513	1743
26	26	$6 \oplus 8$	514	1761
27	27	$7 \oplus 9$	515	1770
28	28	$8 \oplus 10$	516	1774
29	29	$1 \oplus 6$	859	1127
30	30	$2 \oplus 7$	860	1453
31	31	$3 \oplus 8$	861	1625
32	32	$4 \oplus 9$	862	1712
—	33	$5 \oplus 10$	863	1745
—	34	$4 \oplus 10$	950	1713
—	35	$1 \oplus 7$	947	1134
—	36	$2 \oplus 8$	948	1456
—	37	$4 \oplus 10$	950	1713

Hay actualmente 37 códigos C/A PRN, pero dos de ellos (34 y 37) son idénticos. Un subconjunto de los primeros 32 códigos son asignados a (normalmente 24) satélites y reciclado para cuando un satélite viejo muere y nuevos satélites son lanzados al espacio. Los códigos del 33 al 37 son reservados para otros usos, incluyendo transmisiones terrestres.

La generación de código P sigue el mismo principio que el código C/A, excepto que se usan 4 registros de corrimiento de 12 celdas. Dos registros son combinados para producir el código X_1, que tiene una longitud de 15,345,000 chips y se repite cada 1.5 segundos; y dos registros son combinados para producir el código X_2, que tiene una longitud de 15,345,037 chips. Los códigos X_1 y X_2 pueden ser combinados con 37 diferentes retardos sobre el código X_2 para producir 37 diferentes segmentos de una semana del código P. Cada uno de los primeros 32 segmentos son asociados con un satélite.

2.4.4 Propiedades de correlación.

Los códigos Gold son seleccionados como las secuencias dispersas de las señales GPS debido a sus características. Las características más importantes de los códigos C/A son sus propiedades de correlación. Estas propiedades son descritas ahora.

Las dos propiedades de correlación importantes de los códigos C/A pueden ser expresadas de la siguiente manera:

Correlación casi no cruzada (correlación cruzada mínima). Todos los códigos C/A son casi no correlacionados unos con otros. Esto es, para dos códigos C^i y C^k de los satélites i y k, la correlación cruzada puede ser escrita como:

$$r_{ik}(m) = \sum_{i=0}^{1022} C^i(l)C^k(l+m) \approx 0 \qquad \text{para toda } m \tag{2.9}$$

Casi ninguna correlación excepto para el desfase cero. Todos los C/A son casi no correlacionados con ellos mismos, excepto para el desfase cero. Esta propiedad hace esto

fácil para descubrir cuando dos códigos similares están alineados perfectamente. La propiedad de autocorrelación para el satélite k puede ser escrita como:

$$r_{kk}(m) = \sum_{i=0}^{1022} C^k(l)C^k(l+m) \approx 0 \qquad \text{para } |m| \geq 1 \tag{2.10}$$

La figura 2.11 muestra un ejemplo de las propiedades de autocorrelación y correlación cruzada del código C/A. Como se esperaba, la figura muestra una alta correlación en el desfase cero cuando se correlaciona con el mismo código C/A, y baja correlación cuando se correlaciona con otro código C/A.

La autocorrelación mostrada en parte izquierda de la figura 2.11 tiene un pico de magnitud:

$$r_{kk,pico} = 2^n - 1 = 1023 \tag{2.11}$$

donde *n* es el número de estados en el registro de corrimiento. En este caso, *n* es igual a 10. El valor restante satisface la siguiente desigualdad, [9]:

$$|r_{kk}| \leq 2^{(n+2)/2} + 1 \tag{2.12}$$

para $n = 10$, tenemos:

$$|r_{kk}| \leq 65 \tag{2.13}$$

La correlación cruzada en la parte derecha de la figura 2.11 también satisface la desigualdad de la ecuación (2.12).

2.5 Cambios en la Frecuencia Doppler.

En GPS nos enfrentamos a un problema con un cambio en la frecuencia Doppler causado por el movimiento del transmisor (satélite) en relación con el receptor GPS.

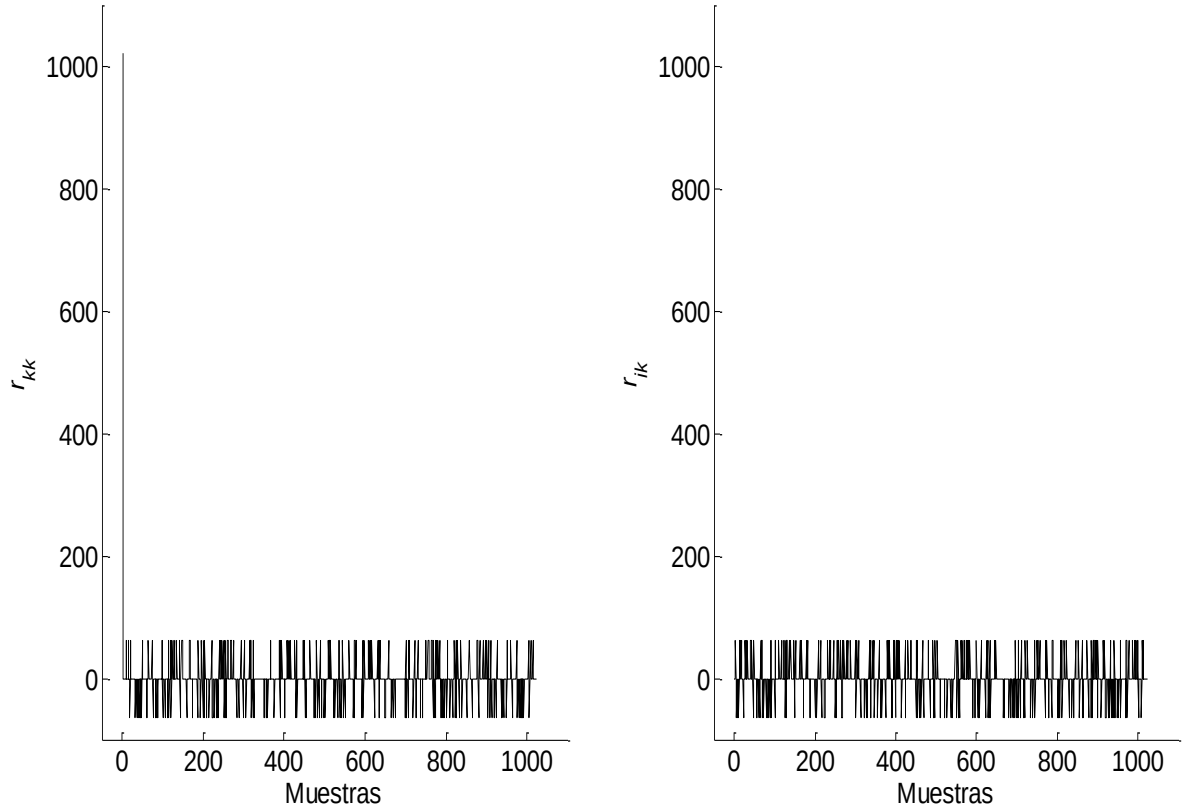

FIGURA 2.11. Propiedades de correlación de los códigos C/A. Derecha: Autocorrelación $r_{kk}(n)$ del código C/A para el PRN 1. Izquierda: Correlación cruzada $r_{ik}(n)$ de los códigos C/A para los PRN 1 y 2.

El cambio en la frecuencia Doppler afecta tanto a la adquisición como al seguimiento de la señal GPS. Para un receptor GPS estacionario el máximo cambio de frecuencia Doppler para la frecuencia L1 es alrededor de ±5 kHz y para un receptor GPS moviéndose a alta velocidad es razonable suponer que el máximo efecto Doppler es de ±10 kHz.

El cambio de frecuencia Doppler sobre el código C/A es debido a la baja tasa de chip del código C/A. El código C/A tiene una tasa de chip de 1.023 MHz, qué es $1575.42/1.023 = 1540$ veces más pequeña que la frecuencia portadora L1. De ello se deduce que la *frecuencia Doppler sobre el código C/A es 3.2 Hz para un receptor GPS sin movimiento y 6.4 Hz para en receptor GPS en movimiento a alta velocidad.*

La frecuencia Doppler sobre el código C/A puede causar desalineamiento entre los códigos recibidos y generados localmente y los valores de la frecuencia Doppler son importantes para el método de seguimiento.

2.6 Datos de navegación.

Los datos de navegación son transmitidos sobre la frecuencia L1 con una velocidad de 50 bps. Esta sección describe la estructura y contenido de los datos de navegación. La figura 2.12 muestra la estructura global de un mensaje de navegación completo en donde cada subtrama contiene 300 bits que dura 6 segundos. Las subtramas 1, 2 y 3 se repiten cada 30 segundos mientras las subtramas 4 y 5 tienen 25 versiones antes de repetir. Es decir, todo el mensaje de navegación se repite cada 12.5 minutos.

El formato básico de los datos de navegación es una trama con longitud de 1500 bits conteniendo 5 subtramas, cada una teniendo una longitud de 300 bits. Una subtrama contiene 10 palabras, cada palabra tiene una longitud de 30 bits. La subtrama 1, 2 y 3 se repiten en cada trama. Las últimas subtramas, 4 y 5, tienen 25 versiones (con la misma estructura, pero con diferentes datos) llamadas como pagina 1 a 25.

Con la tasa de bit de 50 bps, la transmisión de una subtrama dura 6 segundos, una trama dura 30 segundos, y un mensaje de navegación completo dura 12.5 minutos.

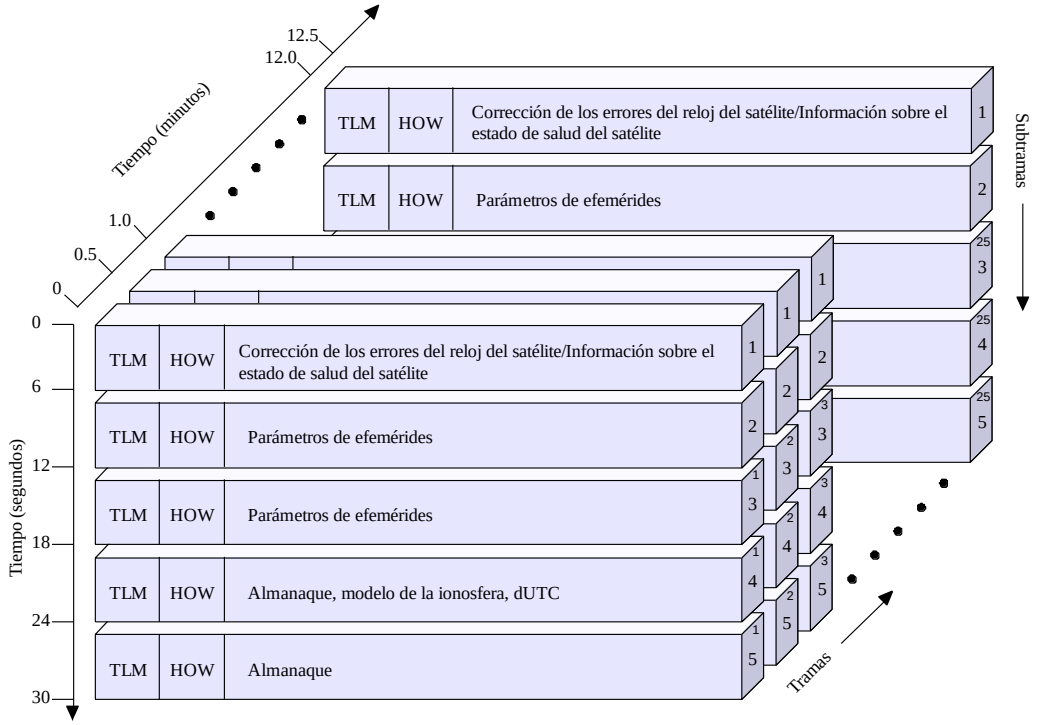

FIGURA 2.12. Estructura de los datos de navegación GPS.

2.6.1 Palabras Telemetría y Handover.

Las subtramas de 10 palabras siempre empiezan con dos palabras especiales, el par de palabras telemetría (TLM) y handover (HOW).

TLM es la primera palabra de cada subtrama y esta se repite cada 6 segundos. Esta contiene un preámbulo de 8 bits seguido por 16 bits reservados y paridad. El preámbulo debe ser usado para sincronización de la trama.

HOW contiene una versión truncada de 17 bits de Tiempo de Semana (TOW – *Time of Week*), seguida por dos banderas que proporcionan información al usuario de la lucha contra la suplantación de identidad, etc. Los siguientes tres bits indican el identificador (ID) de la subtrama que muestran en cuál de las cinco subtramas de la trama actual la palabra HOW está situada.

2.6.2 Datos en el Mensaje de Navegación.

Además de las palabras TLM y HOW, cada subtrama contiene ocho palabras de datos. Esto solo será una somera descripción de los datos en las diferentes palabras y no una descripción completa de todos los bits.

Subtrama 1 – Reloj y datos del estado de salud del satélite: La primera subtrama contiene en primer lugar la información de reloj. Esta información es necesaria para calcular en qué momento el mensaje de navegación es transmitido por el satélite. Además, la subtrama 1 contiene datos de salud del satélite indicando sí o no los datos deben ser de confianza.

Subtrama 2 y 3 – Datos de Efemérides del Satélite: Las subtramas 2 y 3 contienen el dato de efemérides del satélite. El dato de efemérides hace referencia a la órbita del satélite y se requiere para calcular la posición del satélite.

Subtramas 4 y 5 – Soporte de Datos: Como se ha mencionado, las dos últimas subtramas se repiten cada 12.5 minutos, dando un total de 50 subtramas. Las subtramas 4 y 5 contienen los datos de almanaque. Los datos de almanaque son el dato de efemérides y de reloj con una precisión reducida. Además, cada satélite transmite el

dato de almanaque de todos los satélites GPS mientras que sólo transmite los datos de efemérides para sí mismo. El resto de las subtramas 4 y 5 contienen varios datos; parámetros UTC, indicadores de salud, y parámetros de la ionosfera.

Para una descripción más profunda del contenido de los datos de navegación, ver [25].

Capítulo 3

Estructura básica del receptor GPS

En este capítulo se describirá la estructura básica de un receptor GPS. La estructura descrita es la más comúnmente usada. La descripción seguirá el diagrama a bloques de la figura 3.1, comenzando con la antena del receptor. La señal llega a la antena, y la señal de radio frecuencia (RF) es filtrada y es convertida a una frecuencia intermedia (IF) por la terminal de entrada de RF. Después de la conversión de bajada, la señal analógica es muestreada por un convertidor A/D proporcionando una señal digital a las partes restantes del receptor. El procesamiento de la señal es llevado a cabo en canales – uno para cada señal seguida. El último paso en el receptor GPS es aplicar los algoritmos para el cálculo de posición.

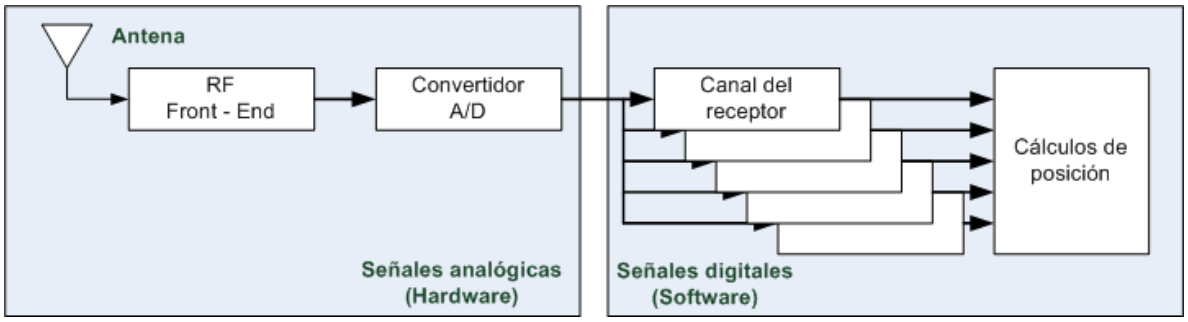

FIGURA 3.1. Estructura básica de un receptor GPS.

3.1 Componentes de la terminal de entrada para L1 de un GNSS.

Una terminal de entrada GNSS L1 completamente funcional es descrita en la figura 3.2. En las secciones siguientes, se discutirá cada uno de los elementos dentro de la figura usando esta implementación como un caso de estudio, [11]. La terminal de entrada RF es el bloque de debe preparar la señal analógica para muestrearla por el convertidor A/D. La terminal de entrada mostrada en la figura 3.2 contiene dos funcionalidades básicas, acondicionamiento de la señal y conversión de bajada.

3.1.1 Antena GNSS.

Una antena GPS es diseñada particularmente para su uso en GPS. Es decir, los requisitos de la antena se determinan a partir del uso en el sistema. Para mayor detalles sobre el diseño de antenas ver [2]. Los requisitos se definen a partir de los siguientes parámetros:

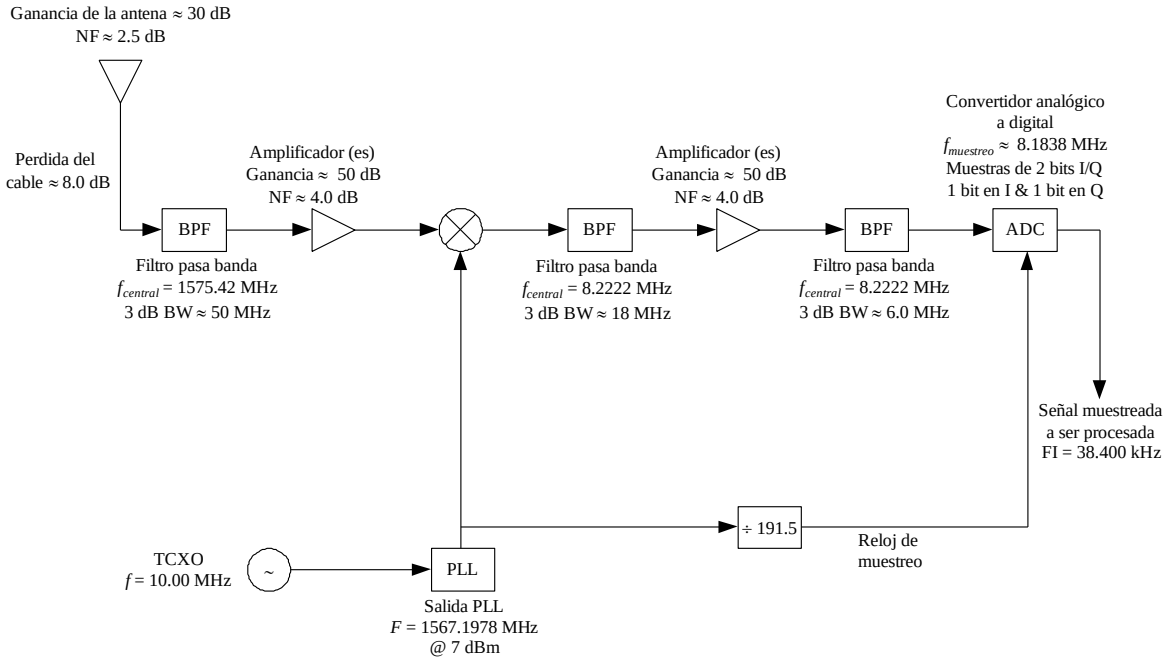

FIGURA 3.2. Terminal de entrada GNSS L1.

Ganancia vs. Elevación: La antena debe tener alta ganancia para ángulos por encima de un específico ángulo de elevación. De este modo las señales procedentes de los satélites serán amplificadas comparadas con las señales procedentes de las fuentes con un ángulo de elevación inferior.

Rechazo a la interferencia: La antena debe rechazar un cierto nivel de interferencia en las señales. Esto podría, por ejemplo, originarse de varios transmisores de radio situados en las cercanías. La antena debe en primer lugar, rechazar todas las señales de diferentes bandas de frecuencia.

Rechazo a la multitrayectoria (multipath): La antena debe rechazar la señales multitrayectoria lo más eficaz posible ya que estas señales son muy difícil de eliminar en una etapa posterior del receptor. La necesidad del rechazo a la multitrayectoria se ajusta bastante bien con las propiedades de ganancia vs. elevación, cuando las señales multitrayectoria tienden a llegar por reflexiones de la tierra por debajo de la antena.

Propiedades físicas: La antena debe ser diseñada para encajar en el ambiente que se supone será usada. Esto define el tamaño, forma, y material de la antena GPS.

3.1.2 Acondicionamiento de la señal.

El principal objetivo del bloque de acondicionamiento de la señal es quitar los componentes de interferencia en la señal. El método para quitar otras señales involucra un filtro pasa banda. La figura 3.3 muestra las frecuencias GPS entre otras señales de radio presentes. El filtro debe de remover todos esos componentes de la señal y en el caso de una sola frecuencia recibir también la frecuencia GPS L2.

La otra funcionalidad importante del bloque de acondicionamiento de la señal es de amplificar la señal recibida. La señal recibida por la antena tiene muy baja amplitud. Esto es el resultado de una combinación de una potencia de transmisión baja del satélite y el largo camino que viaja la señal. La amplificación de la señal mejora el resultado de las partes restantes del receptor.

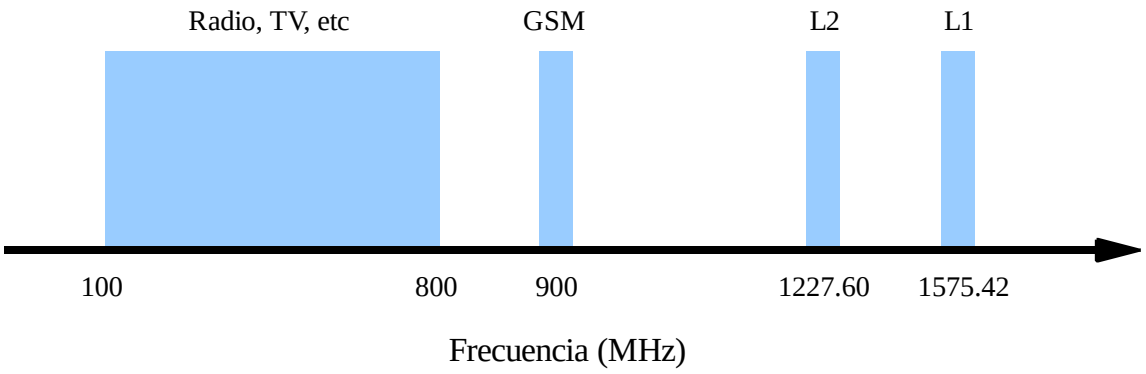

FIGURA 3.3. Las señales GPS entre otras señales de radio en la circundante banda de frecuencias.

3.1.3 Conversión de bajada.

El propósito del bloque de conversión de bajada es como su nombre lo dice bajar a una frecuencia más pequeña la señal proporcionada por el bloque de acondicionamiento de la señal. La señal recibida es convertida de una frecuencia de 1575.42 MHz a una frecuencia intermedia (IF) que es regularmente de unos pocos MHz. La conversión de bajada es llevada a cabo por un mezclador, mezclando la señal de entrada con la señal de un oscilador local (LO). Él LO debe ser muy estable y exacto, por lo que la nueva IF es bien definida. La función del mezclador puede ser calculada como:

$$s_{out}(t) = s_{in}(t) \cdot s_{LO}(t) \tag{3.1}$$

Aplicando la transformada de Fourier a la ecuación (3.1) tenemos:

$$S_{out}(f) = S_{in}(f) * S_{LO}(f) \tag{3.2}$$

donde $S(f)$ es la transformada de Fourier de $s(t)$. Esto es, dos señales multiplicadas en el dominio del tiempo corresponden a la convolución en el dominio de la frecuencia. La figura 3.4 muestra el principio básico de conversión de bajada de la señal RF a una frecuencia intermedia. Como se puede ver en la figura 3.4, la combinación de la señal de RF y la señal del LO corresponden a la convolución de la transformada de Fourier de las dos señales. La frecuencia del LO es seleccionada de la siguiente manera:

$$f_{LO} = f_{RF} - f_{IF} \tag{3.3}$$

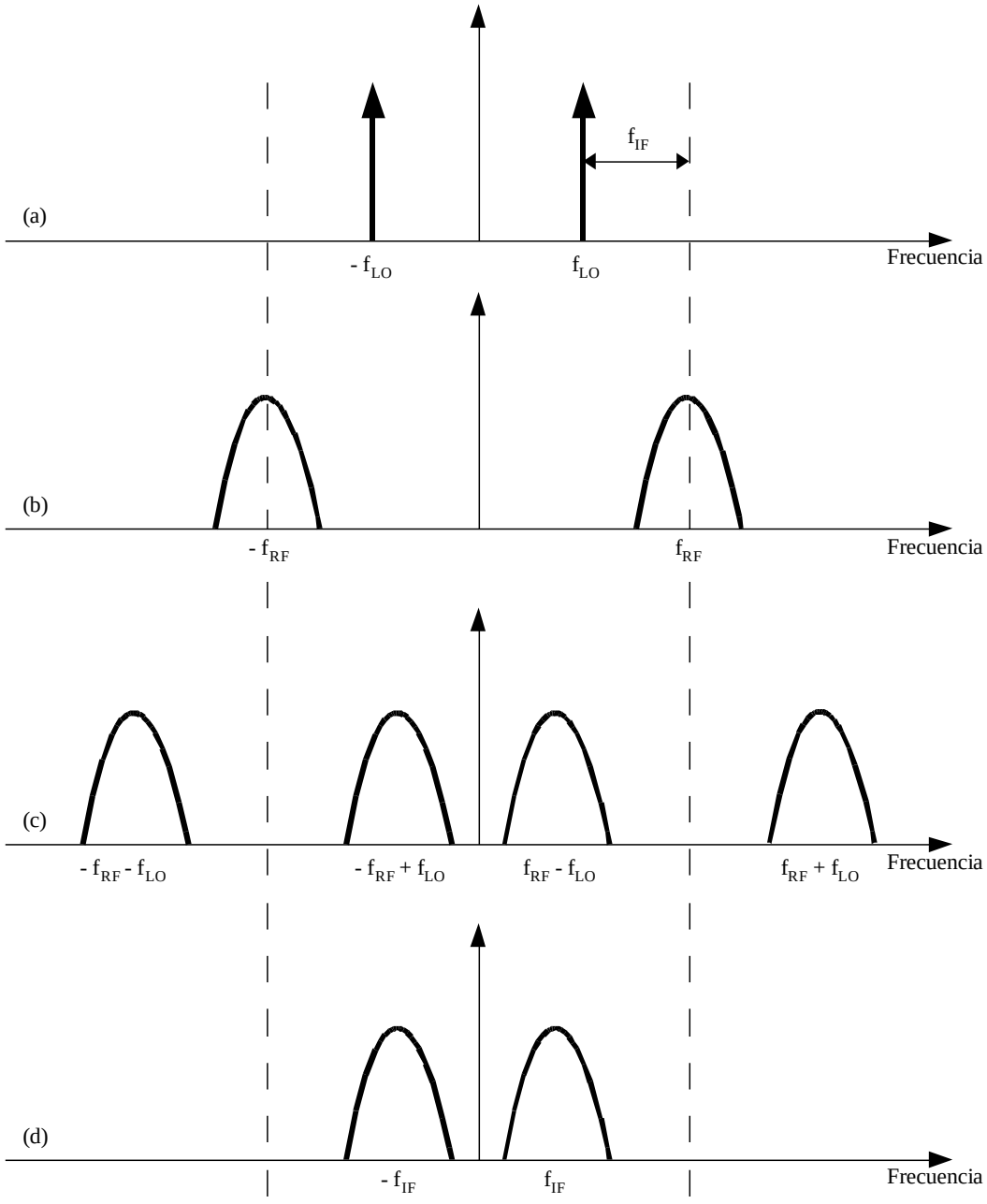

FIGURA 3.4. Conversión de bajada de la señal de RF a una señal con IF. (a) Transformada de Fourier de la señal LO. (b) Transformada de Fourier de la señal de entrada de RF. (c) Resultado de la combinación de la señal del LO y la señal de RF. Esto corresponde a una convolución en frecuencia de la señal (a) y (b). (d) La señal resultante de IF después del filtro pasa bajas.

La mezcla de resultados de cuatro componentes de la señal ubicadas en $\{-f_{RF}-f_{LO}, -f_{RF}+f_{LO}, f_{RF}-f_{LO}, f_{RF}+f_{LO}\}$. Solo los componentes de baja frecuencia se quedaran, por lo tanto el filtro pasa bajas debe ser implementado para remover las dos restantes.

La última señal mostrada en la figura 3.4 representa la señal resultante de IF después de la conversión de bajada y el filtrado pasa bajas. Esto demuestra la idea de la conversión de bajada y también la necesidad de usar un filtro pasa bajas para remover dos componentes de frecuencia no deseados después de la conversión de bajada.

3.1.4 Imagen de frecuencias.

Otro efecto secundario de la conversión de bajada es la imagen de frecuencias. La imagen de frecuencias son componentes de frecuencia que están ubicados en la misma banda de frecuencia que la señal deseada después de la conversión de bajada. Esto tiene que ser considerado para evitar ruido en la señal resultante. A continuación se mostrara como aparece la imagen de frecuencias, y como debe ser evitado.

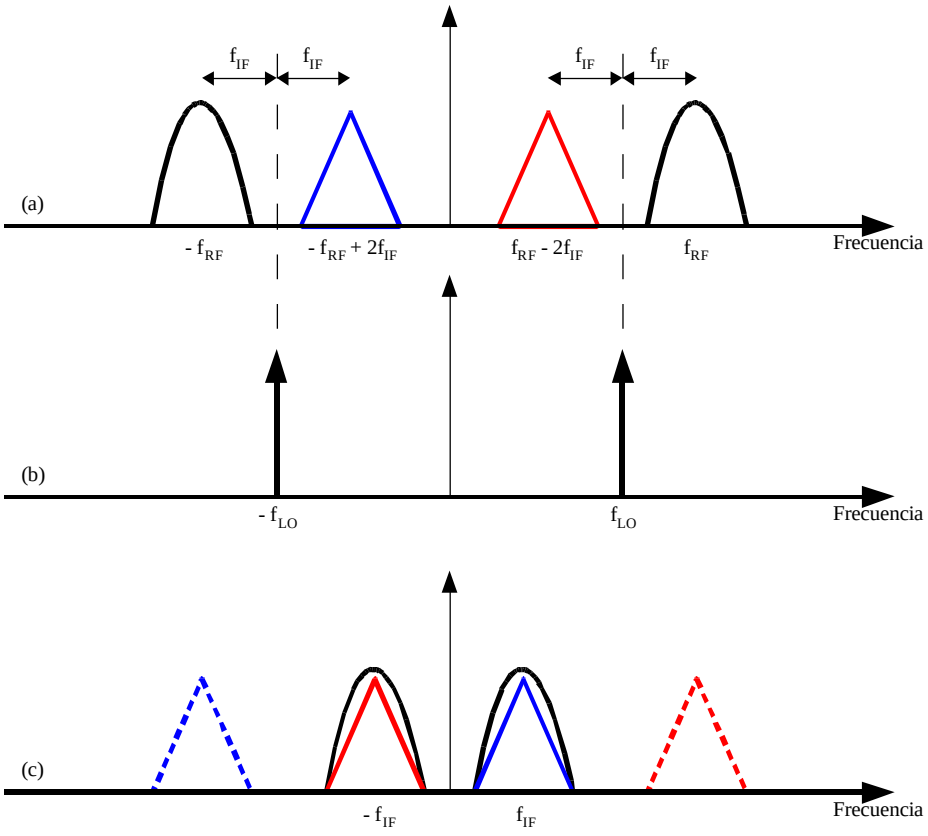

FIGURA 3.5. La imagen de frecuencias es resultado de la conversión de bajada. (a) El triángulo azul y rojo indican la imagen en las bandas de frecuencia. (b) La señal LO usada para la conversión de bajada. (c) Las áreas de la imagen de frecuencias después de la conversión de bajada están ahora ubicadas en la misma banda de frecuencia que la señal GPS.

La figura 3.5(a) muestra la señal GPS y la señal en la banda de imagen de frecuencias representada por los triángulos azul y rojo. Estas bandas están ubicadas en $-f_{RF} + 2f_{IF}$ y $f_{RF} - 2f_{IF}$, con una distancia de $2f_{IF}$ a la frecuencia de la señal GPS. La figura 3.5 (b) muestra la representación de la frecuencia de la señal LO. La figura 3.5(c) muestra el resultado de la conversión de bajada. La parte de frecuencia más alta de la señal GPS no es mostrada aquí debido a que fue removida por el filtro pasa bajas. Los triángulos punteados muestran las dos partes de la imagen de frecuencias que esta ubicadas exactamente en la parte superior de las señales GPS. Para evitar esto, la imagen de frecuencias tiene que ser removida por filtros antes de realizar la conversión de bajada. Estos filtros tienen que atenuar estas frecuencias significativamente, ya que las señales GPS recibidas son muy débiles comparadas con la mayoría de las otras señales.

3.1.5 Convertidor Analógico a Digital (ADC).

El convertidor A/D es responsable de muestrear la señal analógica de conversión de bajada. Para evitar el aliasing de la señal cuando es muestreada una señal sobre una onda portadora, la tasa de muestreo debe ser al menos:

$$f_s = 2 \cdot \Delta f \tag{3.4}$$

donde Δf es el ancho de banda de la señal. Sin embargo, cuando la onda portadora es de interés, la señal deberá ser muestreada como mínimo con una tasa de muestreo de:

$$f_s = 2 \cdot f_{max} \tag{3.5}$$

donde f_{max} es el máximo componente de frecuencia de la señal. En un receptor GPS L1, el ancho de banda de la señal de entrada es aproximadamente:

$$\Delta f \approx 2 \text{ MHz} \tag{3.6}$$

Cuando una señal es convertida a una frecuencia IF arbitraria, el máximo componente de frecuencia es:

$$f_{max} = f_{IF} + \frac{1}{2}\Delta f$$

$$f_{max} = f_{IF} + 1\ \text{MHz} \tag{3.7}$$

El número de bits para la representación digital de la señal muy a menudo se pone a 1 bit. El número pequeño de bits simplifica los cálculos en el receptor GPS más tarde.

Refiriéndonos al diseño de la figura 3.2, se usa una frecuencia de muestreo de 8.1838 MHz para la frecuencia intermedia de 1248.068 MHz, y esta proporciona una conversión a una frecuencia intermedia final de 38.400 kHz.

3.2 Resultado de los datos muestreados.

Ahora que la operación y funcionalidad de la terminal de entrada GNSS ha sido descrita, vale la pena realzar los datos resultantes que han sido coleccionados por el diseño de la terminal de entrada descrito en la figura 3.2. Los parámetros importantes para el procesamiento de señales son:

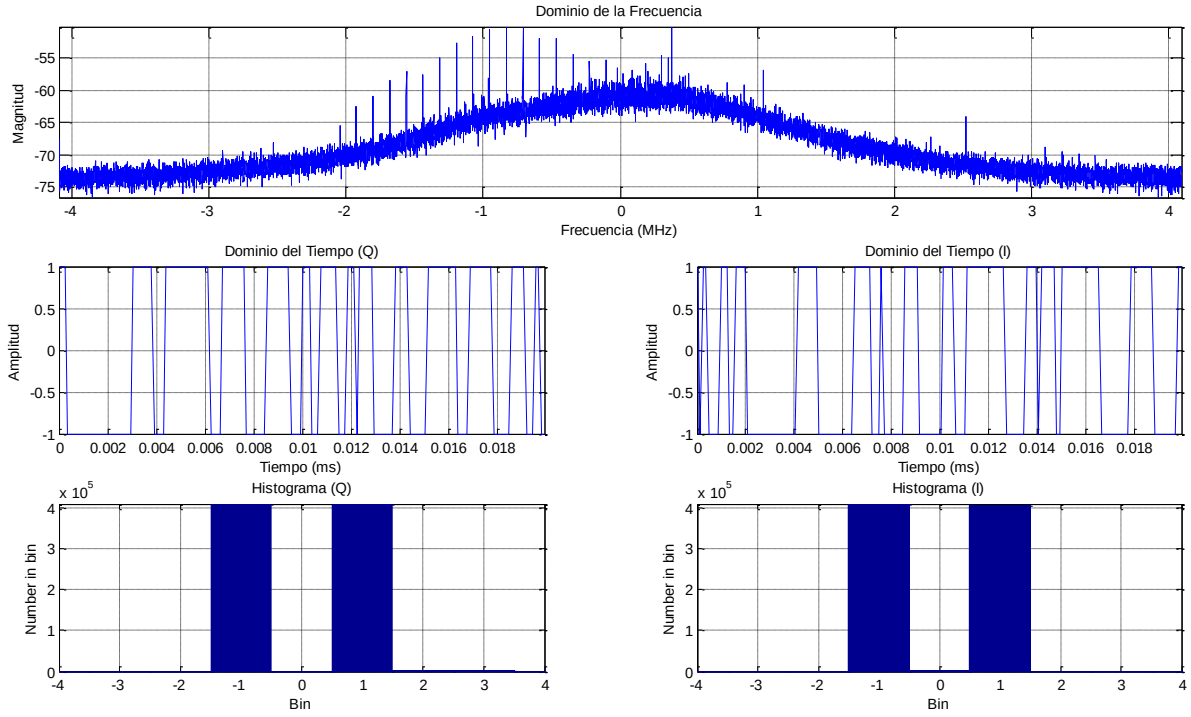

FIGURA 3.6. Datos obtenidos a la salida de la terminal de entrada simulados en Matlab.

- Frecuencia de muestreo: 8.1838 MHz
- Frecuencia intermedia: 38.400 kHz
- Muestras de 2 bits en I/Q. Un bit para I y un bit para Q.

Los parámetros anteriores proporcionan toda la información necesaria para la operación de los algoritmos de procesamiento de la señal. Algunos otros puntos, tales como el tiempo y fecha y lugar aproximado de la colección de los datos, en el algoritmo de adquisición se discuten estos puntos, pero no son requeridos.

La figura 3.6 muestra la descripción en el dominio del tiempo, histograma y en el dominio de la frecuencia de los datos obtenidos por la terminal de entrada.

3.3 Canales del receptor.

El procesamiento de la señal GPS toma lugar en diferentes canales. Cada satélite visible por la antena se destina a su propio canal, limitado por un número máximo de canales en el receptor [4]. La figura 3.7 presenta un panorama general de un canal.

Antes de la asignación de un canal a un satélite, el receptor debe saber que satélites son actualmente visibles. Hay dos maneras comunes de encontrar los satélites visibles inicialmente. Una es conocida como *arranque en caliente (warm start)* y la otra es conocida como *arranque en frio (cold start)*.

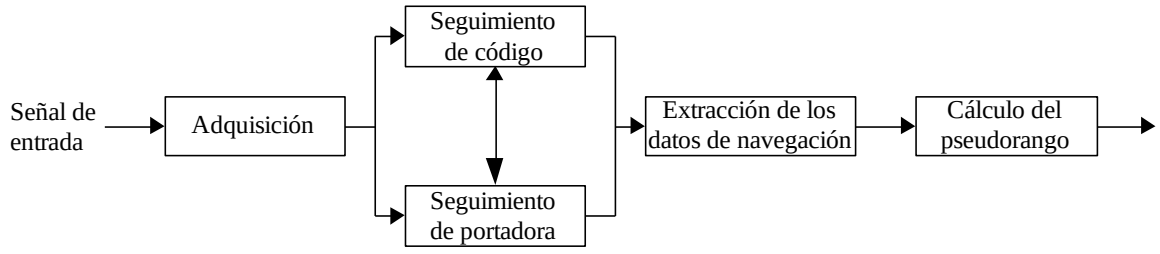

FIGURA 3.7. Un canal del receptor.

Arranque en caliente: En el arranque en caliente, el receptor combina la información almacenada en el dato de almanaque y la última posición calculada por el receptor. El dato de almanaque es usado para calcular las posiciones aproximadas (poco refinadas) de todos los satélites al momento. Estas posiciones son entonces combinadas con la posición del receptor en un algoritmo calculando que satélites deben ser visibles. El arranque en caliente tiene algunas desventajas. Si el receptor se ha movido lejos desde que este fue apagado (por ejemplo, de un continente a otro), la posición del receptor no puede ser confiable y los satélites encontrados no corresponden con los satélites realmente visibles. Otro caso es que el dato de almanaque puede ser obsoleto, por lo que no puede proporcionar buenas posiciones de los satélites. En cualquiera de estos casos, el receptor tiene que hacer un arranque en frio.

Arranque en frio: En un arranque en frio, el receptor no se basa en alguna información almacenada. En lugar de ello, comienza desde cero la búsqueda de los satélites. El método de búsqueda es conocido como adquisición y este se describe en la siguiente sección.

3.3.1 Adquisición.

El objetivo de la adquisición es en primer lugar identificar si un satélite es visible para el usuario. Si el satélite es visible, la adquisición determinara las siguientes dos propiedades de la señal:

Frecuencia: La frecuencia de la señal de un satélite puede diferir de su valor nominal. En el caso de conversión de bajada, la frecuencia nominal de la señal GPS en L1 corresponde a la IF. Sin embargo, las señales se ven afectadas por el movimiento relativo del satélite causando un efecto Doppler. La frecuencia Doppler puede cambiar acercándose a valores tan altos como 10 kHz en caso de una velocidad máxima del satélite combinada con una muy alta velocidad del usuario [26]. Para un receptor estacionario, el cambio de la frecuencia Doppler nunca será superior a 5 kHz.

Fase del código: La fase del código denota el punto en el bloque de datos actual donde comienza el código C/A. Si un bloque de datos de 1 ms es examinado, el dato incluye todo un código C/A y por lo tanto el principio del código C/A.

Muchos métodos diferentes han sido usados, pero todos ellos están de una manera u otra basados sobre las propiedades de la señal GPS. Especialmente las propiedades de correlación del código C/A son las más importantes (ver sección 2.4.4).

La señal recibida s es una combinación de señales de todos los n satelites visibles:

$$s = s^1(t) + s^2(t) + \cdots + s^n(t) \tag{3.8}$$

Cuando es adquirido un satélite k, la señal entrante s es multiplicada con el código C/A generado localmente correspondiente al satélite k. La correlación cruzada entre códigos C/A para diferentes satélites implica que las señales de otros satélites son casi eliminadas por este procedimiento. Para evitar eliminar el componente de la señal deseada, el código C/A generado localmente debe ser correctamente alineado en tiempo, esto se hace con la fase del código correcta.

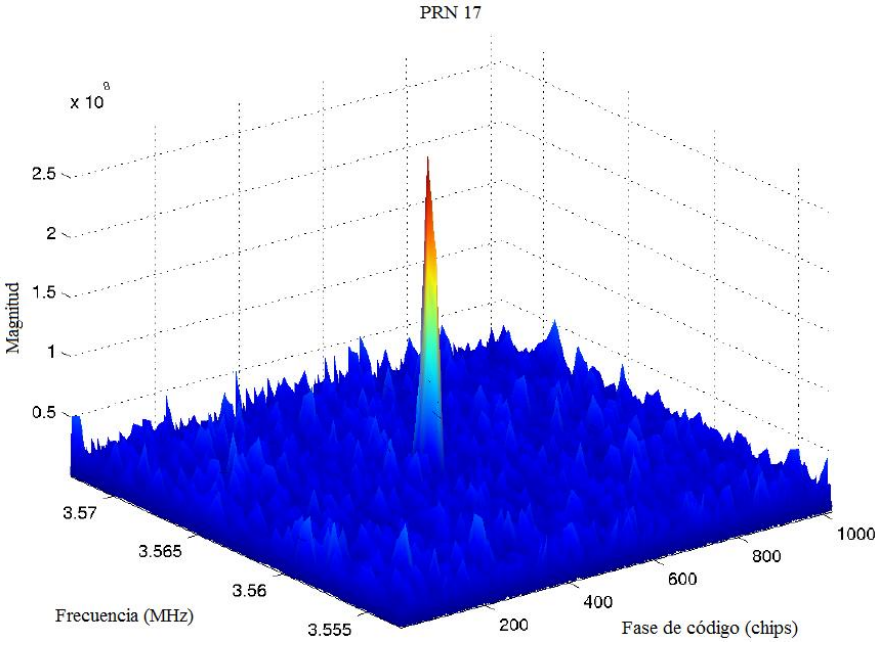

FIGURA 3.8. Gráfica de adquisición para el satélite 17.

Después de la multiplicación con el código generado localmente, la señal será mezclada con una onda portadora generada localmente. Esto es hecho para eliminar la onda portadora de la señal recibida. Para eliminar la onda portadora de la señal, la frecuencia de la señal generada localmente debe ser lo más cercana posible a la frecuencia de la señal portadora. Como mencionamos anteriormente, la frecuencia puede cambiar hasta ±10 kHz de la frecuencia nominal, entonces diferentes frecuencias dentro de esta área deben ser probadas. Para identificar sí o no un satélite es visible, es suficiente buscar la frecuencia en saltos de 500 Hz resultando en 41 frecuencias diferentes en caso de un movimiento rápido del receptor y en 21 en caso de un receptor estático [1].

Después de la mezcla con la onda portadora generada localmente, todos los componentes de la señal son elevados al cuadrado y sumados proporcionando un valor numérico. El proceso de adquisición trabaja como un procedimiento de búsqueda. Para cada una de las diferentes frecuencias 1023 diferentes fases de código son probadas. Cuando todas las posibilidades para la fase del código y la frecuencia han sido probadas, se realiza una búsqueda para el valor máximo. Si el valor máximo excede un determinado umbral, el satélite es adquirido con la correspondiente frecuencia y cambio de fase. La figura 3.8 muestra un pico significante, que indica una alta correlación.

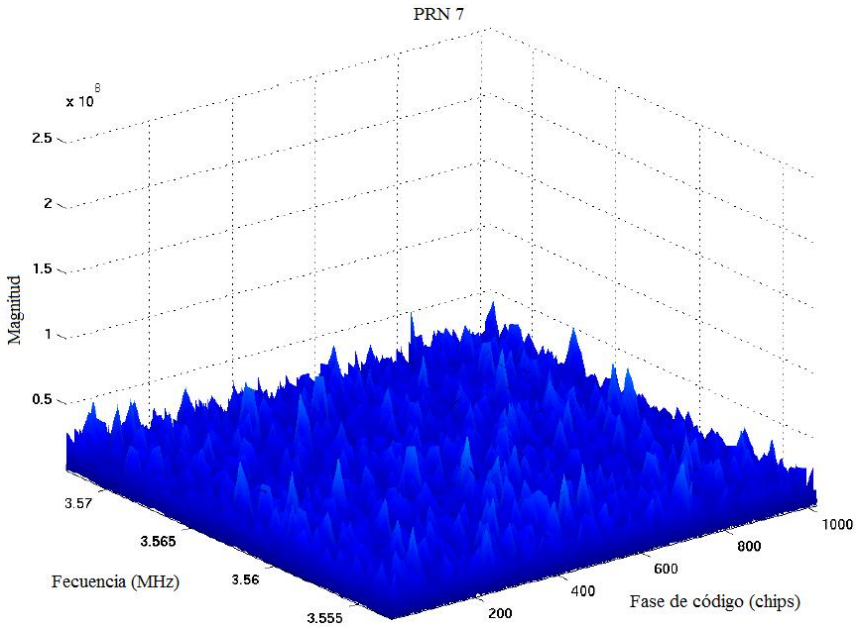

FIGURA 3.9. Gráfica de adquisición para el satélite 7.

La figura 3.9 muestra una gráfica de adquisición típica, realizada por un satélite que no está visible actualmente por el receptor GPS. En esta gráfica, todos los valores son casi idénticos indicando una baja correlación.

3.3.2 Seguimiento.

El principal objetivo del seguimiento es refinar los valores "ásperos" de la fase del código y la frecuencia, y mantener la pista de éstos ya que las propiedades de la señal cambian con el tiempo. La precisión del valor final de código es relacionado con la precisión del cálculo del pseudorango más adelante.

Seguimiento de código: El seguimiento de código es muy a menudo implementado con un lazo de seguimiento de retraso (DLL – *Delay Lock Loop*) donde tres códigos locales (réplicas) son generados y correlacionados con la señal entrante. Estas tres réplicas son conocidas como réplica *early*, *prompt* y *late*, respectivamente. Los tres códigos son regularmente separados por ½ chip.

Seguimiento de fase/frecuencia de la portadora: La otra parte del seguimiento es el seguimiento de la onda portadora. Este seguimiento puede ser hecho de dos maneras. Ya sea por seguimiento de fase de la señal o por seguimiento de la frecuencia.

El seguimiento se está ejecutando continuamente para seguir los cambios en frecuencia en función del tiempo. Si el receptor pierde la pista de uno de los satélites, una nueva adquisición deberá ser realizada sobre el satélite en particular.

3.3.3 Extracción de los datos de navegación.

Cuando las señales son correctamente seguidas, el código C/A y la onda portadora pueden ser eliminados de la señal dejando solo los bits del dato de navegación. El valor de un bit del dato de navegación es encontrado por la integración sobre un periodo de bit de navegación de 20 ms.

Después de leer alrededor de 30 segundos de datos, el comienzo de una subtrama será encontrado para encontrar el momento cuando el dato fue transmitido por el satélite.

Cuando el tiempo de transmisión fue encontrado, el dato de efemérides para el satélite será decodificado. Este se usara después para calcular la posición del satélite en el momento de la transmisión.

La última cosa que se debe hacer antes de realizar los cálculos de posición es calcular el pseudorango. Los pseudorangos son calculados basándose en el momento de transmisión del satélite y el tiempo de llegada al receptor. El tiempo de llegada se basa en el comienzo de la subtrama.

3.4 Cálculos de posición.

La última parte del receptor es calcular la posición del usuario. La posición es calculada del pseudorango y la posición del satélite encontrada por los datos de efemérides. La figura 3.10 da una muestra del método de cálculo de posición usando GPS en donde con el conocimiento de la posición de los satélites y la distancia que viaja la señal, puede ser calculada la posición del usuario. [4].

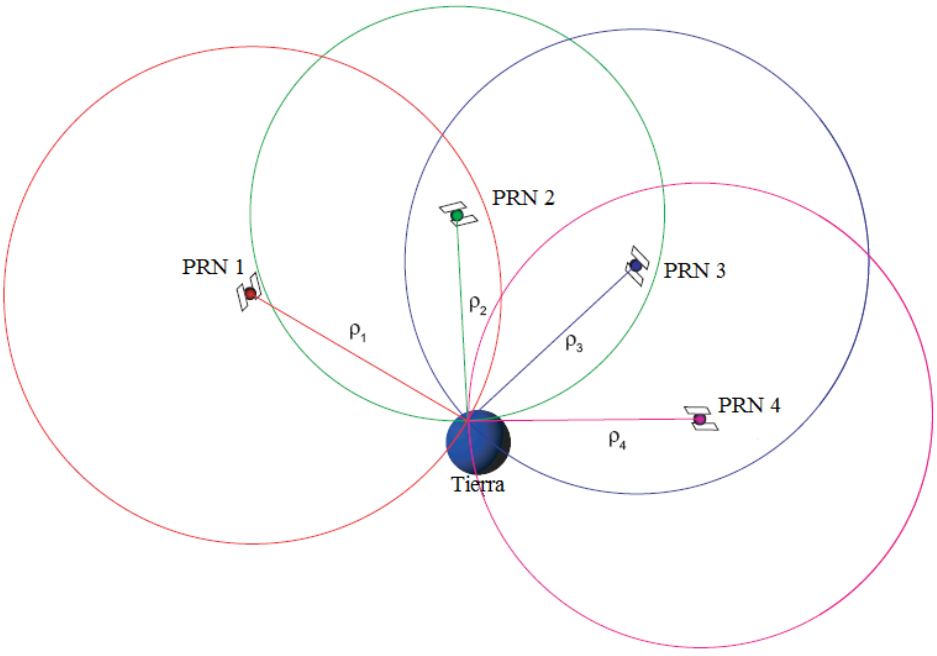

FIGURA 3.10. Principio básico de posicionamiento GPS.

Capítulo 4

Seguimiento de Código y de Portadora y Datos de navegación

En este capítulo se explica la parte teórica del tema de nuestra tesis. Aquí se describirán los métodos de seguimiento de la señal GPS. Primero, se explica un esquema de demodulación para poder extraer los datos de navegación. Además, se explican los métodos de seguimiento de la señal GPS: seguimiento de portadora y seguimiento de código y la decodificación de los datos de navegación.

4.1 Introducción.

La adquisición proporciona solo una estimación de los parámetros de frecuencia y fase de código. El propósito principal del seguimiento es refinar estos valores, mantener la pista, y demodular los datos de navegación para un satélite específico (y proporcionar una estimación del pseudorango). Un esquema de demodulación básico es el mostrado en la figura 4.1 [4].

La figura 4.1 muestra el esquema usado para demodular la señal de entrada para obtener el mensaje de navegación. Primero, la señal de entrada es multiplicada con una réplica de la portadora. Esto se hace para remover la onda portadora de la señal. En el siguiente paso, la señal es multiplicada con una réplica de código, y la salida de esta multiplicación da el mensaje de navegación.

Entonces el modulo de seguimiento tiene que generar dos réplicas, una para la portadora y una para el código, para poder tener un seguimiento perfecto y demodulación de la señal de un satélite. A continuación se dará una explicación detallada del esquema de demodulación.

4.2 Demodulación.

Permitiendo a f_{L1} y f_{L2} ser las frecuencias portadoras de L1 y L2 para la señal transmitida del satélite k con potencias P_C, P_{PL1} y P_{PL2} para el código C/A o código P. La secuencia de código C/A es $C^k(t)$ y la secuencia de código P(Y) es $P^k(t)$. Si la secuencia de datos de navegación es llamada $D^k(t)$, la señal total está dada como:

$$s^k(t) = \sqrt{2P_C}\, C^k(t)D^k(t)\cos(2\pi f_{L1}t) + \sqrt{2P_{PL1}}\, P^k(t)D^k(t)\sin(2\pi f_{L1}t) + \sqrt{2P_{PL2}}\, P^k(t)D^k(t)\sin(2\pi f_{L2}t) \qquad (4.1)$$

La salida de la terminal de entrada incluyendo el filtrado y la conversión de bajada puede ser descrita como:

$$s^k(t) = \sqrt{2P_C}\, C^k(t)D^k(t)\cos(\omega_{IF}t) + \sqrt{2P_{PL1}}\, P^k(t)D^k(t)\sin(\omega_{IF}t) \qquad (4.2)$$

donde ω_{IF} es la frecuencia intermedia con la cual la terminal de entrada hizo la conversión de bajada a la frecuencia portadora. La ecuación (4.2) describe la salida de la terminal de entrada de un satélite.

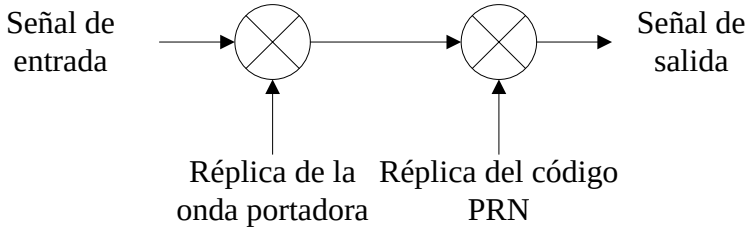

FIGURA 4.1. Esquema básico de demodulación.

Esta señal es entonces muestreada por un convertidor A/D. Porque del filtro pasa banda de banda angosta alrededor del código C/A, el código P es distorsionado. De ésta manera el último término de la ecuación (4.2) es limpiado y no puede ser demodulado y entonces se describe como ruido $e(n)$.

La señal del satélite k después de la conversión A/D puede ser descrita como:

$$s^k(n) = C^k(n)D^k(n)\cos(\omega_{IF}n) + e(n) \tag{4.3}$$

con n en unidades de $1/f_s$ (s); n indica que la señal es discreta en el tiempo.

Para obtener el dato de navegación $D^k(n)$ de la señal anterior, la señal tiene que ser convertida a banda base. La eliminación de la portadora se hace multiplicando la señal de entrada con una réplica de la portadora como se mostró en la figura 4.1. Si la réplica de la portadora es idéntica a la portadora entrante tanto en frecuencia como en fase, el producto de ambos es:

$$\begin{aligned} s^k(n)\cos(\omega_{IF}n) &= C^k(n)\,D^k(n)\cos(\omega_{IF}n)\cos(\omega_{IF}n) \\ &= -\frac{1}{2}C^k(n)D^k(n) - \frac{1}{2}\cos(2\omega_{IF}n)\,C^k(n)D^k(n) \end{aligned} \tag{4.4}$$

donde el primer término es el mensaje de navegación multiplicado por el código PRN y el segundo término es una portadora con el doble de la frecuencia intermedia. La última parte de la señal puede ser removida solo aplicando un filtro pasa bajas. La señal después del filtro pasa bajas es:

$$\frac{1}{2}C^k(n)D^k(n) \tag{4.5}$$

El siguiente paso es remover el código $C^k(n)$ de la señal. Esto se hace correlacionando la señal con una réplica local de código. Si la réplica de código es exactamente la misma como el código en la señal, la salida de la correlación es:

$$\sum_{n=0}^{N-1} C^k(n)C^k(n)D^k(n) = ND^k(n) \tag{4.6}$$

donde $ND^k(n)$ es el mensaje de navegación multiplicado por una cantidad de puntos en la señal N.

La descripción anterior de la demodulación es solo para una señal con un satélite. Esto se hace para reducir la complejidad de las ecuaciones y dar una idea simple del esquema de demodulación. En la señal real hay una contribución de señal de cada satélite visible resultando un grande término de ruido en las ecuaciones; ver [12].

En el esquema de demodulación mostrado en la figura 4.1, dos réplicas locales de la señal son generadas. Para producir una réplica exacta se necesitara algún tipo de retroalimentación. El lazo de retroalimentación para producir la réplica de la portadora es llamado como *lazo de seguimiento de portadora*, y el lazo de retroalimentación que produce la réplica exacta de código es llamado como *lazo de seguimiento de código*.

4.3 PLL de segundo orden.

Tanto el seguimiento de portadora (lazo de Costas) y el seguimiento de código (DLL) tienen un modelo analítico lineal de lazo de seguimiento de fase que puede ser usado para predecir su rendimiento. Este modelo lineal ha sido derivado por [27] y es una herramienta extremadamente poderosa que predice el rendimiento del lazo de seguimiento. Otra excelente referencia, una vez que los modelos fundamentales para el lazo de Costas y el DLL han sido derivados, para el lazo de seguimiento de fase lineal y sus parámetros y rendimiento es por [3, 8].

El sistema del PLL de segundo orden contiene un filtro de primer orden y un oscilador controlado por voltaje (VCO). Notar que la función de transferencia de un filtro de lazo analógico y un VCO es:

$$F(s) = \frac{1}{s}\frac{\tau_2 s + 1}{\tau_1} \tag{4.7}$$

$$N(s) = \frac{K_0}{s} \tag{4.8}$$

donde $F(s)$ y $N(s)$ son las funciones de transferencia del filtro y el NCO, respectivamente. K_0 es la ganancia del NCO. La función de transferencia de un PLL analógico linealizado es [6, 8]:

$$H(s) = \frac{K_d F(s) N(s)}{1 + K_d F(s) N(s)} \tag{4.9}$$

donde K_d es la ganancia del discriminador de fase. Sustituyendo las ecuaciones (4.7) y (4.8) en la función de transferencia (4.9) dando como resultado:

$$H(s) = \frac{2\zeta\omega_n s + \omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \tag{4.10}$$

donde la frecuencia natural es $\omega_n = \sqrt{(K_0 K_d)/\tau_1}$, y el factor de amortiguamiento esta dado por $\zeta = (\tau_2\omega_n)/2$. La función de transferencia anterior está en su versión analógica y para convertir la función de transferencia a la forma digital, se usa una transformación bilineal sobre (4.10). Entonces produce la siguiente función de transferencia para el modelo del PLL digital:

$$H_1(z) = \frac{(4\zeta\omega_n T + (\omega_n T)^2) + 2(\omega_n T)^2 z^{-1} + ((\omega_n T)^2 - 4\zeta\omega_n) z^{-2}}{(4 + 4\zeta\omega_n T + (\omega_n T)^2) + (2(\omega_n T)^2 - 8) z^{-1} + (4 - 4\zeta\omega_n T + (\omega_n T)^2) z^{-2}} \tag{4.11}$$

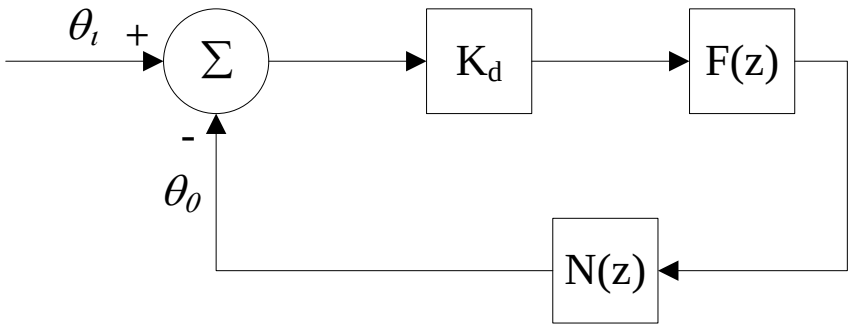

FIGURA 4.2. Modelo del PLL de segundo orden digital linealizado

El modelo del PLL de segundo orden digital linealizado es mostrado en la figura 4.2, donde K_d es el discriminador de la ganancia, $F(z)$ es la función de transferencia del filtro, y $N(z)$ es la función de transferencia del NCO.

Las funciones de transferencia para el filtro digital y el NCO son:

$$F(z) = \frac{(C_1 + C_2) - C_1 z^{-1}}{1 - z^{-1}} \tag{4.12}$$

$$N(z) = \frac{K_0 z^{-1}}{1 - z^{-1}} \tag{4.13}$$

donde $F(z)$ es la función de transferencia del filtro y $N(z)$ es la función de transferencia del NCO. La figura 4.3 muestra el filtro del lazo de seguimiento.

El objetivo es encontrar los coeficientes C_1 y C_2 en el PLL de segundo orden. Esto se hace comparando la función de transferencia para el PLL digital y la función de transferencia del PLL analógico. La función de transferencia para la versión digital puede ser encontrada como:

$$H(z) = \frac{\theta_0(z)}{\theta_i(z)} = \frac{K_d F(z) N(z)}{1 + K_d F(z) N(z)} \tag{4.14}$$

Sustituyendo (4.12) y (4.13) en (4.14) obtenemos lo siguiente:

$$H_2(z) = \frac{K_0 K_d (C_1 + C_2) z^{-1} - K_0 K_d C_1 z^{-2}}{1 + (K_0 K_d (C_1 + C_2) - 2) z^{-1} + (1 - K_0 K_d C_1) z^{-2}} \tag{4.15}$$

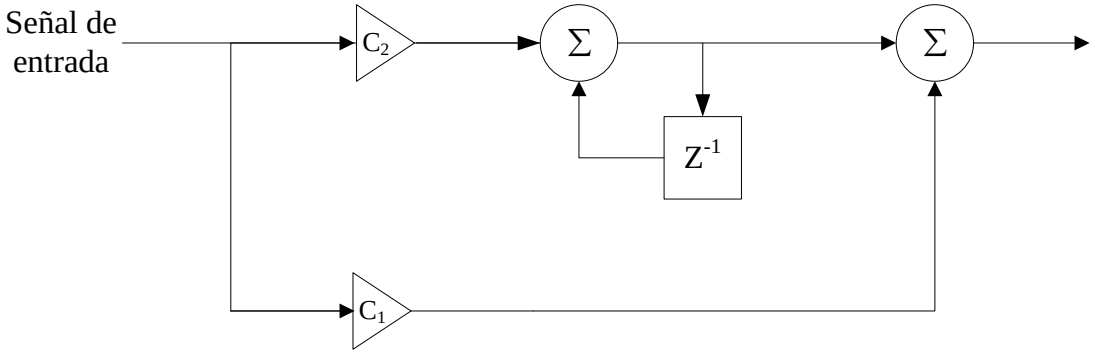

FIGURA 4.3. Filtro para el lazo de seguimiento de fase de segundo orden

Para encontrar una ecuación para los dos coeficientes C_1 y C_2, las ecuaciones (4.11) y (4.15) son comparadas. Esto produce las siguientes dos ecuaciones:

$$C_1 = \frac{1}{K_0 K_d} \frac{8\zeta\omega_n T}{4 + 4\zeta\omega_n T + (\omega_n T)^2} \tag{4.16}$$

$$C_2 = \frac{1}{K_0 K_d} \frac{4(\omega_n T)^2}{4 + 4\zeta\omega_n T + (\omega_n T)^2} \tag{4.16}$$

donde $K_0 K_d$ es la ganancia del lazo, ζ es el factor de amortiguamiento, ω_n es la frecuencia natural, y T es el tiempo de muestreo.

La frecuencia natural puede ser encontrada como:

$$\omega_n = \frac{8\zeta B_L}{4\zeta^2 + 1} \tag{4.18}$$

donde B_L es el ancho de banda del ruido en el lazo; ver [22].

El factor de amortiguamiento y el ancho de banda del ruido son calculados para un caso particular de la señal. Pero en algunos casos en ingeniería podrían cambiar estos valores para aplicaciones e implementaciones específicas. Por lo tanto, es dada una explicación más a fondo acerca de estos parámetros.

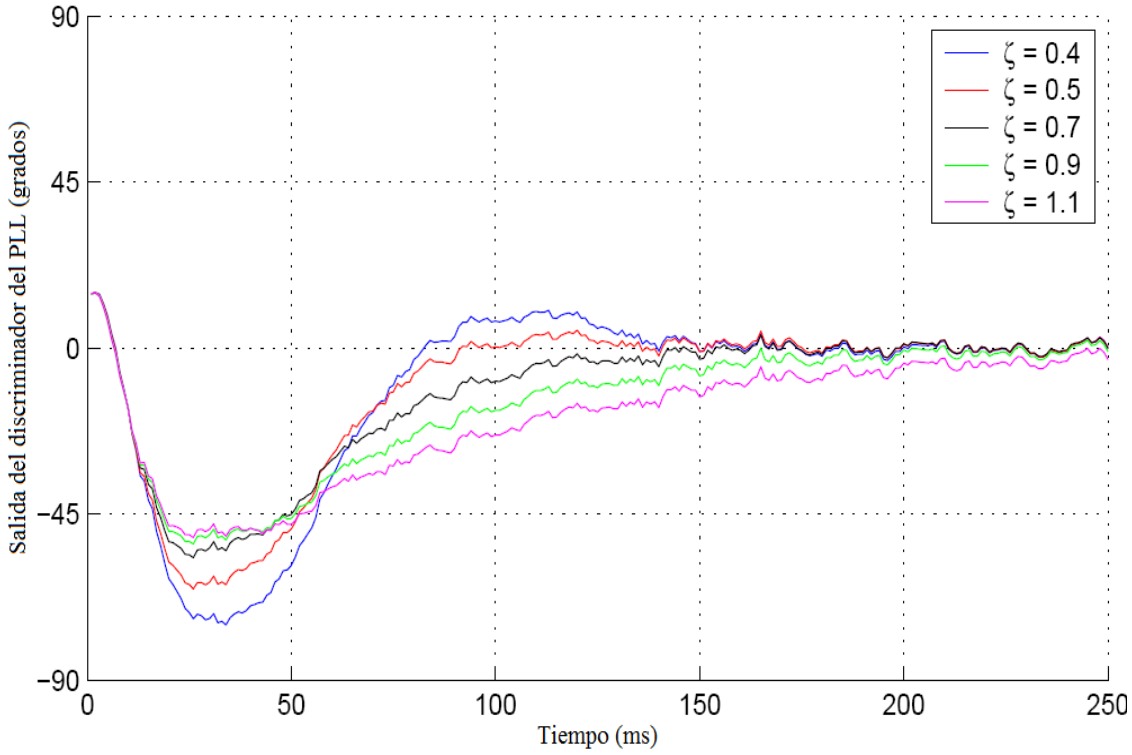

FIGURA 4.4. Error de fase como función de diferentes factores de amortiguamiento ζ.

4.3.1 Factor de amortiguamiento.

El factor de amortiguamiento controla la velocidad con la que el filtro llega a su punto de asentamiento. El factor de amortiguamiento también controla que tanto sobretiro puede tener el filtro. Un tiempo de asentamiento pequeño resulta en un sobretiro mayor y viceversa, un tiempo de asentamiento grande resulta en un sobretiro pequeño de la fase. Esto puede ser visto en la figura 4.4. La selección del factor de amortiguamiento es un compromiso entre el sobretiro y el tiempo de asentamiento. El factor de amortiguamiento es escogido a $\zeta = 0.7$ resultando en un filtro que converge razonablemente rápido y no hace un alto sobretiro.

4.3.2 Ancho de banda del ruido.

El segundo parámetro en el filtro PLL es el ancho de banda del ruido B_L. El ancho de banda del ruido controla la cantidad de ruido permitido en el filtro. Este factor puede también, como el factor de amortiguamiento, controlar el tiempo de asentamiento. Cuando el lazo de seguimiento comienza a seguir la señal la frecuencia de comienzo es la frecuencia encontrada por el algoritmo de adquisición. (Esta fase es algunas veces llamada

en la literatura como Rango de captura (Pull – in) [8]. Es en esta fase donde el filtro está tratando de converger a frecuencia y fase correcta). La frecuencia de comienzo del algoritmo de adquisición puede estar fuera de sitio por algunos hertz (Hz). Entonces, el lazo de seguimiento busca el bloqueo en la frecuencia correcta. Para ver el impacto de varios anchos de banda del ruido, una señal GPS real se utiliza cuando el algoritmo de adquisición encontró una frecuencia que está fuera de sitio alrededor de 21 Hz. La figura 4.5 muestra el desbalance de la frecuencia inicial para tres diferentes anchos de banda de ruido.

La figura 4.5 muestra que si el ancho de banda del ruido es 40 Hz, el lazo de seguimiento inmediatamente encuentra el desbalance de frecuencia correcto alrededor de 21 Hz. También se puede observar que una gran cantidad de ruido está permitida sobre el seguimiento de la frecuencia. En el segundo caso, donde el ancho de banda del ruido es 15 Hz, el lazo de seguimiento también bloquea sobre la señal muy rápido. Aquí se puede observar que el ruido sobre el seguimiento de la frecuencia es mucho más pequeño que con un ancho de banda de ruido de 40 Hz. En el tercer caso, donde el ancho de banda del ruido es 10 Hz, el lazo de seguimiento no es suficientemente rápido en alcanzar la frecuencia real antes de que ocurra un movimiento en la fase. Por lo tanto, no es probable que converja al valor apropiado. Los picos negativos se deben a la transición de fase en el lazo del filtro.

Un ancho de banda de ruido mayor implica que el lazo de seguimiento rápidamente bloqueé a la frecuencia real pero tiene una frecuencia relativamente grande de ruido en el estado de bloqueo. Un ancho de banda de ruido pequeño implica que esté pueda tomar algunos tiempos antes que el lazo de seguimiento pueda estar bloqueado a la frecuencia, pero después el bloqueo de la frecuencia es estable. Algunas implementaciones dividen al PLL en dos filtros, a menudo llamado Rango de Captura (Pull-in) y filtros de seguimiento.

Para aplicaciones, un valor típico para el ancho de banda del ruido está alrededor de 20 Hz.

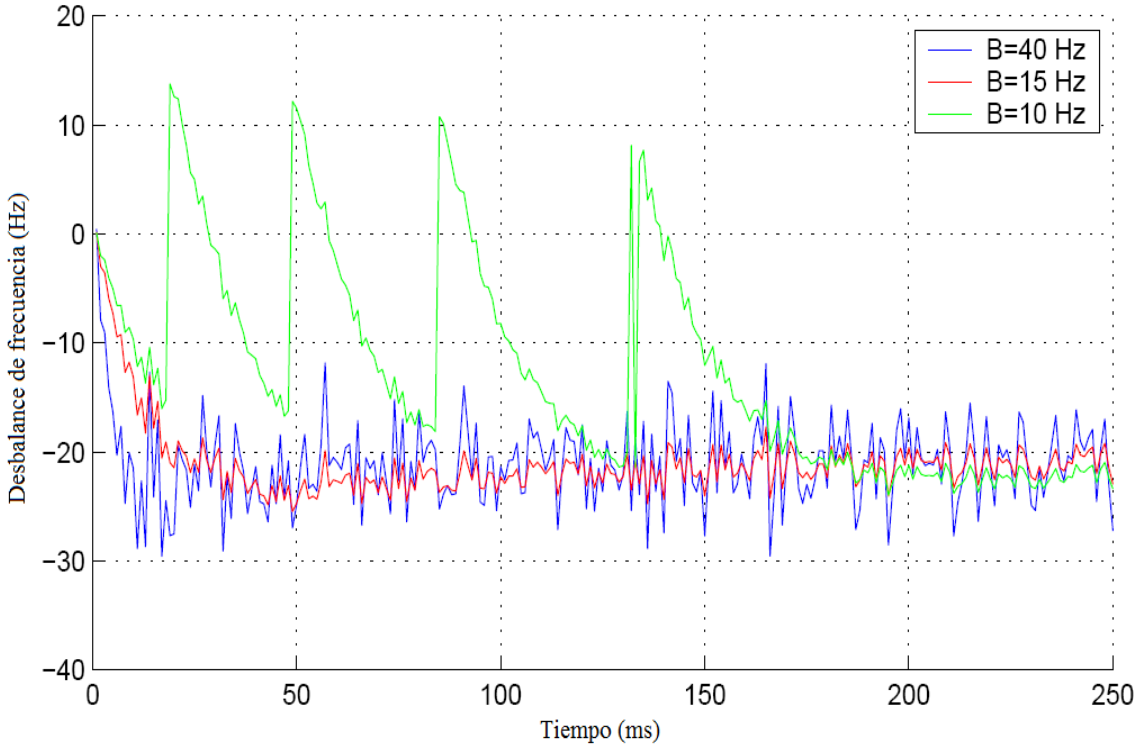

FIGURA 4.5. Desbalance de frecuencia de una señal adquirida.

4.4 Seguimiento de Portadora.

Para demodular los datos de navegación satisfactoriamente se tiene que generar una réplica exacta de la onda portadora. Para seguir la onda portadora de la señal, los lazos de seguimiento de fase (PLL) o los lazos de seguimiento de frecuencia (FLL) son regularmente usados.

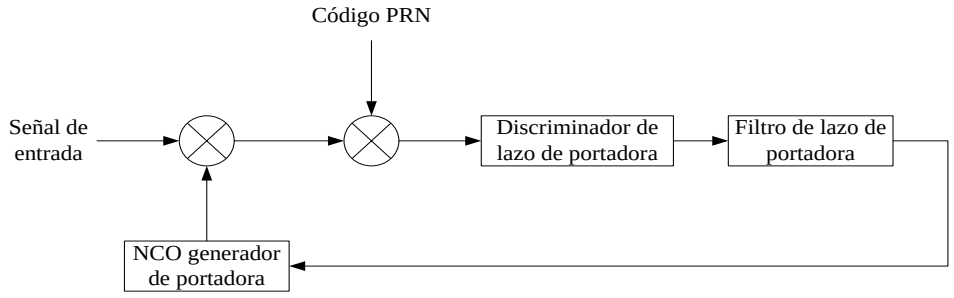

FIGURA 4.6. Diagrama a bloques básico del lazo de seguimiento de portadora del receptor GPS.

La figura 4.6 muestra un diagrama a bloques básico para un lazo de seguimiento de fase. Las primeras dos multiplicaciones quitan la portadora y el código PRN de la señal de entrada. Para quitar el código PRN, la salida I_P del lazo de seguimiento de código early – late descrito más adelante es usada. El bloque del discriminador de lazo es usado para encontrar el error de la fase sobre la réplica local de la onda portadora. La salida del discriminador, qué es el error de la fase(o una función del error de la fase), es entonces filtrada y se utiliza como retroalimentación al oscilador controlado numéricamente (NCO), qué ajusta la frecuencia de la onda portadora local. De esta manera la onda portadora local será una réplica casi precisa de la onda portadora de la señal de entrada.

El problema con usar un PLL ordinario es que este es sensible a los cambios de fase de 180º. Debido a la transición de los bits de navegación, el PLL utilizado en un receptor GPS tiene que ser insensible a los cambios de fase de 180º.

La figura 4.7 muestra un lazo de Costas. Una propiedad de este lazo es que es insensible para los cambios de fase de 180º y por esto un lazo de Costas es insensible a las transiciones de fase debido a los bits de navegación. Esta es la razón por la que usamos este lazo de seguimiento de portadora en los receptores GPS. La figura 4.7 contiene dos multiplicaciones. La primera multiplicación es el producto entre la señal de entrada y la onda portadora local, la segunda multiplicación es entre la onda portadora con un cambio de fase de 90º y la señal de entrada. El *objetivo del lazo de Costas es tratar de mantener toda la energía en la rama I (In-phase).* Para mantener toda la energía en la rama *I*, algunos tipos de retroalimentación al oscilador son necesitados. Si esto asume que la réplica de código en la figura 4.7 está perfectamente alineada, la multiplicación en la rama *I* produce la siguiente suma:

$$D^k(n)\cos(\omega_{IF}n)\cos(\omega_{IF}n+\varphi)=\frac{1}{2}D^k(n)\cos(\varphi)+\frac{1}{2}D^k(n)\cos(2\omega_{IF}n+\varphi), \tag{4.19}$$

donde φ es la diferencia de fase entre la fase de la señal de entrada y la fase de la réplica local de la portadora. La multiplicación en la rama *Q (Quadrature).*está dada por lo siguiente:

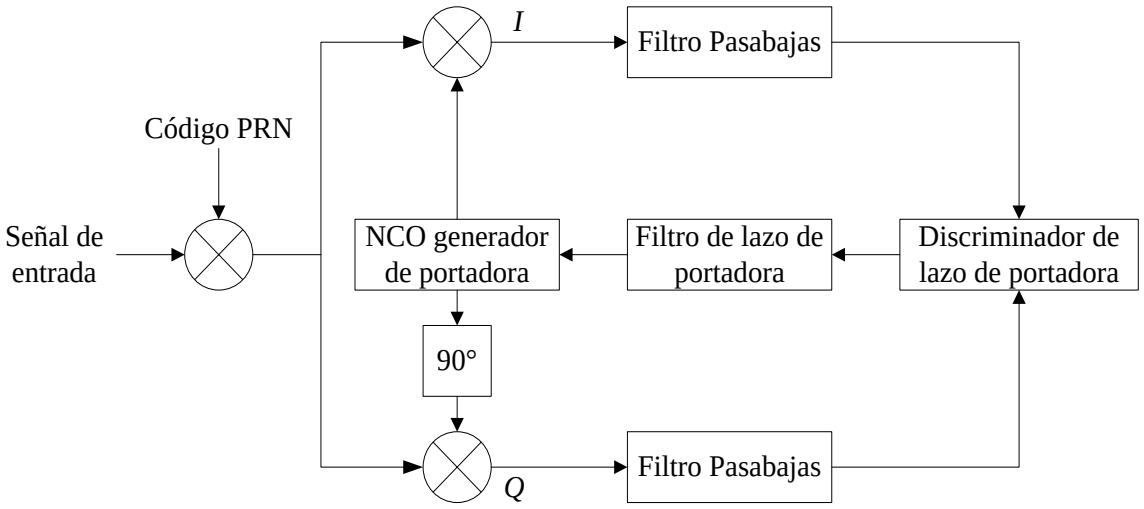

FIGURA 4.7. Lazo de Costas usado para seguir la onda portadora.

$$D^k(n)\cos(\omega_{IF}n)\sin(\omega_{IF}n+\varphi)=\frac{1}{2}D^k(n)\sin(\varphi)+\frac{1}{2}D^k(n)\sin(2\omega_{IF}n+\varphi), \tag{4.20}$$

Si las dos señales son filtradas por el filtro pasa bajas después de la multiplicación, los dos términos con el doble de la frecuencia intermedia son eliminadas y las siguientes dos señales quedan de la siguiente manera:

$$I^k=\frac{1}{2}D^k(n)\cos(\varphi) \tag{4.21}$$

$$Q^k=\frac{1}{2}D^k(n)\sin(\varphi) \tag{4.22}$$

Para encontrar un término de retroalimentación al oscilador de fase de portadora, este puede ser visto como el error de la fase de la réplica local de la portadora y puede ser encontrada como:

$$\frac{Q^k}{I^k}=\frac{\frac{1}{2}D^k(n)\sin(\varphi)}{\frac{1}{2}D^k(n)\cos(\varphi)}=\tan(\varphi) \tag{4.23}$$

$$\varphi = \tan^{-1}\left(\frac{Q^k}{I^k}\right) \tag{4.24}$$

De la ecuación (4.24), puede ser visto que el error de la fase es minimizado cuando la correlación en la rama de fase *Quadrature* es cero y el valor de correlación en la rama *In-phase* es máxima. El discriminador de la ecuación (4.24) es el más preciso del los discriminadores Costas, pero este es también el que más tiempo consume. La tabla 4.1 describe otros posibles discriminadores Costas.

TABLA 4.1. Varios tipos de discriminadores de lazo de seguimiento de fase Costas.

Discriminadores	**Descripción**
$D = sign(I^k)Q^k$	La salida del discriminador es proporcional a $\sin(\varphi)$
$D = I^k Q^k$	La salida del discriminador es proporcional a $\sin(2\varphi)$
$D = \tan^{-1}\left(\frac{Q^k}{I^k}\right)$	La salida del discriminador es el error de la fase

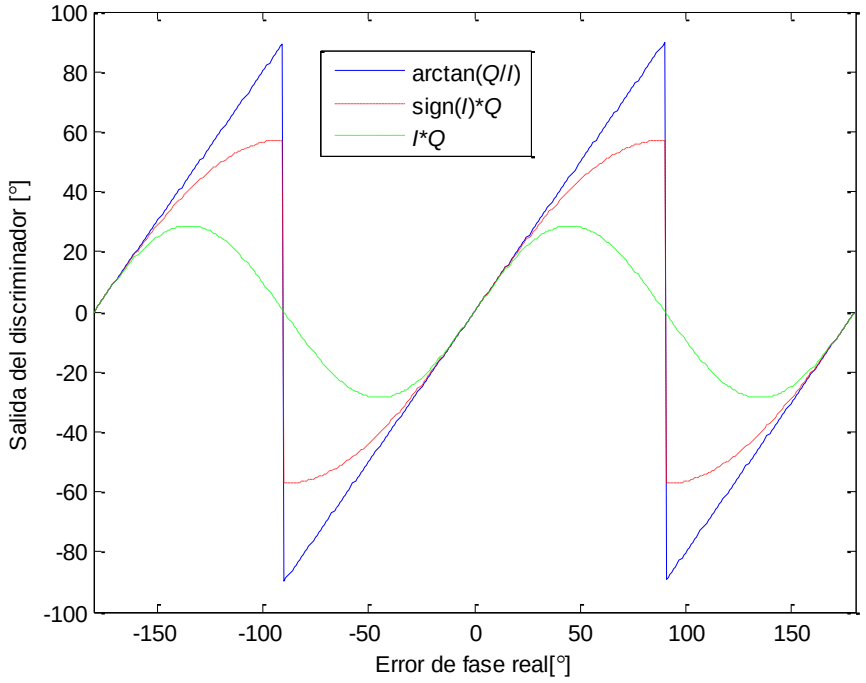

FIGURA 4.8. Comparación entre las respuestas de los discriminadores de lazo Costas

La figura 4.8 muestra las respuestas correspondientes a los diferentes discriminadores. La salida del discriminador de fase en esta figura es calculada usando las expresiones de la tabla 4.1 para todos los posibles errores de fase. En la misma figura se puede ver que la salida del discriminador es cero cuando el error de fase real es 0 y ±180º. Esto es porque el lazo de Costas es insensible a los cambios de fase 180º debido a la transición de los bits de navegación.

El comportamiento del lazo de Costas cuando ocurre un cambio de fase de 180º está claramente ilustrado en la figura 4.9. En esta figura el vector suma de I^k y Q^k es mostrado el vector en el sistema de coordenadas. Si la onda portadora local estaba en fase con la señal de entrada, el vector estará alineado con el eje *I*. Está propiedad asegura que si la transición de un bit de navegación ocurre, el vector sobre el diagrama de fase cambiara 180º (mostrado por el vector punteado en la figura). Si la transición de un bit de navegación ocurre, el lazo de Costas aun así seguirá la señal y nada pasara. Esta propiedad hace que el lazo de Costas sea el escogido comúnmente dentro de los lazos de seguimiento de fase en los receptores GPS; ver [16].

La salida del discriminador de fase es filtrada para predecir y estimar un movimiento relativo del satélite y estimar la frecuencia Doppler.

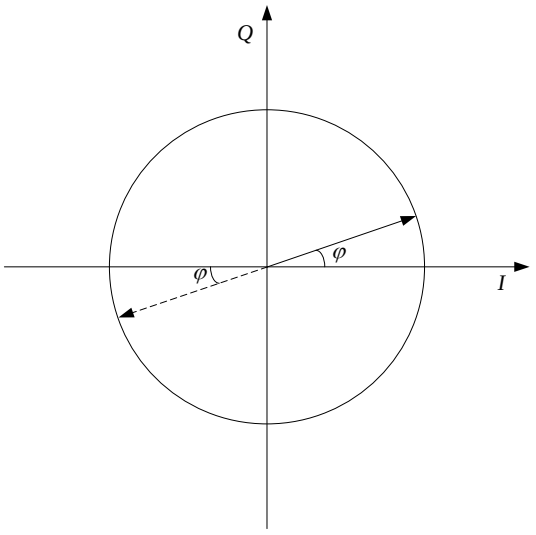

FIGURA 4.9. Diagrama de fase mostrando el error de fase entre la fase de la onda portadora de entrada y la fase de la réplica local de la onda portadora.

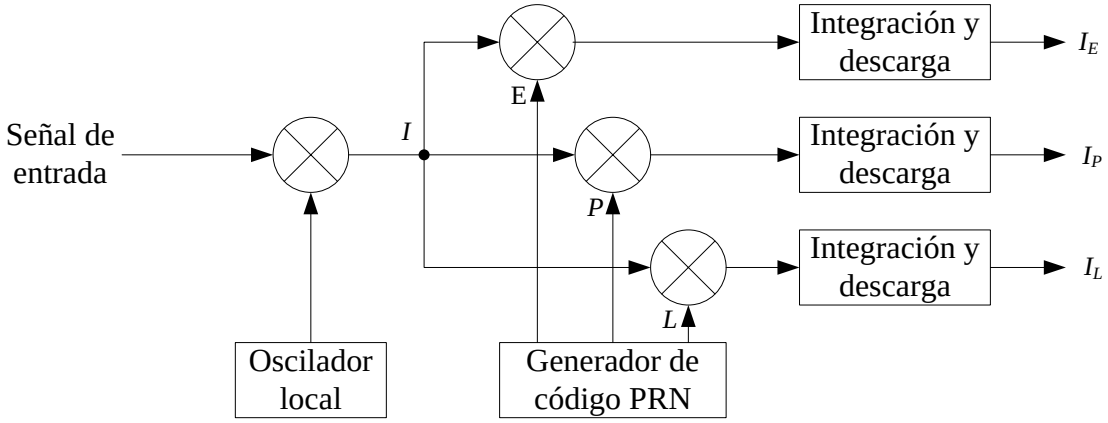

FIGURA 4.10. Diagrama a bloques básico del lazo de seguimiento de código.

4.5 Seguimiento de Código.

El objetivo para un lazo de seguimiento de código es mantener la pista de la fase de código de un código específico en la señal. La salida de un lazo de seguimiento de código es semejante a una réplica de código perfectamente alineada. El lazo de seguimiento de código en los receptores GPS un lazo de seguimiento de retrazo (*DLL – Delay Lock Loop*) llamado también como lazo de seguimiento early – late. La idea detrás del DLL es correlacionar la señal de entrada con las tres réplicas de código como se muestra en la figura 4.10.

El primer paso en la figura 4.10 es convertir el código C/A a banda base, para multiplicar la señal de entrada con una réplica local perfectamente alineada de la onda portadora. Después la señal es multiplicada con tres réplicas de código. Las tres réplicas están normalmente generadas con un espaciamiento de ±1/2 chip. Después de esta segunda multiplicación, las tres salidas pasan a la etapa de integración y descarga. La salida de estas integraciones es un valor numérico el cual indica cuánto se parece la réplica de código con el código en la señal de entrada.

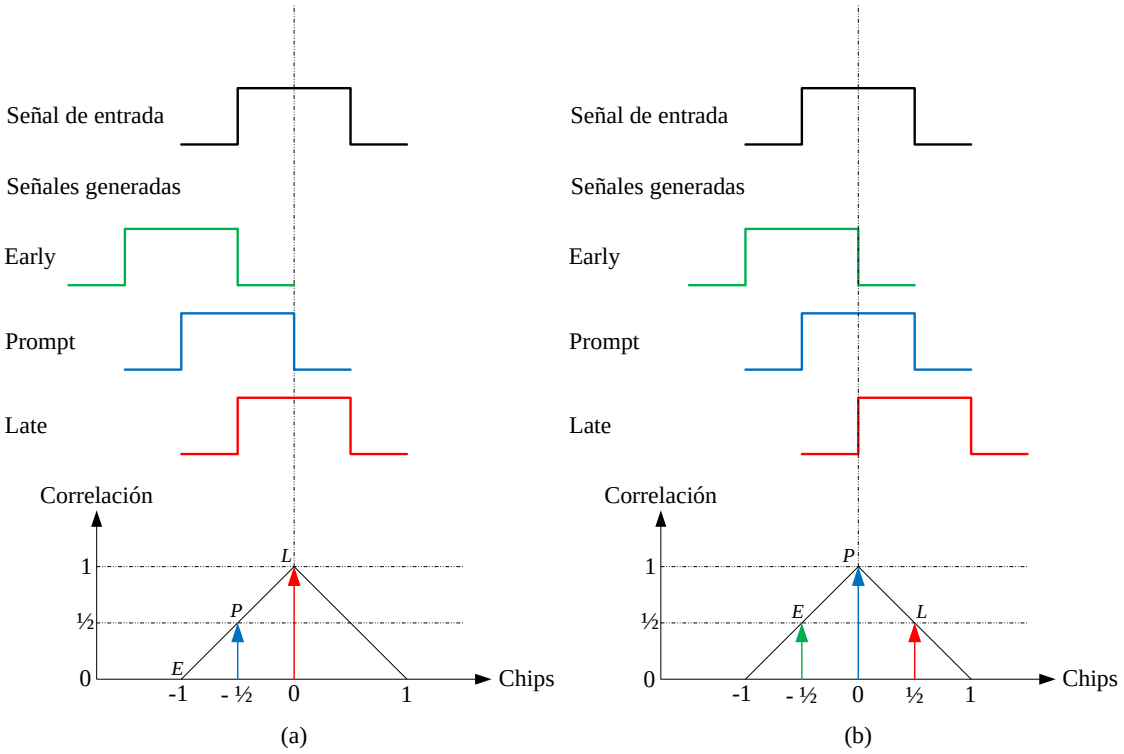

FIGURA 4.11. Seguimiento de código.

Las salidas de las tres correlaciones I_E, I_P y I_L son entonces comparadas para ver cuál de todas proporciona la correlación más alta. La figura 4.11 muestra un ejemplo del seguimiento de código.

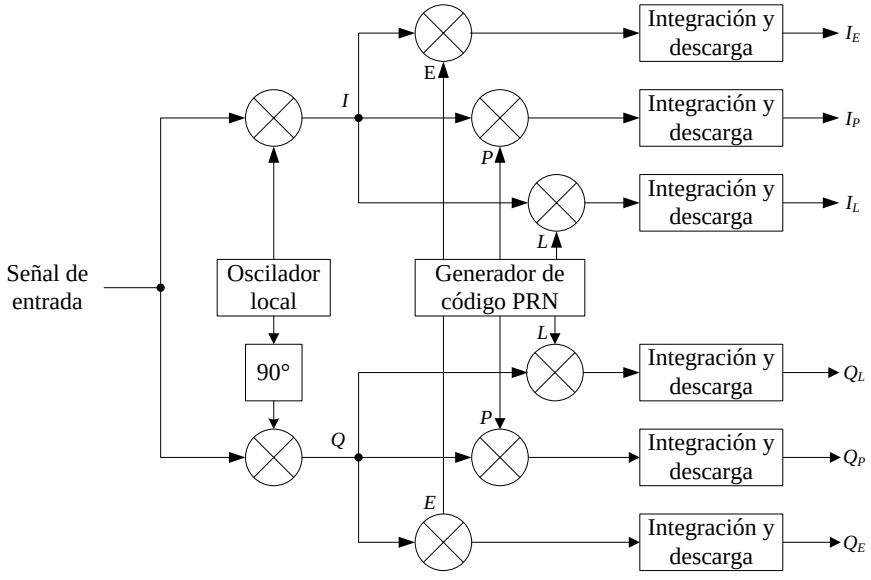

FIGURA 4.12. Diagrama a bloques del DLL con seis correladores.

En la figura 4.11(a) el código late tiene la correlación más alta, entonces la fase de código debe de ser disminuida, por ejemplo, la secuencia de código debe ser retrasada. En

la figura 4.11(b) el pico más alto está localizado en la réplica prompt, y las réplicas early y late tienen la misma correlación, la fase del código es correctamente seguida y el lazo está correctamente sincronizado.

El DLL con tres correladores como el de la figura 4.10 es óptimo cuando la onda portadora local está bloqueada en fase y frecuencia. Pero cuando hay en error de fase sobre la onda portadora local, la señal estará más ruidosa, haciendo esto más difícil para el DLL de mantener el bloqueo sobre el código. Entonces en lugar de eso el DLL en un receptor GPS es regularmente diseñado como se muestra en la figura 4.12. El diseño de la figura 4.12 tiene la ventaja que es independiente de la fase sobre la onda portadora local. Si la onda portadora local está en fase con la señal de entrada, toda la energía estará en la rama *In-Phase*. Pero si la fase de la portadora local cambia comparada con la fase de la señal de entrada, la energía cambiara entre la rama *In-phase* y la rama *Quadrature*. Para propósitos de demostración, la figura 4.13 muestra una situación donde la fase de la réplica de la portadora cambia comparada con la fase de la señal de entrada. En la parte superior muestra la salida de los tres correladores en la rama *I*, y en la parte inferior muestra la salida de correlación en la rama *Q* del DLL con seis correladores. Esta situación es un resultado de diferentes frecuencias la señal y la réplica; esto resulta en cambio constante de diferencias de fase (desalineamiento). Hay pocas razones porque esto puede pasar, por ejemplo, el PLL podrá no estar en estado de bloqueo. La figura 4.14 muestra un caso cuando el PLL está en el estado de bloqueo. Debido a la réplica precisa de la portadora del PLL, puede ser visto en la figura 4.14 que los correladores son constantes sobre el tiempo. Esto no sería el caso si la réplica de la portadora no es ajustada para igualar la frecuencia y fase de la señal de entrada.

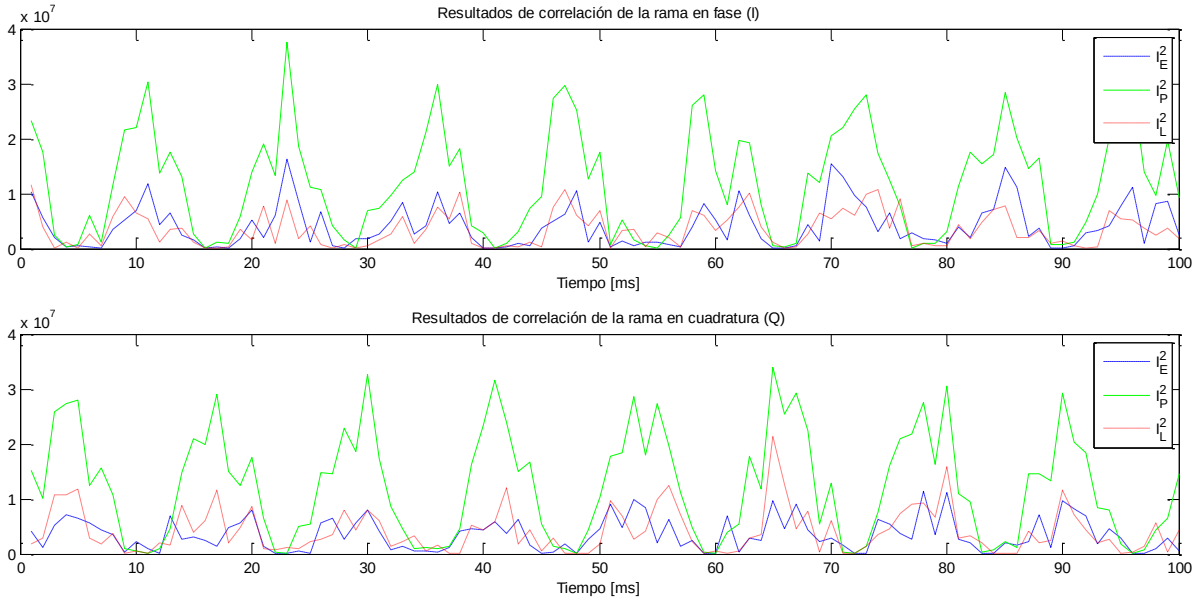

FIGURA 4.13. Salida de los seis correlacionadores en la rama I y Q cuando el PLL no está en bloqueo.

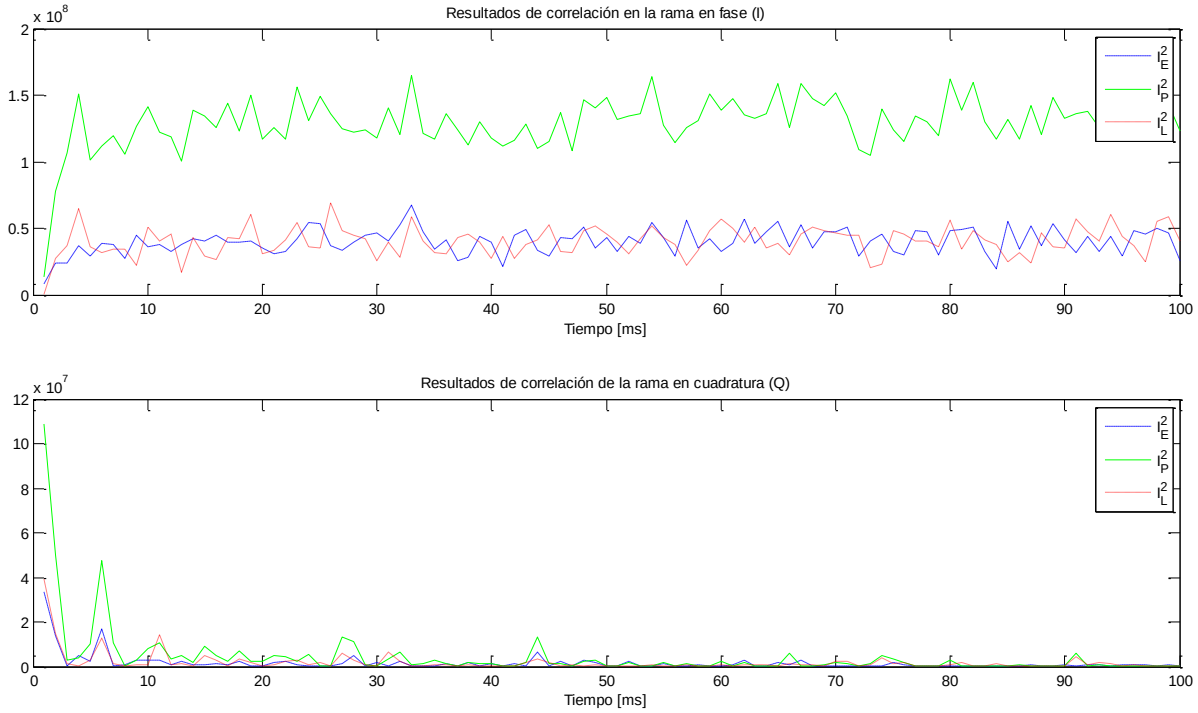

FIGURA 4.14. Salida de los seis correlacionadores en la rama I y Q cuando el PLL está en bloqueo.

TABLA 4.2 Varios tipos de discriminadores de lazo de seguimiento de código y su descripción.

Tipo	**Discriminador *D***	**Características**
Coherente	$I_E - I_L$	El más simple de todos los discriminadores. No requiere de la rama *Q* pero requiere un buen lazo de seguimiento de portadora para una funcionalidad óptima
No coherente	$(I_E^2 + Q_E^2) - (I_L^2 + Q_L^2)$	Early con menos pérdida de potencia. La respuesta del discriminador es aproximadamente la misma como el discriminador coherente dentro ±½ chip.
	$\frac{(I_E^2 + Q_E^2) - (I_L^2 + Q_L^2)}{(I_E^2 + Q_E^2) + (I_L^2 + Q_L^2)}$	Early normalizado con menos pérdida de potencia. El discriminador tiene una gran propiedad cuando el error de chip es mayor que ½ chip; esto ayudara al DLL para mantener el seguimiento en señales ruidosas.
	$I_P(I_E - I_L) + Q_P(Q_E - Q_L)$	Producto punto. Este es el único discriminador DLL que usa las seis salidas del correlador.

El desempeño del lazo de seguimiento de código tiene que ser independiente del desempeño de lazo de seguimiento de fase, el lazo de seguimiento tiene que usar las ramas *In-phase* y *Quadrature* para el seguir el código.

El DLL ahora necesita una retroalimentación para el generador de códigos PRN si la fase de código ha sido ajustada. Algunos discriminadores DLL comunes usados como retroalimentación son listados en la tabla 4.2.

La tabla 4.2 muestra un discriminador coherente y tres no coherentes. Los requerimientos de un discriminador DLL es que este va a depender del tipo de aplicación y el ruido en la señal. Las respuestas de la función del discriminador se muestran en la figura 4.15.

La figura 4.15 muestra el discriminador coherente y tres discriminadores no coherentes usando un correlador estándar. La figura es producida de la función de autocorrelación ideal, y el espacio entre el código early, prompt y late es ±½ chip.

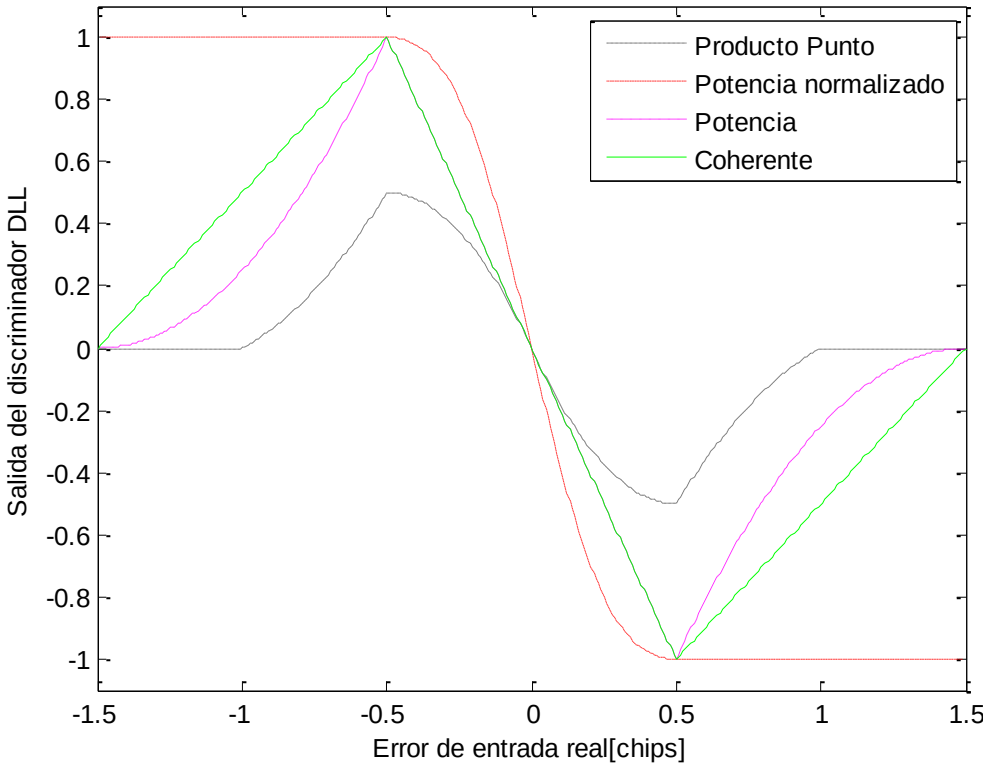

FIGURA 4.15. Comparación entre las respuestas de los discriminadores DLL comunes.

El espacio entre los códigos early, prompt y late determina el ancho de banda del ruido en el DLL. Si el espacio del discriminador es mayor que ½ chip, el DLL será capaz de responder a anchos dinámicos y será más robusto al ruido; un DLL con un espaciamiento más pequeño será más preciso. En un receptor GPS moderno el espaciamiento del discriminador puede ser ajustado mientras el receptor está siguiendo la señal. La ventaja de esto es que si la relación señal a ruido repentinamente decrece, el receptor usa un espaciamiento amplio en los correladores para ser capaz de manejar señales más ruidosas, y mediante esto una posible pérdida del bloqueo de código será eludido; ver [16].

El discriminador de lazo de seguimiento implementado es el early normalizado con menos pérdida de potencia. El discriminador es descrito como:

$$D = \frac{(I_E^2 + Q_E^2) - (I_L^2 + Q_L^2)}{(I_E^2 + Q_E^2) + (I_L^2 + Q_L^2)} \tag{4.25}$$

donde I_E, Q_E, I_L y Q_L son salidas de cuatro de los seis correladores mostrados en la figura 7.13. el discriminador early normalizado con menos pérdida de potencia es escogido porque este es independiente del desempeño del PLL ya que este usa tanto la rama *In-phase* como la de *Quadrature*. La normalización del discriminador causa que el discriminador pueda ser usado con señales con diferentes relaciones señal a ruido y diferentes señales.

El lazo de seguimiento genera tres réplicas locales de código. En esta sección, el espaciamiento de chips entre las réplicas early y prompt es medio chip.

Como fue descrito, el DLL puede ser modelado como un PLL lineal y por lo tanto el desempeño del lazo puede ser predicho basándose sobre este modelo. En otras palabras el diseño del filtro del lazo es el mismo, solo el valor de los parámetros son diferentes.

4.6 Bloque de Seguimiento Completo.

En las secciones previas, el lazo de seguimiento de código y el lazo de seguimiento de portadora son descritos a detalle. La siguiente descripción nos dice como el lazo de seguimiento de código y el lazo de seguimiento de portadora pueden ser unidos para minimizar la carga computacional.

La figura 4.16 muestra el lazo de seguimiento de código y el lazo de seguimiento de portadora combinados. Puede ser visto de la figura que la réplica de código PRN es usada para remover el código PRN en el lazo de seguimiento de portadora es próximo del lazo de seguimiento de código. El diagrama a bloques de la figura 4.19 contiene 11 multiplicaciones. Estas multiplicaciones son las operaciones que consumen más tiempo sobre el diagrama.

La figura 4.17 muestra una versión optimizada de la combinación de los lazos de seguimiento. Aquí las entradas *I* y *Q* al discriminador de fase son las correlaciones I_P y Q_P del lazo de seguimiento de código. De esta manera las tres multiplicaciones en el lazo de Costas son eliminadas, y de esta manera el tiempo de cómputo es reducido [4].

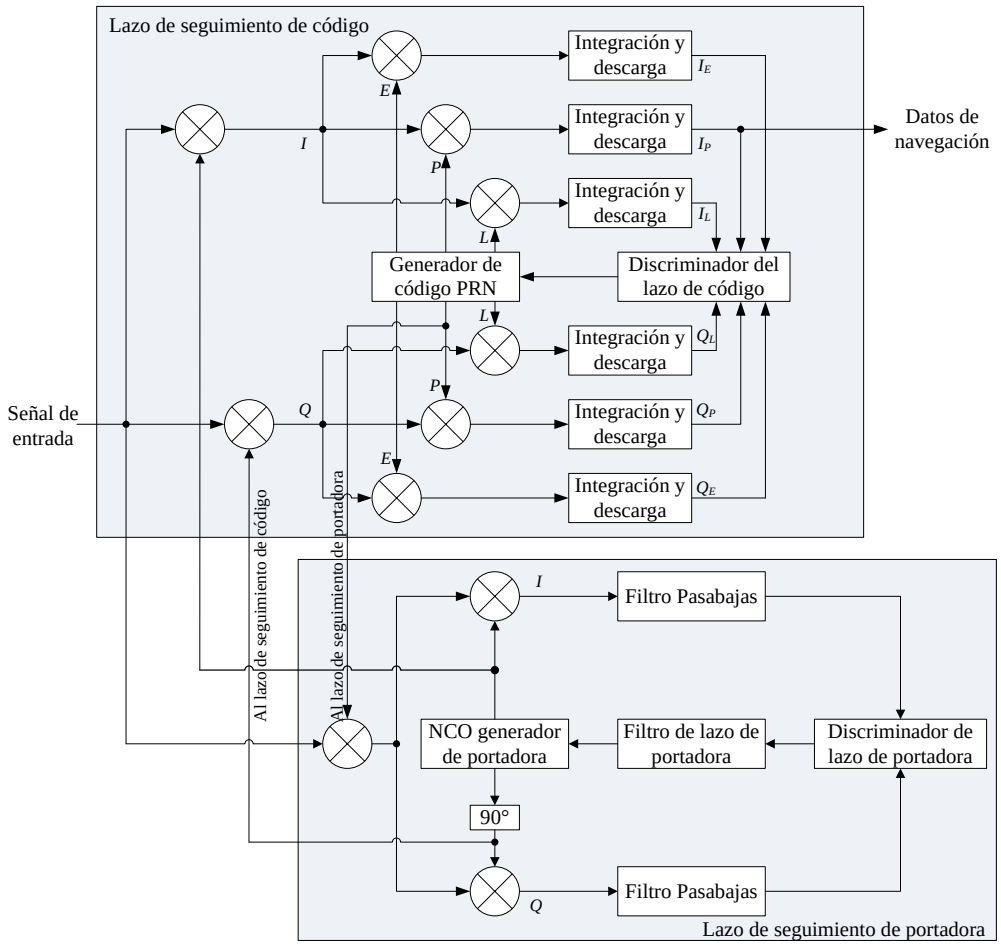

FIGURA 4.16. Diagrama a bloques de la unión de los lazos del DLL y PLL.

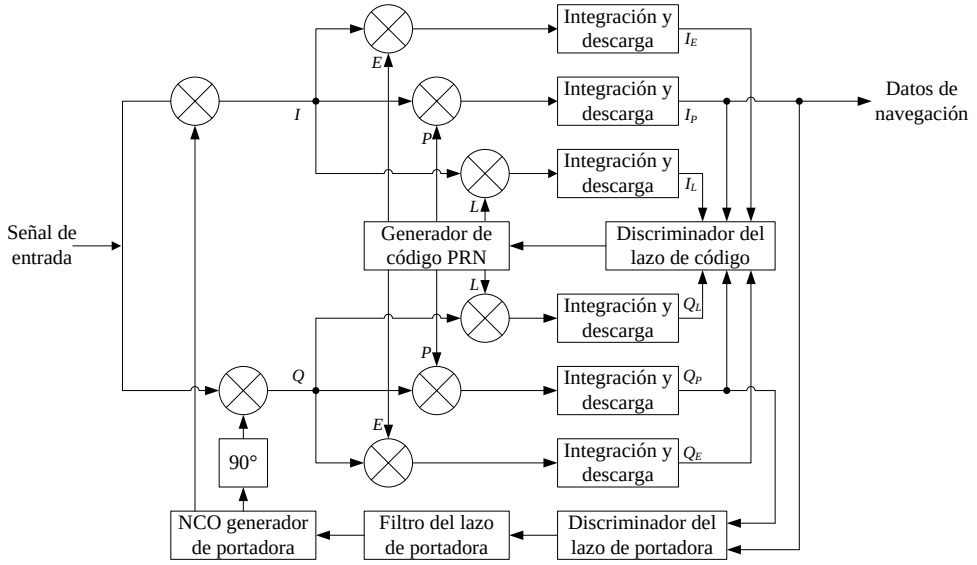

FIGURA 4.17. Diagrama a bloques de un canal de seguimiento completo sobre el receptor GSP.

4.7 Datos de navegación.

4.7.1 Recuperación de los datos de navegación.

La salida del lazo de seguimiento es el valor de la rama *I* (en fase) del bloque de seguimiento truncado a los valores 1 y -1. En teoría se podría obtener un valor de bit cada ms. Sin embargo, nos enfrentamos con señales ruidosas y débiles, por lo que un valor medio de 20 ms es calculado y truncado a -1 o 1. Un bit de navegación dura 20 ms.

La velocidad del dato de navegación es de 50 bps. La velocidad de muestreo de la salida del bloque de seguimiento es de 1000 sps correspondiendo a un valor cada ms. Antes que el dato de navegación pueda ser decodificado, la señal del bloque de seguimiento debe ser convertida de 1000 sps a 50 bps. Esto es, 20 valores consecutivos serán cambiados por solamente 1. A este procedimiento de conversión se le conoce como *sincronización de bit*.

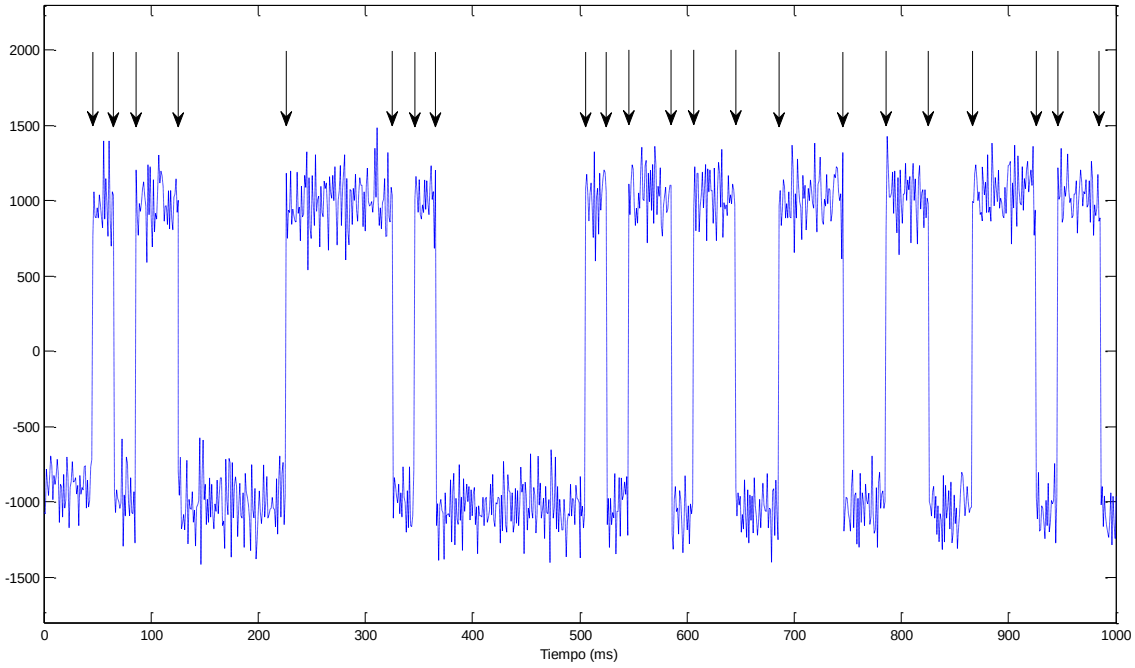

FIGURA 4.18. La figura muestra la salida (cruda) del bloque de seguimiento.

4.7.1.1 Encontrando el tiempo de transición y los valores del bit.

La primera tarea del procedimiento de sincronización de bit es encontrar el tiempo en una secuencia donde ocurre la transición de bit. Primero, se debe detectar los cruces por cero. Un cruce por cero es donde la salida cambia de 1 a -1, o viceversa. Cuando un cruce por cero es localizado, el tiempo de una transición de bit es localizado. Cuando el tiempo de una transición de bit es conocido, es posible encontrar todos los tiempos de la transición de bit. Estos se encuentran 20 ms separados del principio de la primera transición de bit detectada. La figura 4.18 muestra como todas las transiciones de bit son encontradas en una secuencia de 1 segundo. Las líneas punteadas marcan las transiciones de bit separadas cada 20 ms. La señal actual es muy fuerte, una señal débil tendrá puntos más cerca de cero. La salida del bloque de seguimiento truncado a 1 y -1 para una secuencia de 1 segundo se muestra en la figura 4.19.

Cuando el tiempo de la transición de bit es localizado, la señal a 1000 bps será convertida a una tasa de bit de 50 bps. Para hacer esto, 20 muestras serán remplazadas por solo un valor.

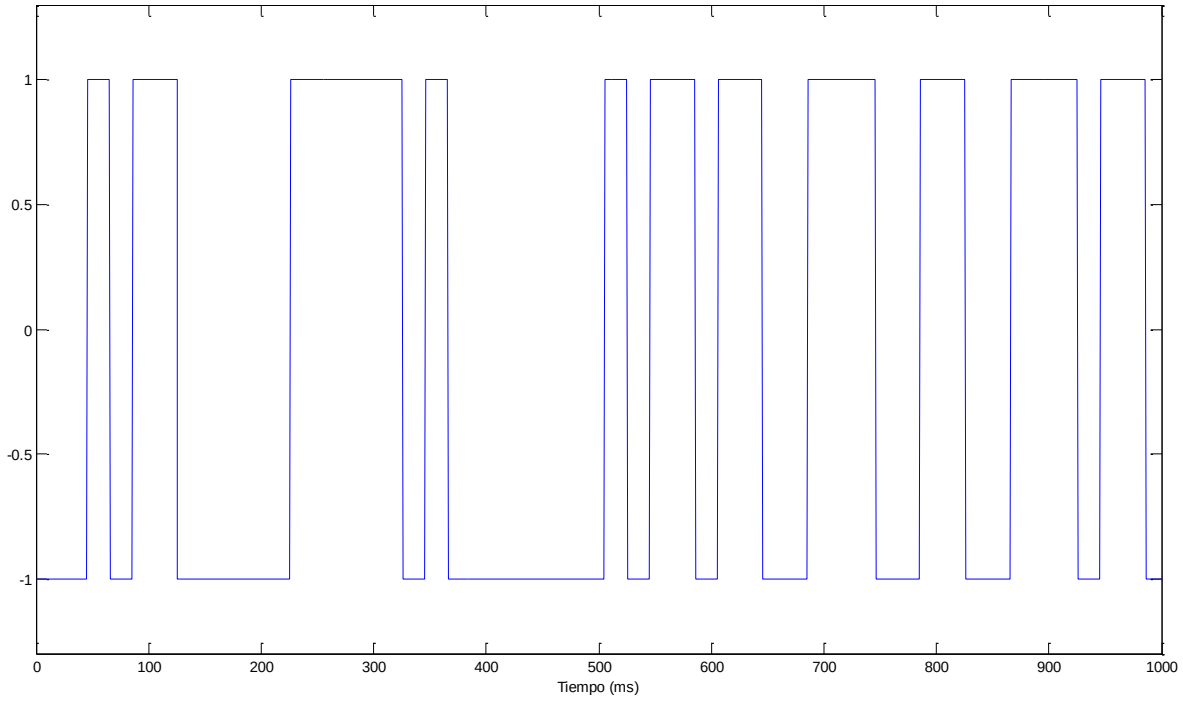

FIGURA 4.19. Salida del bloque de seguimiento para 1 segundo.

4.7.2 Decodificación de los datos de navegación.

La codificación de los datos de navegación sigue un esquema definido en el Documento de Control de Interface GPS, [13].

Cuando los bits de navegación GNSS han sido obtenidos a través de la sincronización de bit, estos deben ser decodificados.

4.7.2.1 Ubicación del preámbulo.

El primer problema en la decodificación de los datos de navegación GPS es determinar la ubicación del comienzo de una subtrama. El comienzo de una subtrama es marcado por un preámbulo de 8 bits de longitud. El patrón del preámbulo es 10001011. Debido a la habilidad del lazo de Costas de seguir la señal con cambios de fase de 180º, este preámbulo puede suceder que sea una versión invertida 01110100. Naturalmente, estos dos patrones de bits pueden ocurrir en cualquier lugar de los datos recibidos, entonces se debe de llevar a cabo un chequeo para autenticar el preámbulo. El procedimiento de autenticación comprueba si el mismo preámbulo se repitió cada 6 segundos correspondiente al tiempo entre la transmisión de dos subtramas consecutivas.

La búsqueda del preámbulo es implementada a través de la correlación. La primer entrada a la función de correlación es la secuencia entrante de bits de los datos de navegación. La segunda entrada a la función de correlación es el preámbulo de 8 bits también representado con -1’s y 1’s. Cuando usamos valores de -1 y 1 en lugar de 0 y 1, la salida de la función de correlación es 160 cuando el preámbulo es localizado y -160 cuando un preámbulo invertido es localizado. Un ejemplo de la correlación entre una secuencia de datos de navegación y el preámbulo puede ser visto en la figura 4.20.

Como se ve en la figura 4.20, la función de correlación da una máxima correlación de 160 varias veces en esta secuencia de 37 segundos. Además del gran numero de valores de correlación máxima, esto también da un valor de correlación mínima de 160 varias veces. Como mencionamos anteriormente, esto significa una instancia invertida del preámbulo ha sido encontrado. El método para distinguir en cuál de los valores de correlación máxima da inicio una subtrama incluye la determinación del retardo entre los valores de correlación máxima consecutivos.

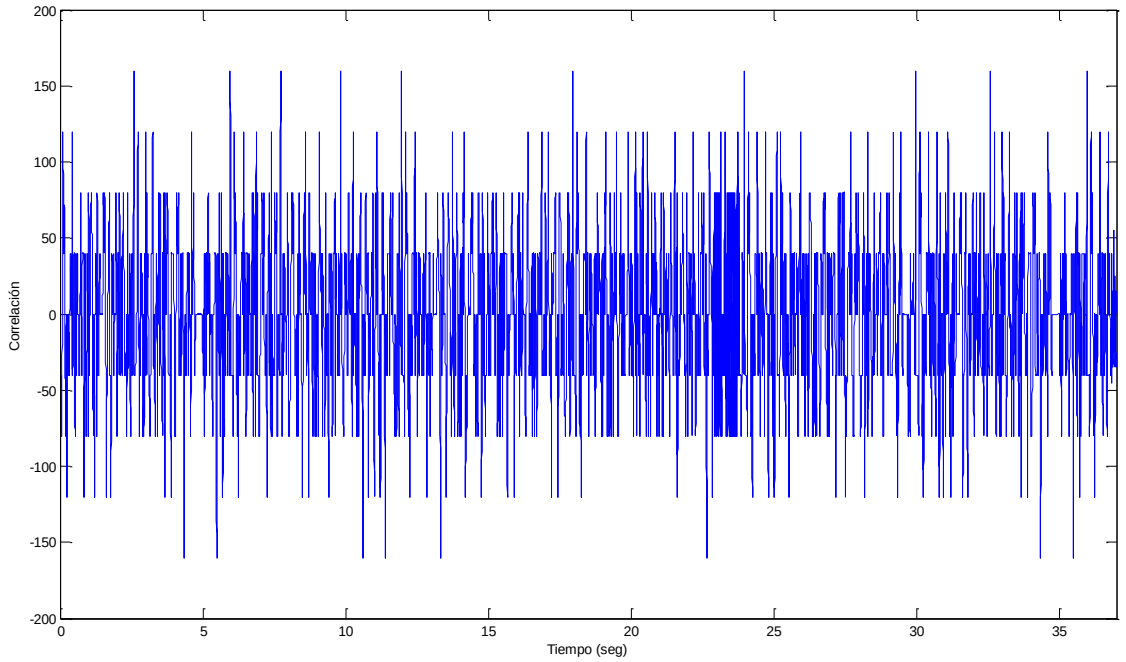

FIGURA 4.20. Correlación entre 37 segundos de datos de navegación y el preámbulo de 8 bits. Los picos indican la ubicación del comienzo de una subtrama.

Solo si el retardo entre los valores de correlación máxima es exactamente 6 segundos y el chequeo de la paridad no falla se estaría asegurando el comienzo de la subtrama indicada [4].

Cuando el preámbulo correcto es localizado, el dato de cada subtrama puede ser extraído. Si la correlación muestra que el preámbulo esta invertido, toda la secuencia de navegación debe ser invertida.

Debido al efecto Doppler la longitud del bit de navegación puede desviarse del valor exacto de 20 ms. Sobre un lapso de tiempo corto esta diferencia de longitud incluso puede acumularse y llegar a ser un valor significativo. Por lo tanto, una mejor solución es buscar un preámbulo en la salida original de 1000 sps del bloque de seguimiento. El algoritmo resulta el mismo, pero cada bit en el patrón del preámbulo de referencia es convertido a 20 valores (muestras). Ahora el pico de la correlación tendrá un valor máximo de $8 \times 20 = 160$ en lugar de 8. En Simultáneamente, este algoritmo modificado también encuentra el tiempo de la transición de bit.

Cabe mencionar que en esta tesis se hace uso del algoritmo mencionado en el párrafo anterior buscando de esta manera el preámbulo en la salida original de 1000 sps que nos proporciona el bloque de seguimiento para que de esta manera se evite cualquier diferencia de longitud del bit que se pudiera presentar, como se menciono anteriormente, todo esto debido al efecto Doppler. Ver la figura 4.20.

4.7.2.2 Extracción de los datos de navegación.

Cada preámbulo correcto marca el principio de una subtrama de los datos de navegación. Cada subtrama contiene 300 bits dividida en 10 palabras de 30 bits. Para mayor detalle referirse a la sección 2.6. La estructura de las primeras dos palabras de una subtrama se muestra en la figura 4.21.

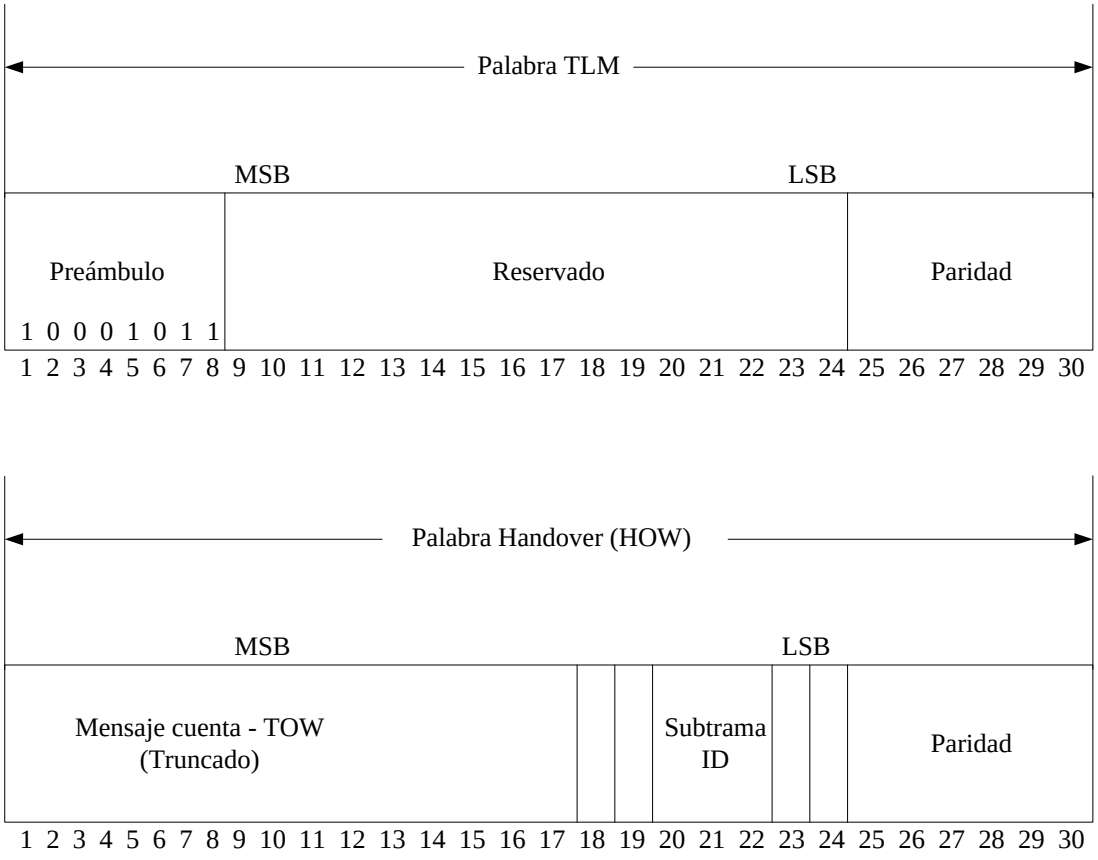

FIGURA 4.21. Las primeras dos palabras de cada subtrama. Estas palabras son conocidas como la palabra telemetría (TLM) y la palabra handover (HOW).

TABLA 4.3 Ecuaciones para la codificación de paridad.

$D_1 = d_1 \oplus D_{30}^*$

$D_2 = d_2 \oplus D_{30}^*$

$D_3 = d_3 \oplus D_{30}^*$

: :

: :

$D_{24} = d_{24} \oplus D_{30}^*$

$D_{25} = D_{29}^* \oplus d_1 \oplus d_2 \oplus d_3 \oplus d_5 \oplus d_6 \oplus d_{10} \oplus d_{11} \oplus d_{12} \oplus d_{13} \oplus d_{14} \oplus d_{17} \oplus d_{18} \oplus d_{20} \oplus d_{23}$

$D_{26} = D_{30}^* \oplus d_2 \oplus d_3 \oplus d_4 \oplus d_6 \oplus d_7 \oplus d_{11} \oplus d_{12} \oplus d_{13} \oplus d_{14} \oplus d_{15} \oplus d_{18} \oplus d_{19} \oplus d_{21} \oplus d_{24}$

$D_{27} = D_{29}^* \oplus d_1 \oplus d_3 \oplus d_4 \oplus d_5 \oplus d_7 \oplus d_8 \oplus d_{12} \oplus d_{13} \oplus d_{14} \oplus d_{15} \oplus d_{16} \oplus d_{19} \oplus d_{20} \oplus d_{22}$

$D_{28} = D_{30}^* \oplus d_2 \oplus d_4 \oplus d_5 \oplus d_6 \oplus d_8 \oplus d_9 \oplus d_{13} \oplus d_{14} \oplus d_{15} \oplus d_{16} \oplus d_{17} \oplus d_{20} \oplus d_{21} \oplus d_{23}$

D_{29}

$= D_{30}^* \oplus d_1 \oplus d_3 \oplus d_5 \oplus d_6 \oplus d_7 \oplus d_9 \oplus d_{10} \oplus d_{14} \oplus d_{15} \oplus d_{16} \oplus d_{17} \oplus d_{18} \oplus d_{21} \oplus d_{22} \oplus d_{24}$

$D_{30} = D_{29}^* \oplus d_3 \oplus d_5 \oplus d_6 \oplus d_8 \oplus d_9 \oplus d_{10} \oplus d_{11} \oplus d_{13} \oplus d_{15} \oplus d_{19} \oplus d_{22} \oplus d_{23} \oplus d_{24}$

Checar Paridad: Además de los 24 bits de datos, cada palabra de 30 bits contiene una paridad de 6 bits. La paridad se utiliza para revisar si hay bits malinterpretados en los datos de navegación. La paridad es calculada a través de las ecuaciones de la tabla 4.3. Aquí el símbolo $\oplus$ denota la operación en modulo 2 o la operación or-exclusiva (xor).

$D_1 - D_{24}$ son los 24 bits de datos en una palabra, mientras $D_{25} - D_{30}$ son los 6 bits de paridad. Los dos bits D_{29}^* y D_{30}^* son los últimos dos bits de paridad de la palabra anterior. Cuando el dato de navegación es recibido, un chequeo de paridad debe ser llevada a cabo para verificar que el dato recibido es interpretado correctamente.

Tiempo de transmisión: Cuando el chequeo de paridad ha sido llevado a cabo satisfactoriamente, el contenido de la secuencia de los datos de navegación puede ser decodificada. La decodificación se hace siguiente el esquema de. [13], que da detalles de cada palabra similar a la que es mostrada en la figura 4.21. El primer tema importante es determinar el tiempo cuando la subtrama actual fue transmitida del satélite GPS.

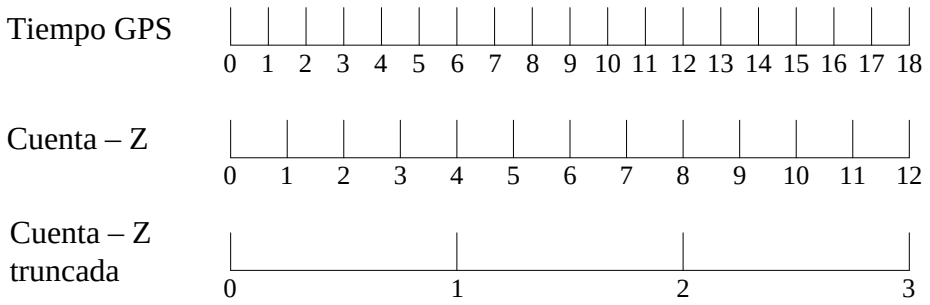

FIGURA 4.22. La relación entre tiempo GPS, cuenta – Z, y la cuenta – Z truncada en la palabra HOW de los datos de navegación.

La segunda palabra de cada subtrama es llamada HOW que incluye una versión truncada de la TOW. Este número es conocido como cuenta – Z. La cuenta – Z es el número de segundos que han pasado desde el "*rollover*" de la última semana GPS en unidades de 1.5 segundos. El "*rollover*" ocurre a media noche entre Sábado y Domingo. El valor máximo de la cuenta – Z es 403,199 ya que una semana tiene 604,800 segundos y 604,800 s/1.5 = 403,200 s.

El valor de la cuenta – Z en la palabra HOW es una versión truncada conteniendo solo los 17 bits más significativos (MSB). Este truncamiento hace aumentar la cuenta – Z en pasos de 6 segundos correspondientes al tiempo entre la transmisión de dos subtramas de navegación completas. La figura 4.22 muestra la relación entre las tres cantidades de tiempo; tiempo GPS, cuenta – Z, y cuenta – Z truncada.

El valor de la cuenta – Z truncada en la palabra HOW corresponde al tiempo de transmisión de la siguiente subtrama de los datos de navegación. Para tener el tiempo de transmisión de la subtrama actual, la cuenta – Z truncada será multiplicada por 6 y 6 segundos serán subtruncados para obtener el resultado.

Parámetros restantes: Los parámetros restantes de los datos de navegación son también decodificados como se menciona en el documento [13]. En el completo a dos, el signo del bit (+ o -) ocupa en MSB. La unidad de semicírculo es multiplicada con π para ser convertida a unidades de radianes.

El parámetro IODE es el nombre corto de Efemérides de Emisión de Datos (IODE – *Issue of Data Ephemerides*). IODE es un número de 8 bits que únicamente

identifica el conjunto de datos. Todos los parámetros están listados en la tabla 4.4 y para ver detalles de estos parámetros es necesario revisar el algoritmo de cálculo de posición el cual no se menciona en esta tesis. Los parámetros de efemérides son decodificados como se muestra en la tabla 4.5 y 4.6.

TABLA 4.4 Parámetros de efemérides.

IODE	Emisión de dato, efemérides
Δn	Significa corrección de movimiento
μ_0	Significa anomalía de t_{oe}
e	Excentricidad
$\sqrt{a}$	Raíz cuadrada del eje semi-mayor
t_{oe}	Época de referencia de efemérides
Ω_0	Longitud de nodo ascendente a t_{oe}
i_0	Inclinación de t_{oe}
ω	Argumento de perigeo
$\dot{\Omega}$	Velocidad de Ω_0
IDOT	Velocidad de i_0
C_{rs}, C_{rc}	Coeficiente de corrección para los términos seno y coseno de r
C_{is}, C_{ic}	Coeficiente de corrección para los términos seno y coseno de i
C_{us}, C_{uc}	Coeficiente de corrección para los términos seno y coseno de ω

TABLA 4.5 Esquema de decodificación para los parámetros de efemérides GPS en los datos de navegación. n^* significa que los bits actuales n serán decodificados usando el complemento a dos.

Parámetro	**Número de bits**	**Factor de escala (LSB)**	**Unidades**
IODE	8		
C_{rs}	16^*	2^{-5}	m
Δn	16^*	2^{-43}	Semicírculo/seg
μ_0	32^*	2^{-31}	Semicírculo
C_{uc}	16^*	2^{-29}	Radianes
e	32	2^{-33}	Adimensional
C_{us}	16^*	2^{-29}	Radianes
$\sqrt{a}$	32	2^{-19}	$m^{1/2}$
t_{oe}	16	2^4	Segundos
C_{ic}	16^*	2^{-29}	Radianes
Ω_0	32^*	2^{-31}	Semicírculo
C_{is}	16^*	2^{-29}	Radianes
i_0	32^*	2^{-31}	Semicírculo
C_{rc}	16^*	2^{-5}	m
ω	32^*	2^{-31}	Semicírculo
$\dot{\Omega}$	24^*	2^{-43}	Semicírculo/seg
IDOT	14^*	2^{-43}	Semicírculo/seg

TABLA 4.6 Parámetros de la subtrama 1.

Parámetro	**Número de bits**	**Factor de escala (LSB)**	**Unidades**
Código sobre L2	2	1	
Número de semana	10	1	semana
Bandera de dato P L2	1	1	
Precisión SV	4		Ver [14]
Salud SV	6	1	
T_{GD}	8^*	2^{-31}	Segundos
IODC	10		Ver [14]
t_{oc}	16	2^4	Segundos
a_{f2}	8^*	2^{-55}	seg/seg^2
a_{f1}	16^*	2^{-43}	seg/seg
a_{f0}	22^*	2^{-31}	Segundos

Capítulo 5

Arquitectura propuesta de un receptor GPS basado en software

En este capítulo se presenta la arquitectura de un receptor GPS basado en software. Describiremos cada uno de los detalles mostrados en la figura 5.1, desde la antena GPS, posteriormente pasando a la terminal de entrada. Después explicaremos la aplicación encargada de grabar las tramas GPS sobre la computadora y el uso de software de procesamiento de señales para poder llevar el algoritmo implementado (lazo de seguimiento) sobre el microprocesador de un FPGA. Y por último se darán los detalles del kit de desarrollo XtremeDSP Pro de Nallatech.

La figura 5.1 muestra los detalles de la arquitectura propuesta para un receptor de señales GPS basado en software, actualmente se está trabajando en los algoritmos de adquisición, lazo de seguimiento y decodificación de los datos de navegación, estos dos últimos algoritmos presentados en este trabajo de tesis, para más detalles ver capitulo 4.

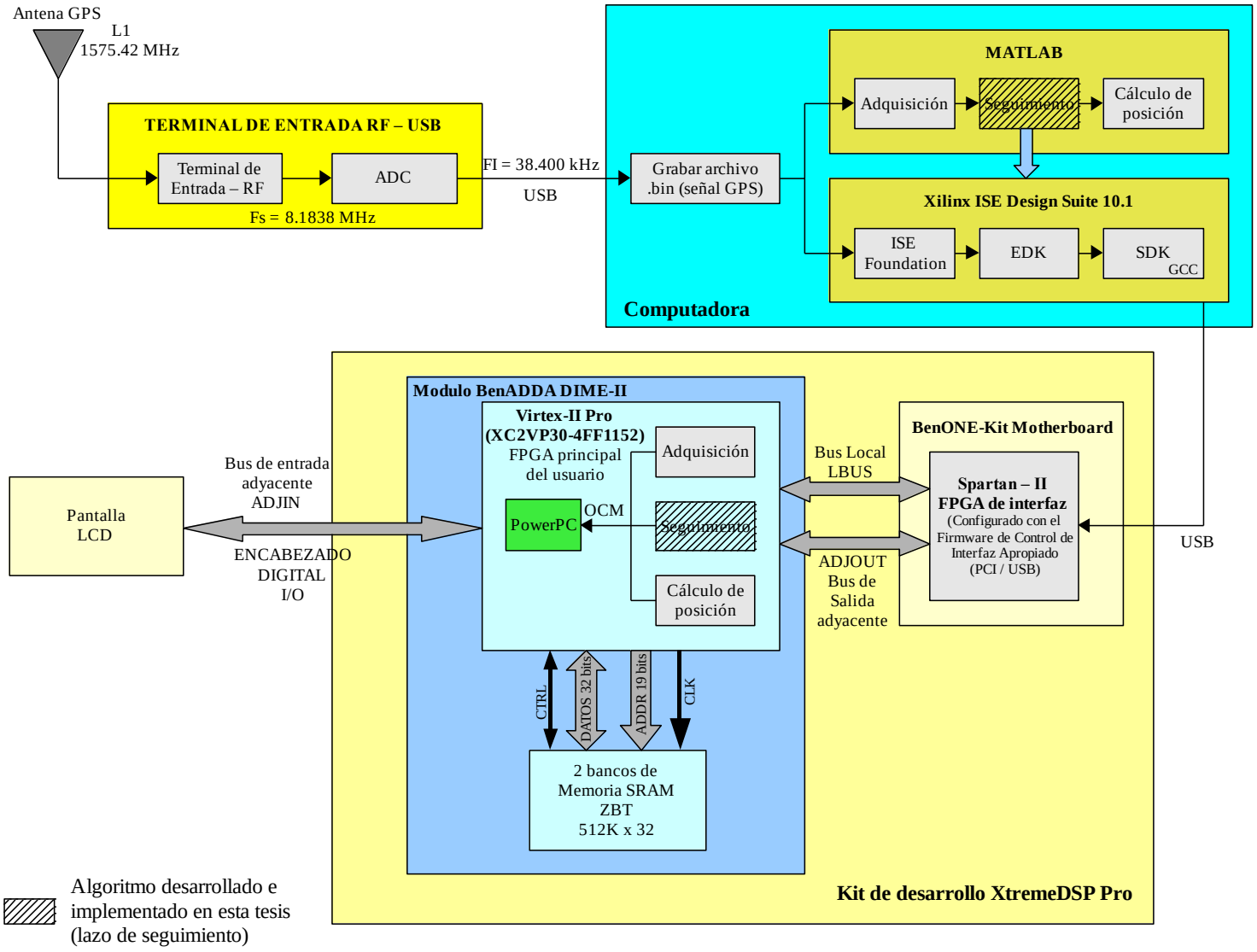

FIGURA 5.1. Arquitectura propuesta de un receptor de señales GPS basado en software

5.1 Antena.

La antena típicamente no se considerada parte del diseño de la terminal de entrada (front–end), pero es el primer componente en la trayectoria de la señal y dicta los elementos que siguen.

La antena es diseñada para inducir un voltaje de la propagación de ondas de radio en la frecuencia L1 o 1575.42 MHz. Además, el diseño debe tener la capacidad de un ancho de banda apropiado de la señal deseada. Esto se hace usualmente usando dos parámetros adicionales en la antena: la Razón de Voltaje de Onda Estacionaria (VSWR – *Voltage Standing Wave Ratio*) y la impedancia. Prácticamente todos los componentes de la terminal de entrada GNSS utilizan una impedancia de 50 Ω, qué es típico para la mayoría de las frecuencias de radio. VSWR es una medida de desacoplamiento de impedancia o la medida de que tanto la potencia incidente será absorbida y que tanto será reflejada. Y, por supuesto, esto es una función de frecuencia. El VSWR está típicamente sobre el orden de 2:1, lo que equivale al 90% de absorción de energía en todo el ancho de banda de la frecuencia deseada.

Características de la antena:

- Ganancia de 26 dB
- VSWR < 2.0
- Voltaje: 3.3 V +/- 0.5 V
- Corriente: 12 mA
- Peso: 50 gr.
- 5 metros de cable

5.2 Terminal de entrada RF – USB (GN3Sv2).

Este dispositivo no nos dará la solución de una posición pura y simple como los demás módulos GPS. En lugar de eso, el GN3S es diseñado para capturar directamente los

datos de la señal de bajo nivel (muestras crudas de frecuencia intermedia) que será entregada por la red de satélites GPS y será procesada por la terminal de entrada de RF SiGe. Los datos capturados pueden ser procesados usando Matlab o lenguaje C.

El programa de captura desarrollado para la plataforma de Windows XP tiene un límite de 600 MB (o 38.4 segundos). Esto significa que se puede capturar un registro completo de GPS. En este momento, la captura de archivos más grandes no es posible con el código de la aplicación actual.

La arquitectura propuesta utiliza la segunda versión del GN3S (GN3S v2.0) la cual está construida sobre el SiGe 4120 GPS ASIC. Este dispositivo proporciona un flujo de datos con una frecuencia de muestreo baja (8.1838 MHz) y un par de muestras I/Q. La figura 5.2 muestra antena y la terminal de entrada de nuestro receptor de señales GPS [48]. Los parámetros específicos de los datos capturados de este modulo son los siguientes:

- Frecuencia de muestreo: 8.1838 MHz
- Frecuencia intermedia: 38.400 kHz
- 2 bits de muestras I/Q (1 bit para I y 1 bit para Q) en un formato binario schar[2]

  (sI0, sQ0, sI1, sQ1, sI2, sQ2, …)

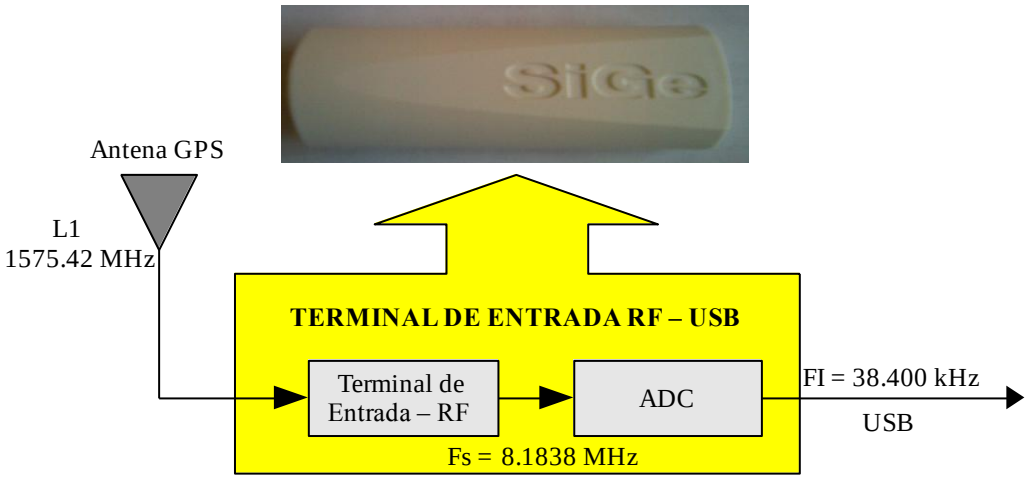

FIGURA 5.2. Antena y terminal de entrada dentro de la arquitectura del receptor GPS propuesto.

[2] schar: Es un tipo de dato carácter. Carácter con signo de 8 bits.

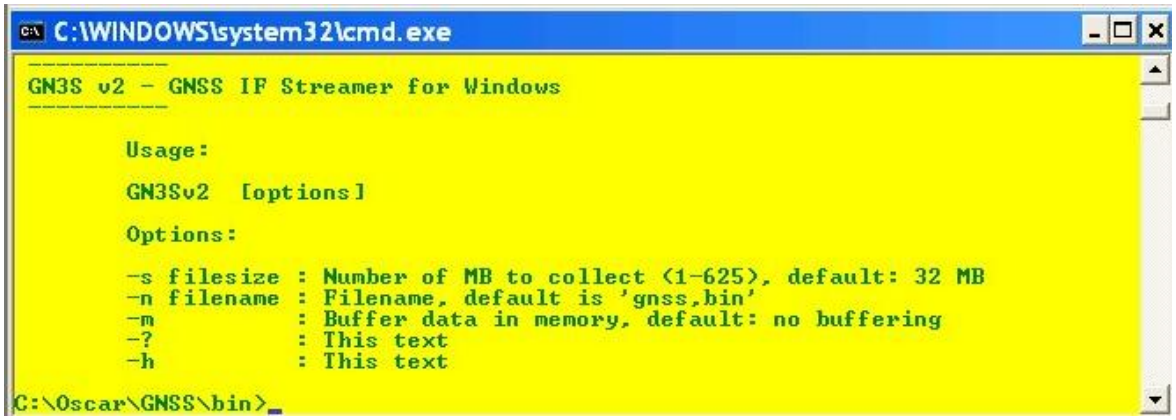

FIGURA 5.3. Línea de comandos disponibles dentro de la aplicación GN3S

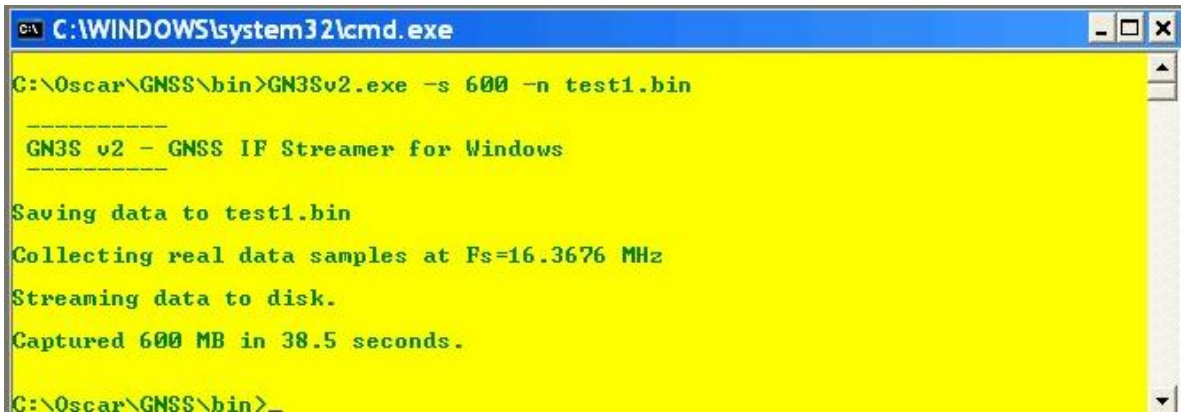

FIGURA 5.4. Aplicación para grabar los datos de la señal GPS en un archivo con extensión .bin

5.3 Computadora

Dentro de la computadora se llevan a cabo varias tareas, la primera es grabar un archivo con extensión *.bin con los datos de la señal GPS; para esto se siguen los pasos mostrados en la figura 5.3 y 5.4. Una vez ya teniendo el archivo con la señal GPS, se procede a realizar el procesamiento de la señal en Matlab y en lenguaje C para poder obtener el cálculo de la posición. Si el procesamiento lo realizamos en Matlab el proceso es el mostrado en la figura 5.5 en donde todos los algoritmos se realizan sobre la computadora y podemos ver la posición donde se coloco nuestra antena GPS y obtuvimos la señal (grabación del archivo *.bin).

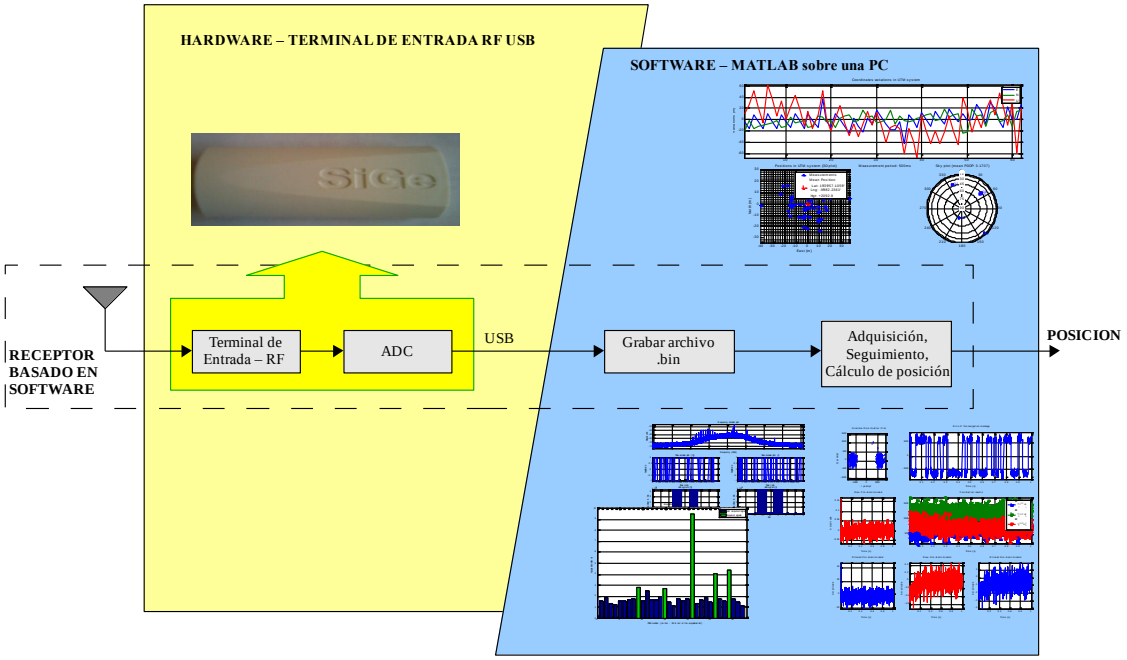

FIGURA 5.5. Procesamiento de la señal GPS en Matlab.

También se hace el uso de herramientas de software de Xilinx como se muestra en la figura 5.1. Pero esto se hace únicamente para implementar los algoritmos ya desarrollados en Matlab sobre un microprocesador PowerPC 405 que viene incrustado dentro del FPGA Virtex –II Pro de nuestro Kit de desarrollo XtremeDSP Pro. Para esto, es necesario implementar el algoritmo en lenguaje C nativo, ya que el software SDK cuenta con un compilador GCC que nos ayuda a poder implementar cualquier aplicación que uno quiera dentro del microprocesador PowerPC 405 del FPGA, para esto es necesario revisar el apéndice C donde se muestra un manual detallado sobre la implementación de nuestra aplicación utilizando las herramientas de software de Xilinx.

5.4 Kit de desarrollo XtremeDSP Pro

El kit de desarrollo XtremeDSP Pro mostrado en la figura 5.6 sirve como una plataforma de desarrollo ideal para la tecnología de FPGA Virtex – II Pro y proporciona una entrada escalable hacia el sistema DIME – II de Nallatech. Sus dos canal de alto rendimiento ADC y DAC, así como el dispositivo Virtex – II Pro programable por el

usuario son características ideales para implementar aplicaciones de alto rendimiento en procesamiento de señales como son Radio Definido por Software, 3G Wireless, Networking, HDTV o Video e Imagen [40].

5.4.1 Características principales del Kit de desarrollo XtremDSP Pro.

La tarjeta de desarrollo XtremeDSP se compone de una tarjeta madre poblada con un modulo (tarjeta hija). A la tarjeta madre se le llama "Tarjeta madre del kit BenONE" y el modulo es llamado como "Modulo BenADDA DIME – II".

- ✓ Tarjeta madre del Kit BenONE.
  - Apoya el suministro solo para el modulo BenADDA DIME – II.
  - FPGA Spartan –II para 3.3V/5V o interfaz USB.
  - LEDs de estado
  - Configuración de cabeceras JTAG.
- ✓ Modulo BenADDA DIME – II.
  - FPGA de usuario Virtex – II Pro: XC2VP30 – 4FF1152
  - 2 canales ADC independientes: ADC AD6645 (14 bits hasta 105 MSPS)
  - 2 canales DAC independientes: DAC AD9772 (14 bits hasta 169 MSPS)
  - Soporte para reloj externo, oscilador y relojes programables.
  - Dos bancos de ZBT – SRAM (133 MHz, 512x32 bits por banco).
  - Opciones de múltiple sincronización: interna y externa.
  - LEDs de estado.

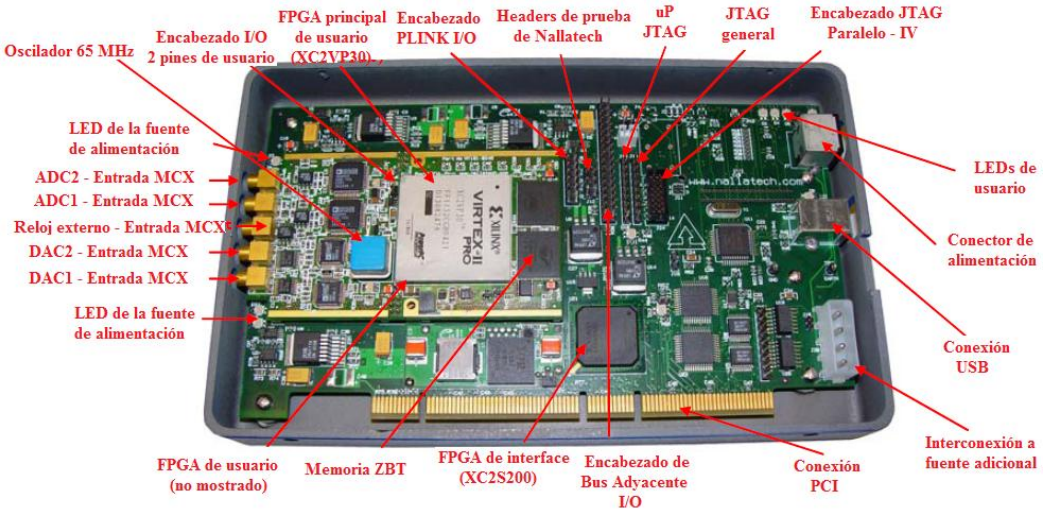

FIGURA 5.6. Kit de desarrollo XtremeDSP Pro (Vista Frontal).

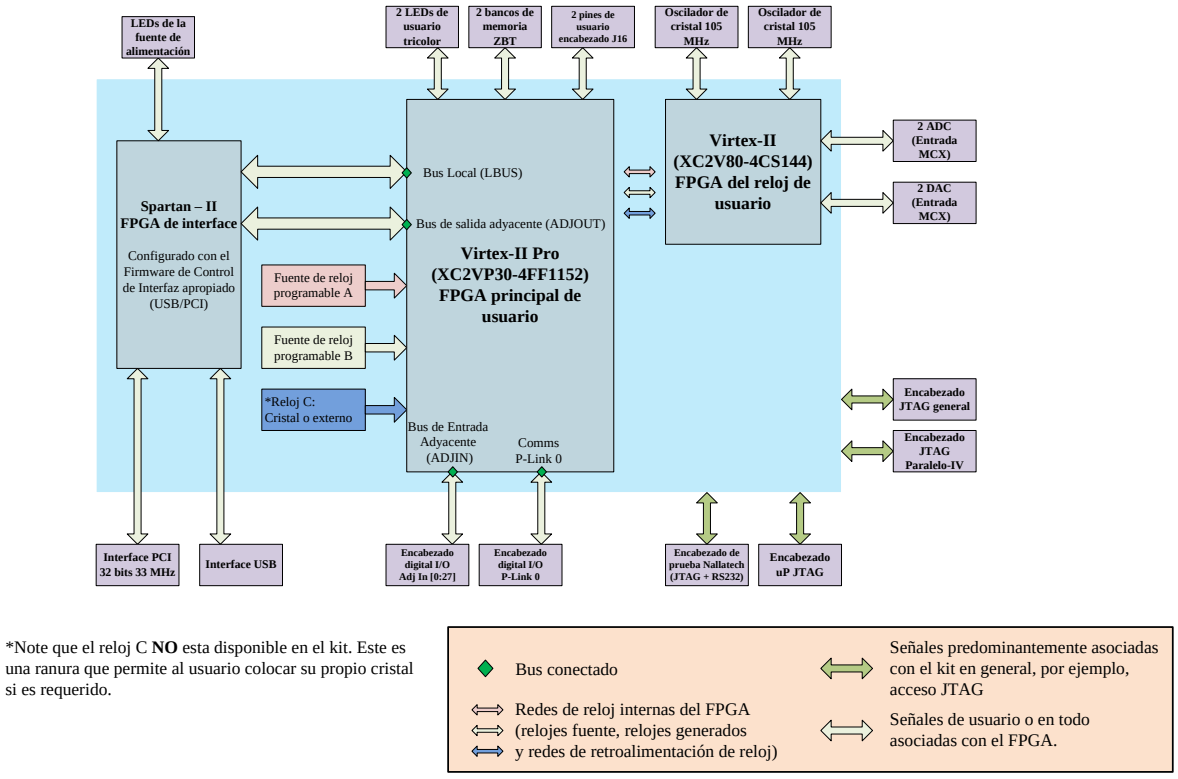

FIGURA 5.7. Diagrama a bloques principal del Kit de desarrollo XtremeDSP Pro.

5.4.2 Diagrama funcional del Kit de desarrollo XtremDSP Pro.

El Kit de desarrollo XtremeDSP se caracteriza por tener tres FPGAs de Xilinx; un FPGA de usuario Virtex – II Pro, un Virtex – II Pro para administración del reloj y un Spartan – II para la interface con el FPGA como se muestra en la figura 5.7. El dispositivo Virtex – II está disponible exclusivamente para el usuario mientras que el Spartan – II proporciona la pre-configuración con firmware para la interfaz PCI/USB. La interfaz del FPGA se comunica directamente con el FPGA de usuario (XC2VP30-4FF1152) por medio de una comunicación en bus dedicada que es hecha por los buses LBUS y ADJOUT.

El dispositivo Virtex – II Pro XC2VP30-4FF1152 está diseñado para ser usado para la parte principal del diseño de los usuarios. El Virtex – II XC2V80-4CS144 es diseñado para ser usado como un dispositivo de configuración del reloj en un diseño [40].

El dispositivo Virtex – II Pro XC2VP30-4FF1152 esta acoplado al Kit con la tarea de trabajar las comunicaciones entre los diferentes dispositivos dentro del Kit XtremeDSP. Entonces este FPGA Virtex – II Pro, comunica a estos elementos, que pueden ser dispositivos de almacenamiento, de salida o entrada o otros FPGA's. Una vez nuestros

datos han pasado por nuestro distribuidor, a partir del registro OPB llega al FPGA de usuario.

Con este FPGA Virtex II Pro podemos desarrollar nuestra tarea de procesamiento de señales. Con su núcleo PowerPC (PPC), es posible conseguir los resultados a partir de la etapa interna en el FPGS BenADDA DIME II, el cual fue desarrollada exclusivamente para el PDS. Nuestros datos en el FPGA de usuario son mandados al PPC a través del registro ILMB, para trabajar las etapas de las señales GPS.

Las tareas desarrolladas en el PPC y el FPGA de usuario son tres; primeramente se encuentra la etapa de adquisición, la cual al igual que las etapas posteriores se desarrollan utilizando las ventajas que brinda el FPGA y el uso de su núcleo. Las etapas posteriores son las de seguimiento y las del cálculo de posición de nuestro receptor, que son entrelazadas entre sí, vía software, utilizando nuestro bloque de memoria el cual es de 512 x 32, esto con el registro ADDR.

Finalmente necesitamos exhibir nuestros resultados y esto por medio de una etapa de visualización, la cual ya es externa a nuestra tarjeta de trabajo. Sin embargo puede realizarse utilizando, desde una pantalla LCD, hasta procesos un poco más complicados, utilizando nuestros encabezados de salida o en su caso el protocolo UARTLITE a través del registro GPIO.

Capítulo 6

Diseño e Implementación.

En este capítulo se presenta la implementación de la arquitectura mostrada en la figura 5.1 y se muestran los resultados obtenidos de la implementación del algoritmo de seguimiento de la señal GPS realizado en Matlab y en lenguaje C. Por último, daremos los detalles de la implementación del algoritmo de seguimiento de señales GPS sobre el microprocesador PowerPC 405 de nuestro kit de desarrollo XtremDSP Pro.

6.1 Implementación del algoritmo de seguimiento en Matlab y en lenguaje C.

Como se vio en el capítulo 4 el lazo de seguimiento de la señal GPS se divide en dos partes: lazo de seguimiento de código y lazo de seguimiento de portadora. Todo esto con la finalidad de demodulador los datos de navegación y obtener un cálculo de posición correcto. Para su implementación tanto en Matlab como en lenguaje C se sigue el diagrama a bloques mostrado en la figura 6.1.

Este diagrama a bloques del código, está diseñado para "seguir" todas las señales GPS asignadas de cada uno de los satélites adquiridos.

La función toma los siguientes parámetros: un bloque de la señal grabada de la terminal de entrada, una estructura llamada `canal`, tablas de códigos C/A y generación de señales seno y coseno.

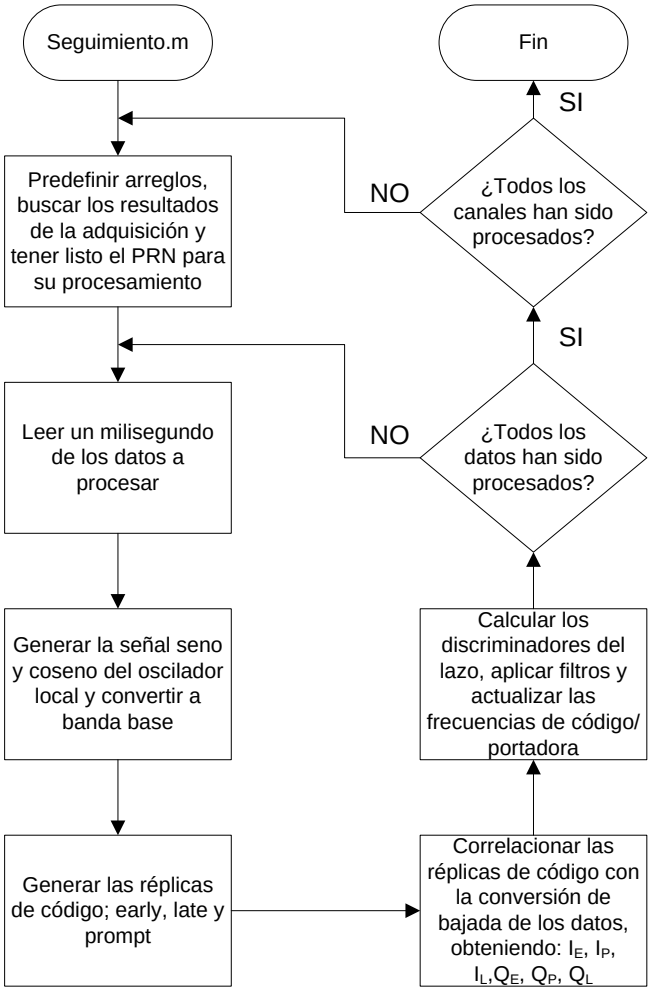

FIGURA 6.1. Diagrama de flujo del seguimiento de un GNSS

La función procesa el bloque de las muestras y regresa dos estructuras: los resultados del seguimiento guardados en la estructura *trackResults* y una actualización de la estructura *canal*.

La estructura *canal* es usada para pasar información de cada canal y es también usada para almacenar información del canal actual. El segundo propósito de esta estructura es hacer un seguimiento continuo. De esta manera, el procesamiento de dos o más bloques señales pueden ser realizado continuamente. La estructura contiene información actual (de los últimos milisegundos procesados) de la frecuencia de portadora, fase de código, numero de PRN del satélite seguido, valores temporales de los filtros del lazo, y generador de señal local.

Los parámetros del lazo de seguimiento se encuentran en la estructura *settings*:

PLL_dampingRatio	Factor de amortiguamiento
PLL_noiseBandwidth	Ancho de banda de ruido del PLL

La siguiente lista muestra las variables específicas del seguimiento de código en la estructura *settings*:

DLL_CACorrelatorSpacing	Espacio entre los correladores early y late, unidades de chip
DLL_dampingRatio	Factor de amortiguamiento para el lazo de seguimiento de código
PLL_noiseBandwidth	Ancho de banda de ruido del DLL

La estructura *trackResults* es la salida principal de la función de seguimiento. Esta estructura contiene el resultado de todos los canales y para cada milisegundo del bloque procesado: información acerca de las propiedades de la señal (frecuencia portadora y fase de código) y la salida de los seis correladores y del discriminador de lazo.

La salida del seguimiento de código es usada como entrada para el siguiente bloque del receptor GPS que sería la extracción de los datos de navegación.

Sabemos que la finalidad del lazo de seguimiento es demodular la señal de entrada para obtener el mensaje de navegación. Algunas simulaciones importantes ya fueron mostradas en el capítulo 4, en este capítulo mostraremos los detalles de lo que obtenemos a la salida del lazo de seguimiento.

En el capítulo 4 se mencionaron dos tipos de discriminadores para el lazo de seguimiento de código: discriminador coherente y discriminador no coherente. Estos mostrados en la tabla 4.2. La potencia de la señal no está en la rama en fase, si el PLL no se encuentra en estado de bloqueo sobre la portadora; en estos casos es necesario seguir la fase de código en la rama de cuadratura, el diagrama mostrado en la figura 4.12 da todos estos detalles.

La figura 6.2 muestra la salida de los 6 correlacionadores, cuando el PLL está en estado de bloqueo. Esto es suficiente para usar solo los correlacionadores de la rama en fase del discriminador DLL de la ecuación 4.25.

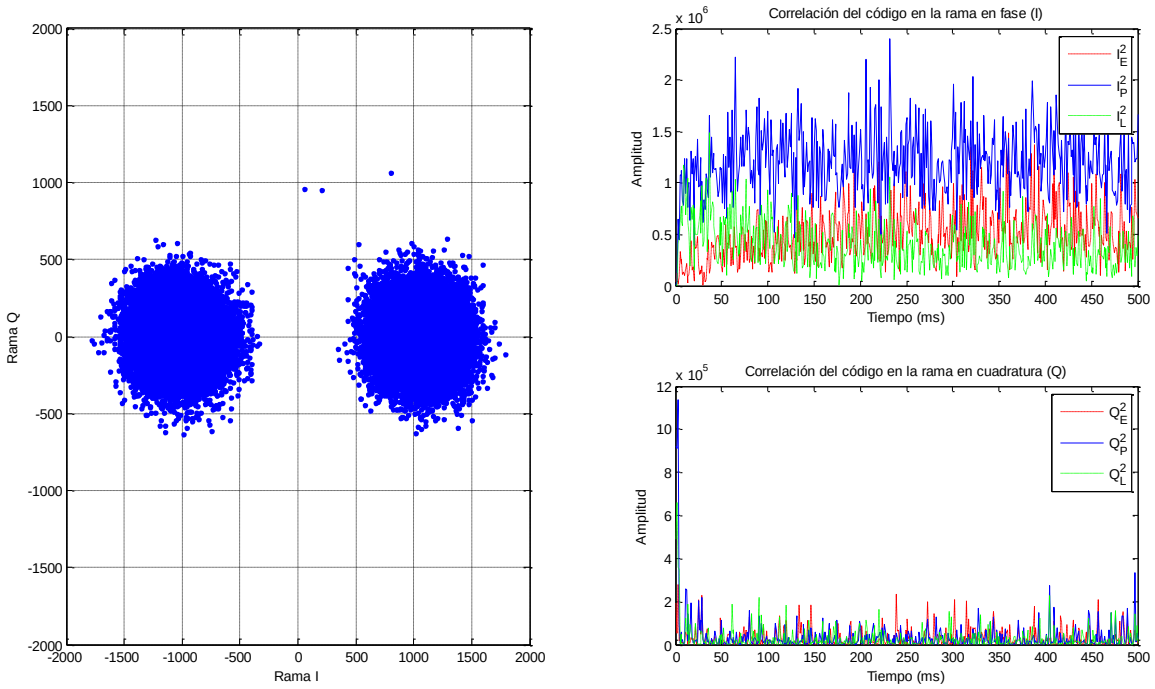

FIGURA 6.2. Salida de los seis correlacionadores en la rama en fase y en cuadratura del lazo de seguimiento cuando el PLL se encuentra en estado de bloqueo.

Por otra parte la figura 6.3 muestra la salida de los 6 correlacionadores, cuando el PLL no se encuentra en estado de bloqueo (esto no es lo ideal).

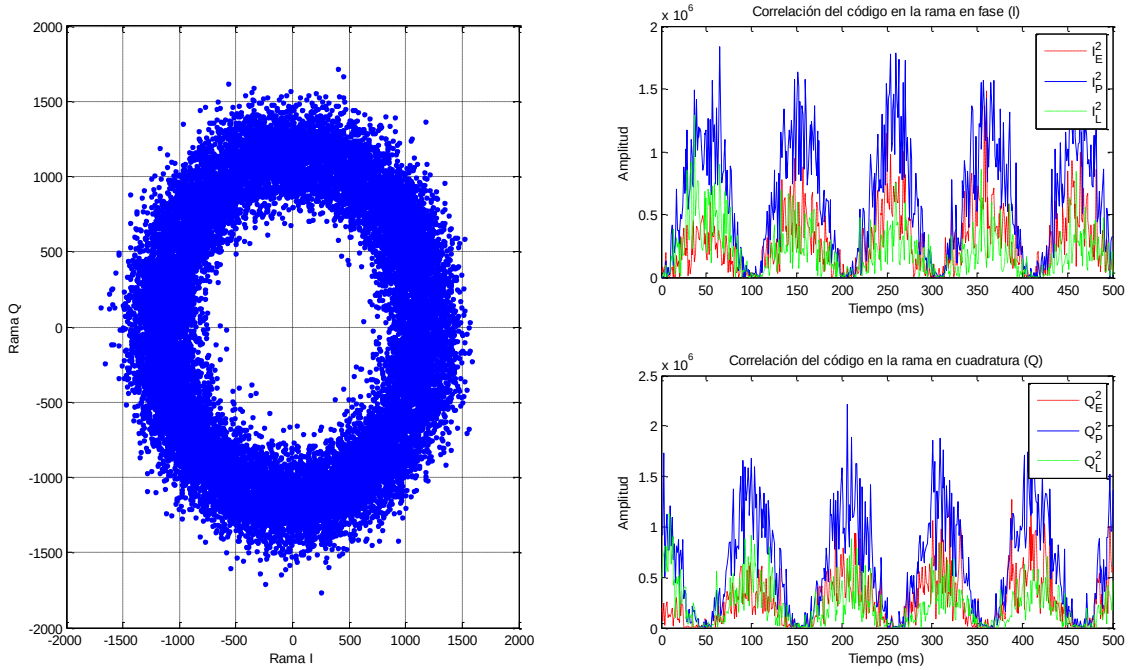

FIGURA 6.3. Salida de los seis correlacionadores en la rama en fase y en cuadratura del lazo de seguimiento cuando el PLL no se encuentra en estado de bloqueo.

6.1.1 Bits de navegación.

El lazo de seguimiento produce un valor de los datos de navegación cada milisegundo. La longitud de un bit de navegación es de 20 ms (GPS); si el efecto Doppler es 0. Para poder decodificar los datos de navegación es necesario leer el valor de cada bit en el mensaje de navegación. La figura 6.4 muestra el mensaje de navegación generado a la salida del bloque de seguimiento.

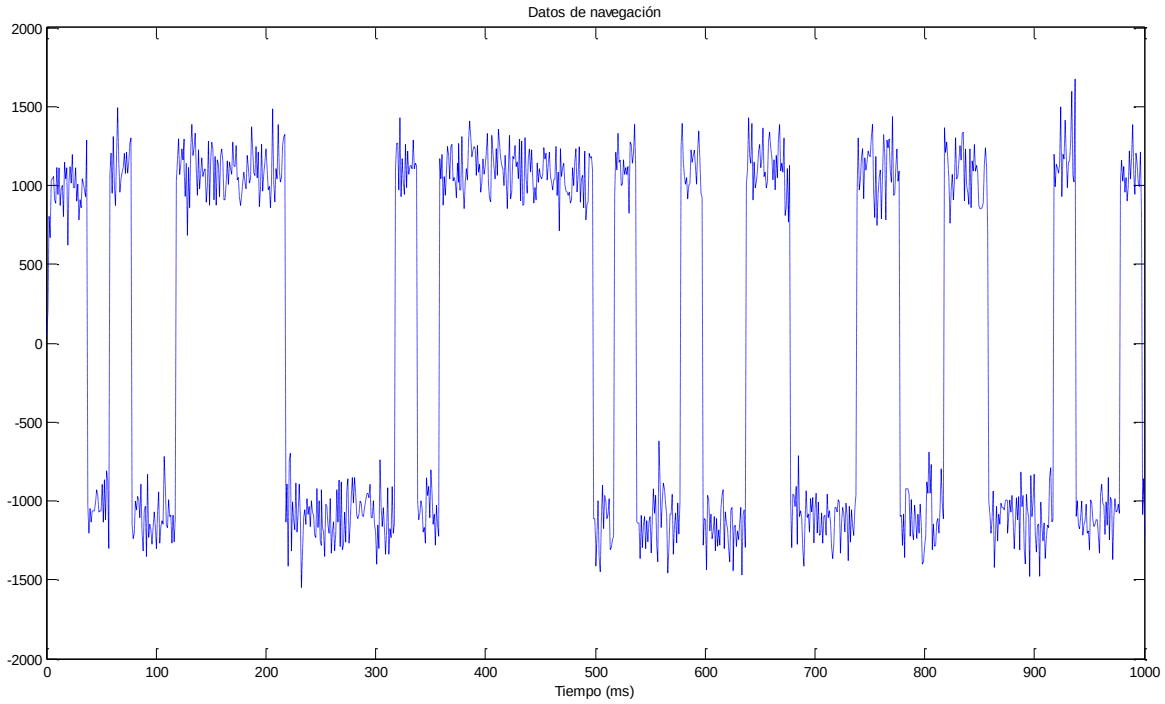

FIGURA 6.4. Mensaje de navegación tal y como es generado por el lazo de seguimiento.

Las señales ruidosas presentan dos problemas principalmente mientras se están leyendo el valor de los bits:

- El valor del bit es corrompido por el ruido.
- La frontera del bit debe ser encontrado.

El valor del bit para señales ruidosas puede ser encontrado haciendo una suma (sin necesidad de un promedio) de varias “muestras de bits”, si las fronteras de los bit es conocidas.

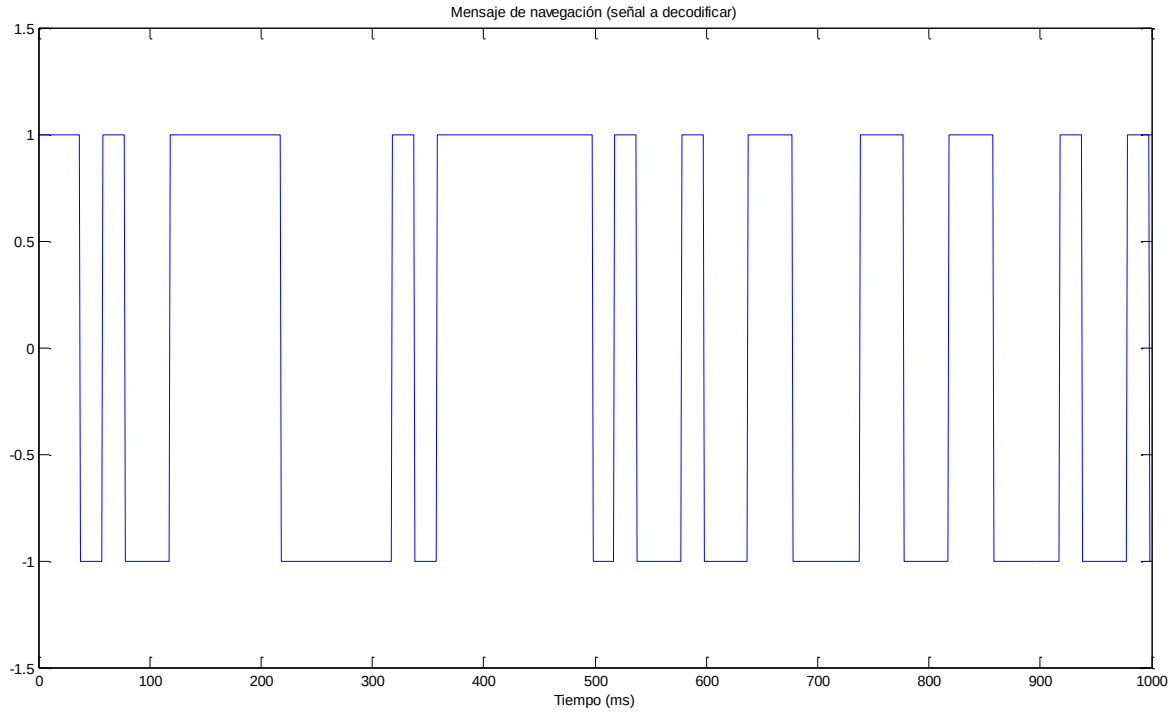

FIGURA 6.5. Salida del bloque de seguimiento truncado a valores de 1 y -1. Mensaje de navegación a decodificar.

6.1.2 Sincronización del bit.

La salida del lazo de seguimiento es el valor de la rama en fase del bloque de seguimiento truncado a valores de -1 y 1, la figura 6.5 muestra este truncamiento para 1 segundo. El bloque de seguimiento proporciona esta salida cada ms. Como se menciono en el capítulo 4, la velocidad de bit de los datos de navegación es de 50 bps. La velocidad de bit del bloque de seguimiento es de 1000 bps correspondiendo a un valor cada ms. Antes de que los datos de navegación puedan ser decodificados, la señal del bloque de seguimiento debe convertirse de 1000 bps a 50 bps. Esto es, veinte valores consecutivos serán reemplazados por solo un valor. A este proceso de conversión se le conoce como sincronización del bit.

6.1.2.1 Encontrando el tiempo de la transición del bit.

La primera tarea del procedimiento de sincronización de bit es encontrar el tiempo en una secuencia donde ocurre la transición de bit. Primero, se debe detectar los cruces por cero. Un cruce por cero es donde la salida cambia de 1 a -1, o viceversa. Cuando un cruce por cero es localizado, el tiempo de una transición de bit es localizado.

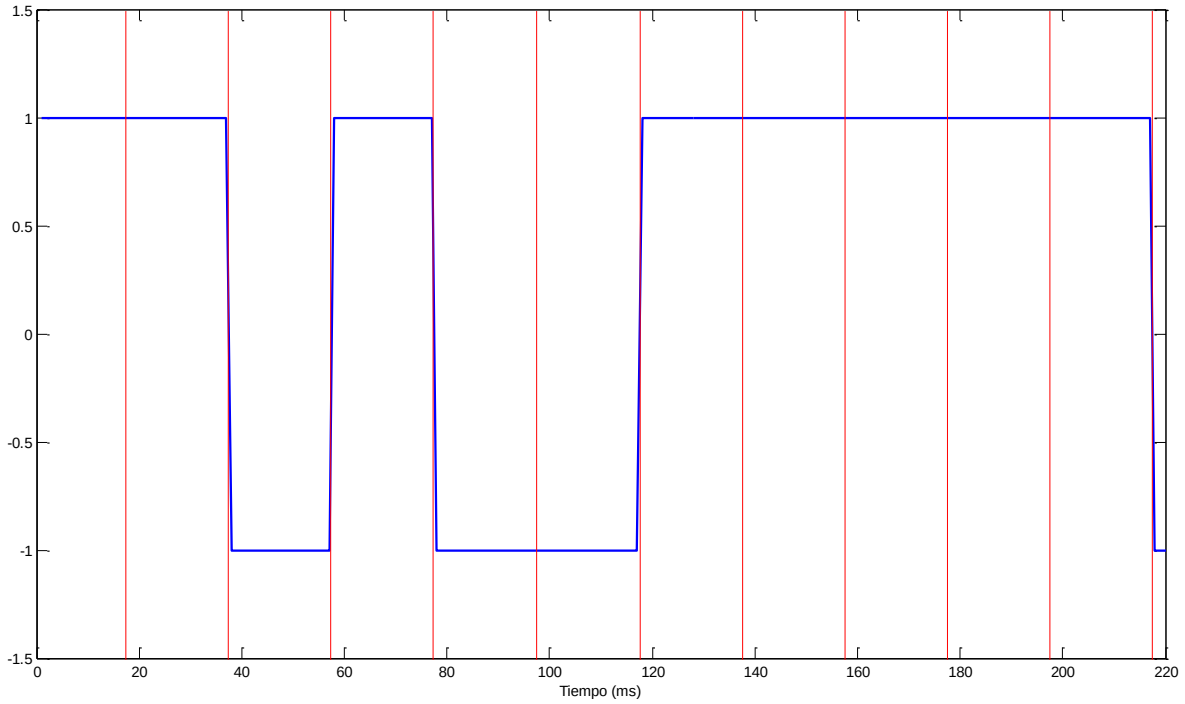

FIGURA 6.6. Tiempo de transición del bit. La línea azul es la salida del bloque de seguimiento. La línea roja marca la transición del bit separado 20 ms.

Cuando el tiempo de una transición de bit es conocido, es posible encontrar todos los tiempos de la transición de bit. En la figura 6.6 muestra como todos los tiempos de la transición del bit son encontrados en una secuencia de 200 ms. La transición del bit es marcado por las líneas rojas.

6.1.2.2 Encontrando los valores del bit.

Cuando el tiempo de la transición del bit ha sido encontrado, la señal de 1000 bps debe ser convertida a la velocidad de bit de 50 bps. Para ello, 20 muestras deben ser remplazadas por solo un valor. El método para encontrar este valor es por medio de un cálculo simple. Un promedio es calculado para las veinte muestras y los valores de salida son encontrados de acuerdo a la tabla 6.1.

TABLA 6.1. Salida de la sincronización del bit como una función del valor promedio calculado

Promedio	**Salida**
≤ 0.9	-1
	Error en la sincronización del bit
≥ 0.9	1

Como podemos ver en la tabla 6.1 el algoritmo deja uno de los veinte valores que es diferente a los 19 restantes sin tener un error. La razón para esto es, que el tiempo de la transición de bit puede mover una muestra de un tiempo a otro. Esto es debido a variaciones que se pueden presentar en el reloj de muestreo [4].

6.1.3 Decodificación de los datos de navegación.

La figura 2.12 muestra el contenido de los datos de navegación. El mensaje de navegación contiene información del satélite. Las subtramas 1 a 3 son necesarias para calcular la posición satelital. Para hacer una demodulación correcta del mensaje de navegación se tienen que seguir lo que se menciona en la sección 4.7.2. Resumiendo tenemos lo siguiente:

- Como primer punto tenemos que encontrar las subtramas en el mensaje de navegación.
- Localizar el preámbulo (palabra TLM) [10001011]
- Representar la secuencia de datos con 1 y -1
- Calcular la correlación con el preámbulo [1 -1 -1 -1 1 -1 1 1]
- Buscar los picos de 160 y -160 en la grafica de correlación, ver figura 6.7

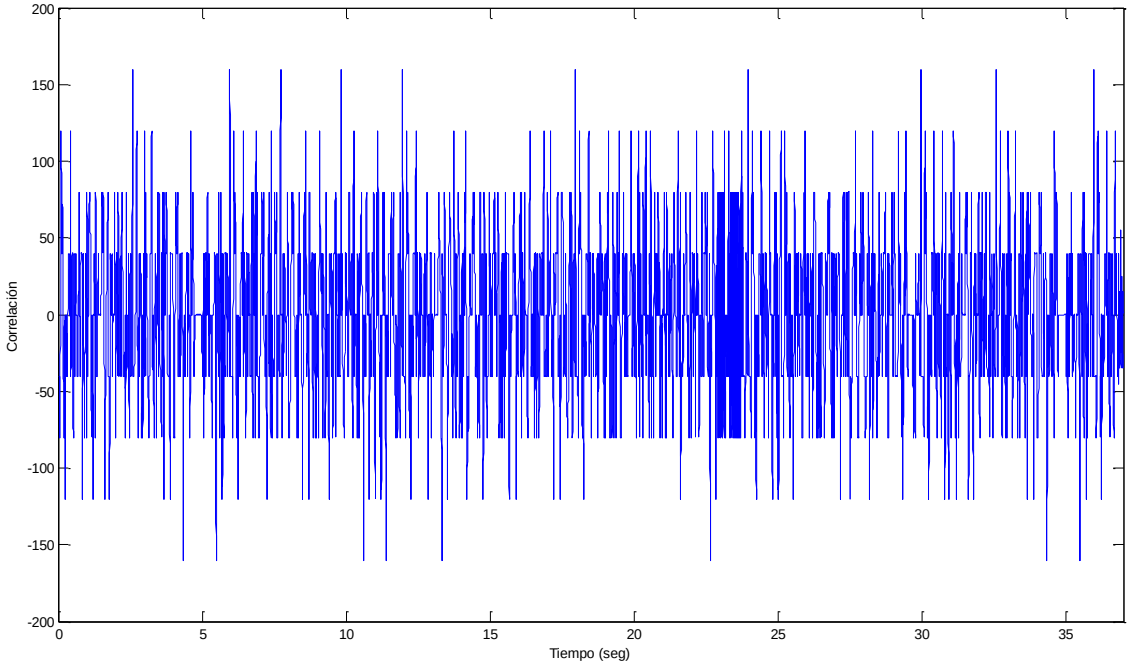

FIGURA 6.7. Correlación entre 37 segundos del dato de navegación y 8 bits del preámbulo.

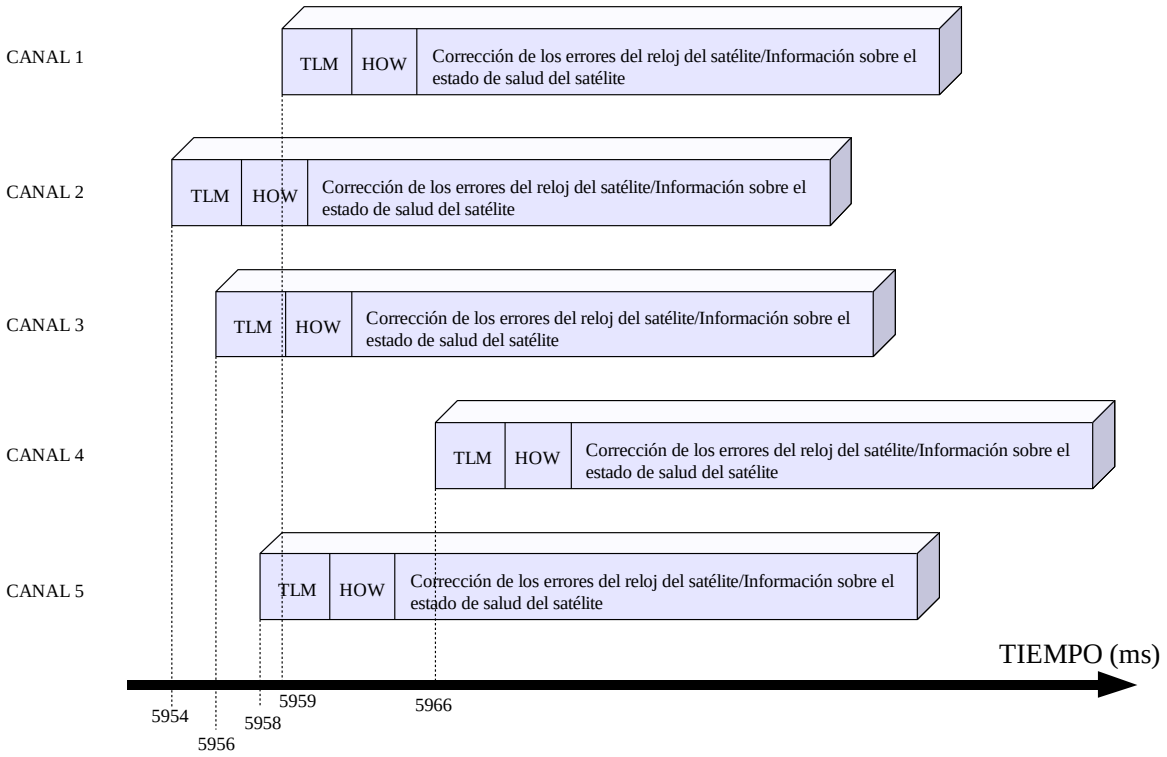

FIGURA 6.8 La figura muestra el inicio de la palabra TLM para cada uno de los satélites adquiridos.

- Checar la paridad de la subtrama
- Encontrar el identificador de la subtrama.
- Decodificar cada subtrama (1 a 3).

La figura 6.8 muestra el inicio del preámbulo en la primera subtrama de cada uno de los satélites adquiridos. En esta figura identificamos el inicio de la palabra TLM en la primera subtrama de cada uno de los canales; y en base a este tiempo se procede a calcular los parámetros de efemérides así como todos los pseudorangos.

Así como en la figura 6.8 se muestra el inicio de la palabra TLM en nuestro receptor GPS, en la tabla 6.2 se muestran los parámetros de efemérides obtenidos en Matlab y en lenguaje C para cada uno de los satélites adquiridos.

Los datos de efemérides obtenidos en la tabla 6.2 se ocupan en el algoritmo de cálculo de posición y estimación de los pseudorangos.

Con esta tabla terminamos la implementación de nuestro algoritmo de lazo de seguimiento; cabe señalar que el lazo de seguimiento únicamente contempla hasta la gráfica de los datos de navegación pero con esto demostramos que nuestro algoritmo no tiene algún

error en su implementación, es por eso que mostramos los parámetros de efemérides de cada uno de los satélites adquiridos.

La implementación en lenguaje C sigue el mismo diagrama a bloques presentado en la figura 6.1 obteniendo los mismos resultados que en Matlab.

TABLA 6.2. Resultados de los parámetros de efemérides obtenidos en Matlab y en lenguaje C.

Parámetro	**Canal 1 (PRN 21)**	**Canal 2 (PRN 29)**	**Canal 3 (PRN 26)**	**Canal 4 (PRN 9)**	**Canal 5 (PRN 15)**
IODE	102	57	20	55	17
C_{rs}	72.5938	-26.4688	8.4063	85.5000	-2.4063
Δn	5.4224e-009	4.2330e-009	3.6319e-009	4.2470e-009	4.3609e-009
μ_0	-1.5622	-1.5301	0.2645	-1.5625	1.8274
C_{uc}	3.5651e-006	-1.2424e-006	7.5996e-007	4.7404e-006	8.3819e-008
e	0.0149	0.0034	0.0203	0.0203	0.0014
C_{us}	5.8562e-006	1.3277e-005	1.2418e-005	6.1356e-006	1.2381e-005
$\sqrt{a}$	5.1536e+003	5.1537e+003	5.1549e+003	5.1536e+003	5.1536e+003
t_{oe}	237600	237600	237584	237600	237584
C_{ic}	1.7881e-007	-3.7253e-008	-1.9744e-007	2.1607e-007	4.2841e-008
Ω_0	0.7512	-0.2906	2.8800	-2.4443	2.8127
C_{is}	-1.7881e-007	8.1956e-008	2.7753e-007	4.7497e-007	-1.8626e-009
i_0	0.9345	0.9604	0.9932	0.9755	0.9570
C_{rc}	248.1875	122.9688	158.8438	269.1250	134.6875
ω	-2.6325	-1.3427	0.9859	1.4656	-0.5760
$\dot{\Omega}$	-8.4804e-009	-7.8678e-009	-7.8107e-009	-8.0939e-009	-7.9182e-009
IDOT	7.0146e-010	7.7146e-011	-2.7465e-010	3.5359e-010	-2.8501e-010
No. semana	1524	1524	1524	1524	1524
Precisión SV	0	0	0	0	0
Salud SV	0	0	0	0	0
T_{GD}	-1.1642e-008	-8.8476e-009	-6.5193e-009	-5.5879e-009	-9.7789e-009
IODC	231	237	242	244	235
t_{oc}	237600	237600	237584	237600	237584
a_{f2}	0	0	0	0	0
a_{f1}	-1.9327e-012	3.9790e-012	2.2737e-012	2.0464e-012	-4.6612e-012
a_{f0}	2.2821e-005	2.1048e-006	5.3004e-005	4.7112e-005	-2.7293e-004

6.2 Implementación del algoritmo de seguimiento sobre el microprocesador PowerPC 405 del FPGA

Antes de pasar a la implementación de nuestro algoritmo sobre el PowerPC 405 es necesario tener descargado e instalado las herramientas de diseño de *Xilinx ISE design suite 10.1* que incluyen el ISE Foundation, EDK y el SDK con su compilador GCC interno; además de lo anterior es necesario haber desarrollado el algoritmo de seguimiento en lenguaje C nativo.

Comenzaremos con una pequeña introducción al software de desarrollo. El Kit de Desarrollo Embebido (EDK – *Embedded Development Kit*) de Xilinx es un conjunto de herramientas y Propiedad Intelectual (IP – *Intellectual Property*) que permite tener en un diseño un sistema completo de procesador embebido para implementación en un dispositivo FPGA. Aquí describiremos el flujo de diseño para desarrollar un sistema de procesamiento embebido personalizado usando EDK [32].

6.2.1 ¿Cómo EDK simplifica el diseño de procesadores embebidos?

Los sistemas embebidos son algo complejos. Obtener las porciones de hardware y software de un diseño embebido para su funcionamiento son proyectos propios del sistema. La fusión de los dos componentes del diseño para que funcione como un sistema trae retos adicionales. Agregar un proyecto de diseño con FPGA, la situación se tornaría en tener un sistema con un potencial de convertirse en realidad muy confuso.

Para simplificar el proceso de diseño, Xilinx ofrece varios conjuntos de herramientas. Esto es buena idea para conocer los nombres de estas herramientas básicas, nombre de los archivos de proyecto, acrónimos y abreviaturas. Los componentes de EDK se mencionan en las siguientes líneas.

6.2.1.1 Entorno de Software Integrado.

El Entorno de Software Integrado (ISE – *Integraded Software Environment*) es la base para el diseño lógico sobre FPGA de Xilinx. Debido a que el diseño con FPGA puede ser un proceso complicado, Xilinx ha proporcionado herramientas de desarrollo de software que te permiten simplificar algunas de estas complejidades. Varias utilidades, como son restricciones de entrada, análisis de tiempo, ruteo lógico, y programación de dispositivos han sido integradas en ISE.

6.2.1.2 Kit de Desarrollo Embebido.

El kit de desarrollo embebido (EDK – *Embedded Development Kit*) es un conjunto de herramientas y Propiedad Intelectual (IP) que permite en tu diseño tener un completo sistema de procesador embebido para implementación en dispositivos FPGA de Xilinx. El software Xilinx ISE también debe ser instalado para ejecutar EDK.

Plataforma de Estudio de Xilinx: La plataforma de estudio de Xilinx (XPS – *Xilinx Platform Studio*) es el ambiente de desarrollo o GUI usada para el diseño de la porción de hardware del sistema de procesador embebido.

Kit de Desarrollo de Software: El kit de desarrollo de software (SDK – *Software Development Kit*) es un ambiente de desarrollo integrado, complementario a XPS, que es usado para la creación y verificación de aplicaciones de software embebido desarrolladas en C/C++. SDK es construido sobre Eclipse; entonces esta herramienta de desarrollo de software puede parecer familiar para varios de nosotros.

Otros componentes de EDK: A continuación se muestra una lista de algunos otros elementos de EDK:

- Hardware IP para el procesador embebido de Xilinx.
- Drivers y librerías para el desarrollo de software embebido.
- Compilador GNU y depurador para desarrollo de software en C/C++ que tiene la finalidad de desarrollar sobre procesadores MicroBlaze y PowerPC.
- Documentación.
- Proyectos muestra.

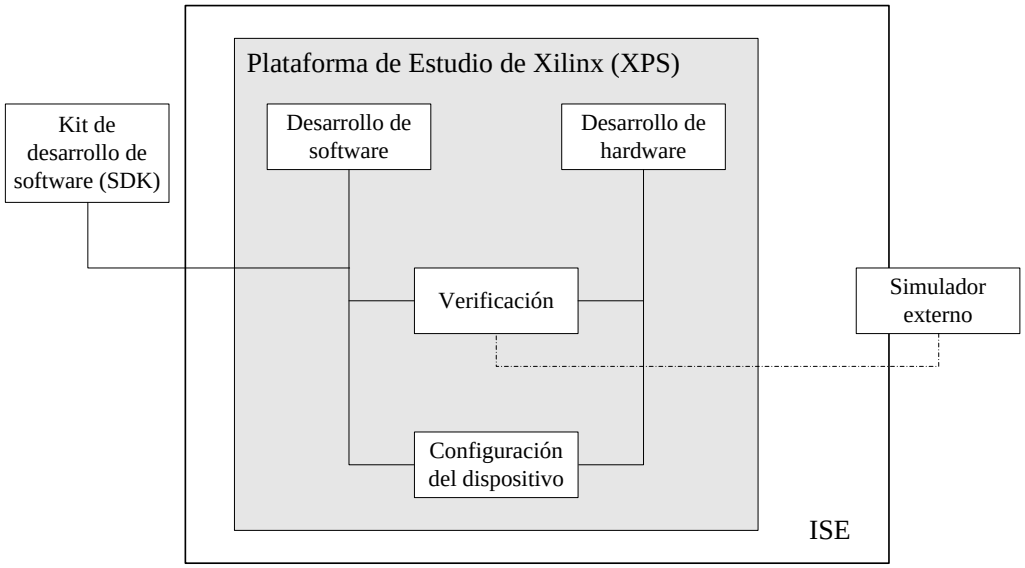

FIGURA 6.9 Diagrama de flujo de un proceso embebido básico.

6.2.2 ¿Cómo hacer para que las herramientas faciliten el proceso de diseño?

La figura 6.9 muestra un diagrama de flujo simplificado para un diseño embebido.

A continuación se da una descripción general de cómo estas herramientas trabajan juntas para simplificar el proceso del diseño.

- El diagrama de flujo de la figura 6.9 recomienda comenzar con un proyecto ISE, y entonces agregar una fuente de procesador embebido al proyecto ISE.
- XPS es usado principalmente el desarrollo de sistemas hardware de procesador embebido. Configuración del microprocesador, periféricos, y la interconexión de estos componentes, junto con la asignación de sus respectivas propiedades, tiene lugar en XPS.
- SDK es el ambiente de desarrollo de software recomendado para aplicaciones de software simples y complejas. Mientras el desarrollo de software básico puede ser realizado dentro de XPS, esta capacidad será removida en las siguientes versiones de la herramienta de desarrollo.
- Verificar el correcto funcionamiento de su plataforma de hardware puede ser realizado ejecutando el diseño a través de un simulador de Lenguaje de Descripción de Hardware (HDL – *Hardware Description Language*). XPS facilita tres tipos de simulación:

a) Situacional
b) Estructural
c) Tiempo real

XPS automáticamente instala el proceso de verificación, incluyendo archivos HDL para la simulación.

6.2.3 Requerimientos de instalación: ¿qué se necesita para ejecutar las herramientas EDK?

6.2.3.1 ISE Xilinx.

Varias utilidades en EDK usan la funcionalidad que se entrega en las herramientas contenidas en ISE. Entonces, para usar las herramientas de EDK, primero es necesario tener instalado el ISE. Estar seguro que también se tiene instalado el último *service pack* de ISE. Para obtener información y descargar el producto para el software ISE, ver [43].

6.2.3.2 Registrar el producto.

Es necesario tener un ID de registro del software para poder instalar EDK. Se puede hacer en línea a través de [43].

6.2.3.3 Instalación de EDK.

Las instrucciones exactas de instalación varían, dependiendo de si el software se obtuvo a partir de una descarga electrónica o un DVD. Es importante que el ISE y EDK sean la misma versión. Por ejemplo, si tu estas instalando la versión 10.1 de EDK, también debes instalar la versión 10.1 de ISE [43].

6.2.3.4 Requerimiento de instalación para simulación.

Para realizar simulación usando las herramientas de EDK, se tienen que tener los siguientes pasos completados:

1. Un simulador instalado (ModelSim PE/SE v6.3c o Mentor Graphics IUS v6.1) es requerido para los pasos de simulación.
2. Compilar las librerías de simulación.

6.2.4 Creando el proyecto.

El constructor de sistema base (BSB – *Base System Builder*) es un asistente que rápida y eficientemente establece un grupo de trabajo de diseño, que también se puede modificar. Nuestro proyecto lo creamos usando el BSB. Xilinx recomienda el uso del asistente BSB para crear la base de cualquier nuevo proyecto de diseño embebido. BSB puede ser todo lo que necesitas para crear tu diseño, pero si más adecuaciones son requeridas, BSB le ahorra mucho tiempo debido a que automatiza configuraciones básicas de la plataforma hardware y software de las tareas comunes de la mayoría de los diseños con procesador. Después de ejecutar el asistente, uno tiene un proyecto que contiene todos los elementos básicos necesarios para construir uno o más sistemas personalizados o complejos, si fuera necesario.

Usando el asistente BSB, uno puede crear el archivo del proyecto, escoger una tarjeta, seleccionar y configurar un microprocesador y el uso de interfaces I/O, agregar periféricos internos, crear software, y generar un reporte con el resumen del sistema. BSB reconoce los componentes y configuraciones del sistema de la tarjeta seleccionada y establece las opciones adecuadas para nuestra selección [32].

6.2.4.1 Creando el archivo de nivel superior del proyecto (.xmp).*

Un archivo llamado Proyecto de Microprocesador de Xilinx (XMP – *Xilinx Microprocessor Project*) es el archivo de nivel superior que da la descripción del sistema embebido durante el desarrollo. Toda la información del proyecto del XPS es guardada en el archivo XMP, incluyendo la ubicación del los archivos de Especificación de Hardware del Microprocesador (MHS – *Microprocessor Hardware Specification*) y la Especificación de Software del Microprocesador (MSS - *Microprocessor Software Specification*).

El archivo XMP también contiene información sobre código C y archivos de cabecera que XPS compila, así como algún archivo ejecutable que el SDK compile. El

proyecto también incluye las familias de arquitecturas FPGA y el tipo de dispositivo para que la herramienta de hardware sea ejecutada.

Para ejecutar el asistente BSB, es necesario primero ejecutar el navegador de proyecto de ISE, y crear un proyecto con un sistema de procesador embebido.

1. Ejecutar `ISE Project Navigator`

2. Seleccionar `File > New Project`. Esto inicia el asistente de nuevo proyecto.
3. Usa la información listada en la tabla 6.3 para configurar los parámetros de diseño sobre la pantalla del asistente.

TABLA 6.3. Pasos a ejecutarse en el ISE Project Navigator.

Pantalla del asistente	Propiedad del sistema	Configuración / comando de uso
Crear nuevo proyecto	• `Project name` • `Project location` • `Top-level source type`	• Escoge un nombre para el proyecto (no usar espacios). • Escoge la ubicación del proyecto. • Selecciona HDL (default) • Dar click en **`Next`.**
Propiedades del dispositivo	• `Product category` • `Family` • `Device` • `Package` • `Speed` • `Synthesis Tool` • `Simulator` • `Preferred Language`	• All • Virtex2P • XC2VP30 • FF1152 • -5 • XST (VHDL/Verilog) • ModelSim-PE Mixed • VHDL • Aceptar otras características y dar click en **`Next`**
Crear nueva fuente		• Dar click en **`New Source`**
Seleccionar tipo de fuente	En el árbol de menú en el panel izquierdo, dar click en `Embedded Processor.`	• Introduce el nombre (en nuestro caso colocamos "tesis") en el campo `File name` • Dar click en **`Next`** • Dar click en **`Finish`** • Dar click en **`Yes`** • Dar click en **`Next`**
Agregar fuentes existentes	No agregar alguna	• Dar click en **`Next`**
Resumen del proyecto		• Dar click en **`Finish`**

Después de pasar por el navegador de proyecto de ISE y el asistente de nuevo proyecto, este reconoce que tenemos un sistema de procesador embebido, y ejecutara la plataforma de estudio con el mensaje. *Este proyecto parece ser un proyecto en blanco. ¿Le gustaría crear un sistema base usando el asistente BSB?* Damos click en **`Sí.`**

Ahora que el asistente BSB ha empezado, podemos crear un proyecto usando las características descritas en la tabla 6.4

TABLA 6.4. Pasos a ejecutarse en el asistente BSB.

Pantalla del asistente	Propiedad del sistema	Configuración / comando de uso
Bienvenido al constructor de sistema base (BSB)	Opciones del tipo de proyecto	• Selecciona de opción de crear un nuevo diseño (`I would to create a new design`)
Seleccionar tarjeta	Seleccionar una tarjeta de desarrollo	• Seleccionamos que deseamos crear un sistema para una tarjeta personalizada (`I would like to create a system for a custom board`)
Selector de procesador	Tipo de procesador	• Seleccionar `PowerPC`
Configurar el procesador PowerPC	• Frecuencias de reloj	• `Reference clock frequency` = 100 MHz • `Processor clock frequency` = 100 MHz • `Bus clock frequency` = 100 MHz
Agregar fuentes existentes	• `Processor configuration (Debug I/F, OCM)`	• Escoger `FPGA JTAG` • `On-chip memory (OCM)` - `Data: 32 KB` - `Instruction: 64 KB` Cache debe ser deshabilitada
Configurar Interfaces de I/O	Dar click en agregar dispositivo (`Add Device`)	• Agregar UART RS232 y dar click en OK. Paridad ninguna.
Agregar periféricos internos	`XPS_BRAM_IF_CNTLR` con un tamaño de memoria de 8 KB	• Seleccionar 64 KB de tamaño de memoria
Configuración de software	• Dispositivos para entrada, salida y memoria	• STDIN: RS232 • STDOUT: RS232 • Boot memory: `xps_bram_if_cntlr_1`
	• Selección de aplicación muestra	• Ninguna

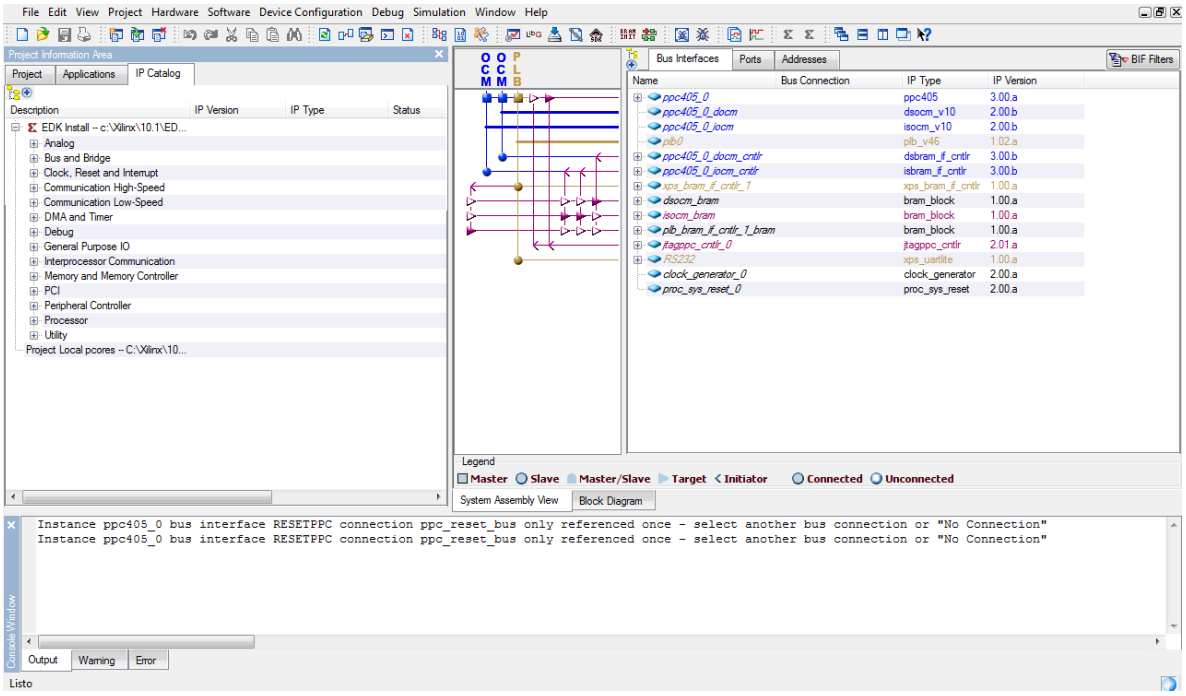

FIGURA 6.10 Interface de usuario del XPS

Para crear una nueva aplicación de software, tenemos que ir a la pestaña `Applications` del panel izquierdo del XPS mostrado en la figura 6.10. Una vez estando allí, damos doble click sobre `Add Software Application Project` como se muestra en la figura 6.11

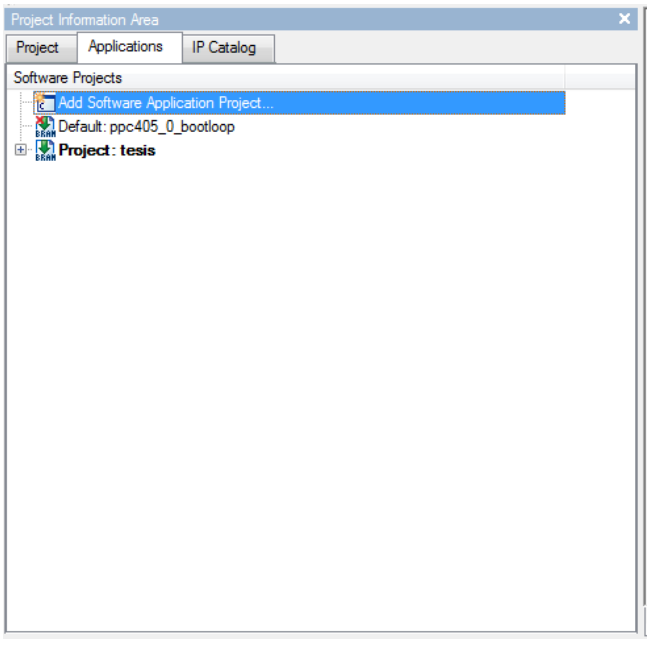

FIGURA 6.11 La figura muestra como agregar una nueva aplicación de software

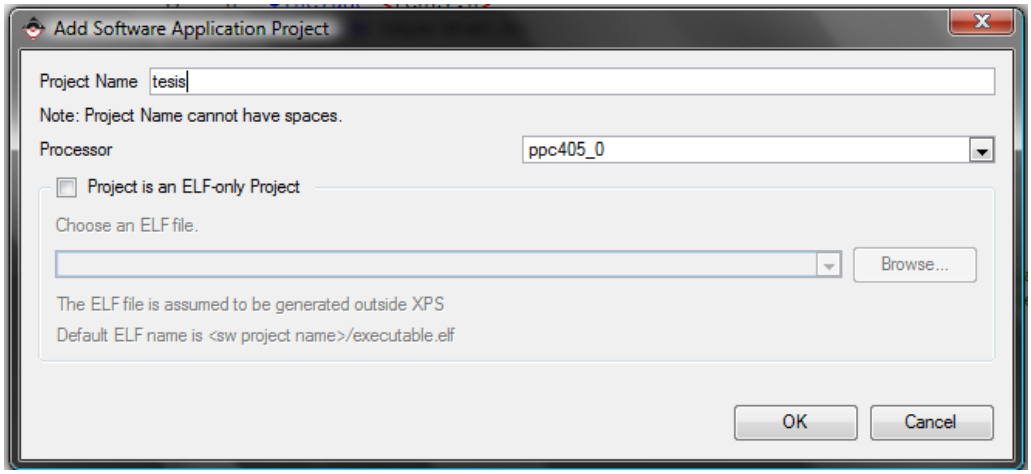

FIGURA 6.12 Configuración de los parámetros requeridos al agregar una nueva aplicación de software.

Una vez realizados los pasos anteriores, ya se tiene listo un árbol para el proyecto de software embebido. Ahora damos click en la rama llamada `Sources` que abrirá un cuadro de búsqueda de archivo, aquí cargamos el proyecto en lenguaje C, en nuestro caso el proyecto se llama "`seguimiento.c`". Una vez añadido el archivo "`seguimiento.c`" se pueden cambiar varias opciones del compilador, haciendo doble click en `Compiler Options`, todo este proceso se muestra en la figura 6.13

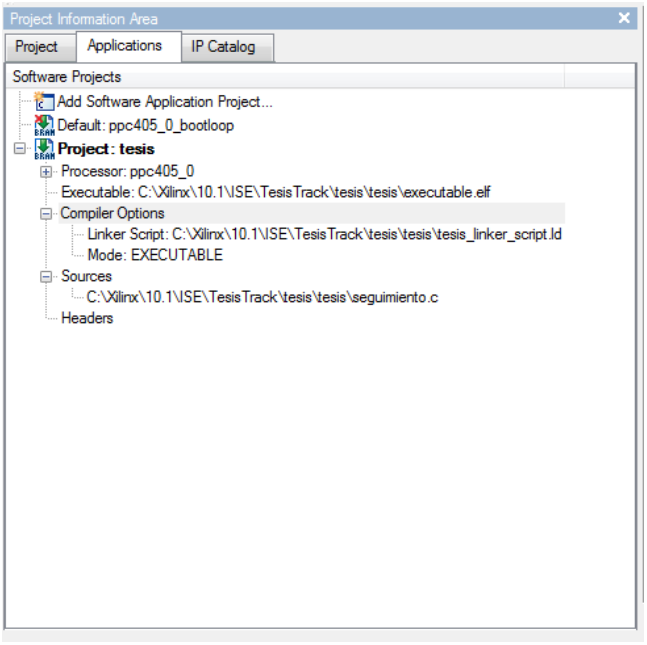

FIGURA 6.13 Aquí se muestra como es cargada una aplicación nueva de software embebido.

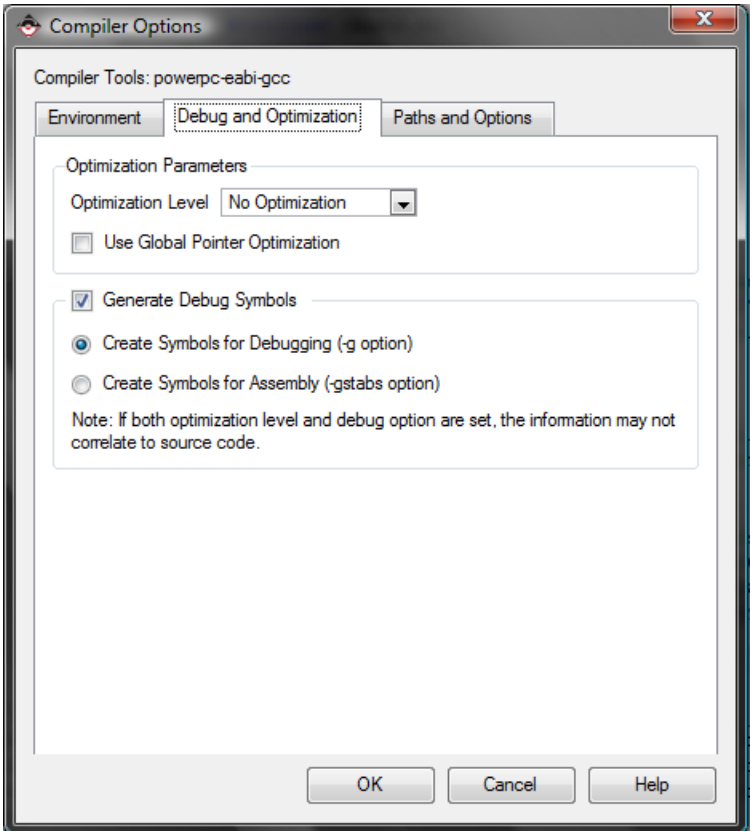

FIGURA 6.14 Aquí se muestra los parámetros que se deben cambiar en las opciones del compilador

Desde el menú de opciones del compilador se pueden cambiar parámetros como dirección de arranque del programa, optimizaciones, etc. Para nuestra aplicación, modificamos la opción de nivel de optimización, desde la pestaña `Debug and Optimization` y el parámetro `Optimization level` lo cambiamos por `No Optimization`, como se muestra en la figura 6.14.

Ahora solo queda compilar la aplicación e incluirla en la memoria del sistema para que se ejecute cuando se descargue al XtremeDSP. Pero para esto es necesario seguir los pasos indicados en la siguiente sección.

6.2.5 Implementación del diseño.

Una vez teniendo completo el diseño, ahora podemos implementarlo en hardware. Para esto nos situamos en el XPS y seleccionamos `Hardware > Generate Netlist` dentro del menú. Este comando genera la utilidad Generador de Plataforma XPS (Platgen) para leer la información de la plataforma de diseño contenida en el archivo Especificación de Hardware del Microprocesador (MHS). Los archivos de salida de Platgen son archivos HDL, que pueden ser encontrados en `<project name>\hdl\`.

Cuando seleccionamos `Hardware > Generate Netlist`, la Tecnología de Sintetización de Xilinx (XST – *Xilinx Synthesizer Technology*) sintetiza estos archivos de diseño HDL para generar los archivos `IP netlist (NGC)`, como se muestra en la figura 6.15.

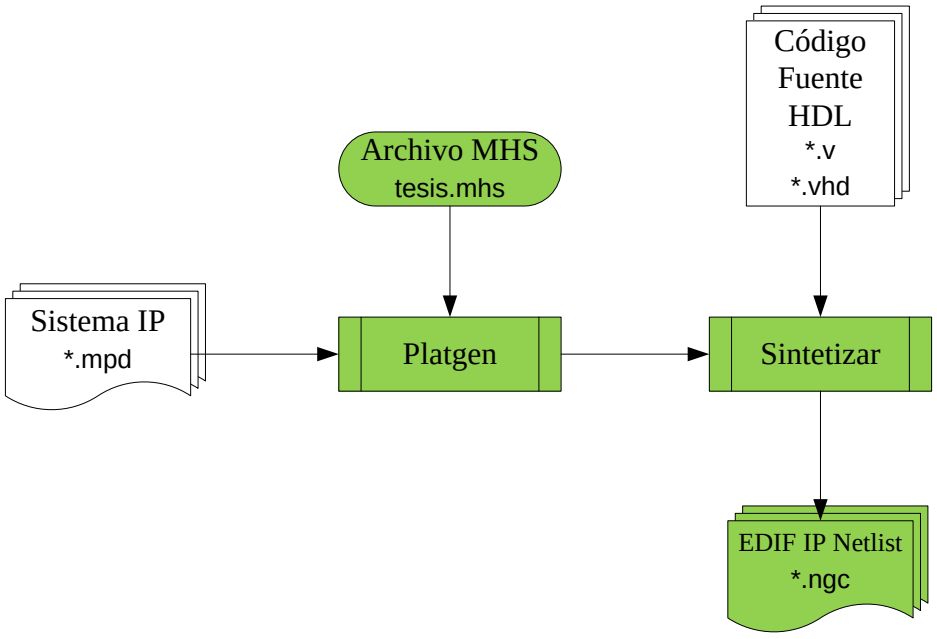

FIGURA 6.15 Elementos y etapas de la generación del Netlist

Para referencia general los archivos resultantes NGC residen en:`<nombre del proyecto>\implementation`. Notar que estos archivos no necesitan ser cambiados. ISE usa los archivos Netlist NGC durante la implementación del diseño, que ocurre cuando invocamos el **`Generate Programming File`** del navegador de proyecto de ISE. Cuando damos doble click en `Generate Programming File`, se hacen los pases mostrados en la figura 6.16.

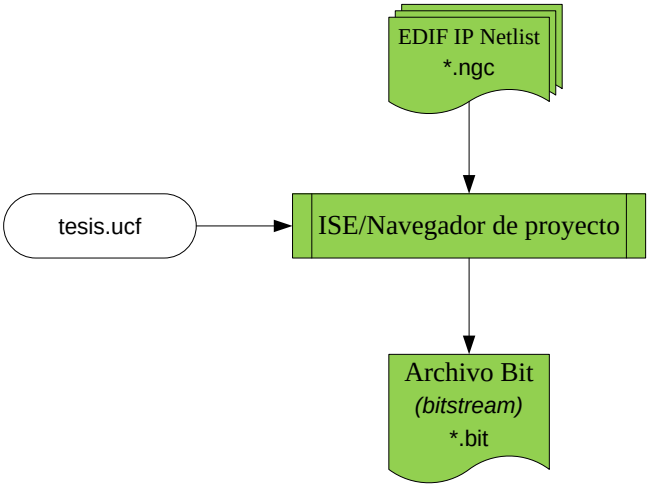

FIGURA 6.16 Elementos y etapas de la generación del Bitstream

Los archivos NGC y restricciones de "tesis" son procesados a través de las herramientas restantes de ISE (NGDBuild, MAP, PAR y TRACE) y BITGEN del navegador de proyecto de ISE. El Navegador de Proyecto solo necesita un archivo, aparte del de diseño de procesador de alto nivel, es el archivo de restricciones de usuario (UCF – *User Constraints File*). Este archivo contiene las restricciones (limitaciones) del diseño, ver figura 6.17. Si no estamos familiarizados con el diseño sobre FPGA, el uso de las restricciones del diseño permite a las herramientas identificar y satisfacer las limitaciones en tiempo real. Un ejemplo de restricciones podría ser tan simple como información de reloj o colocación de algún pin, o podría ser alguna colocación compleja y parámetros de tiempo para satisfacer rutas lógicas críticas.

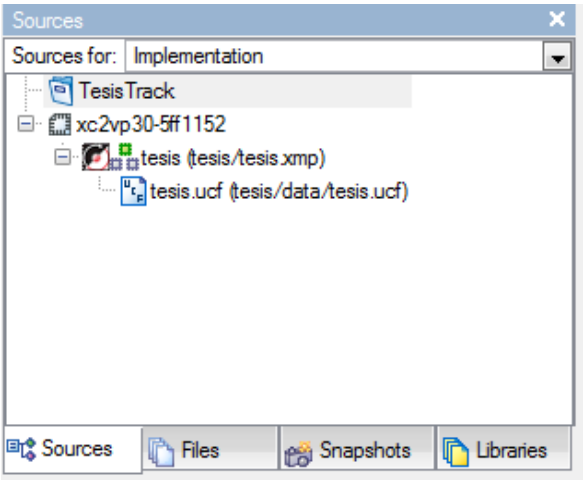

FIGURA 6.17 Muestra de ubicación del archivo de restricciones de usuario (UCF)

El proyecto de esta tesis es un diseño centrado en el procesador; esto es, consiste solo de la plataforma de procesador embebido. No hay lógica externa asociada con el sistema del procesador. Por lo tanto, solo se necesitara agregar pocas restricciones antes de generar el *bitstream*. Nosotros ya tenemos un archivo UCF debido a que cuando ejecutamos el asistente BSB, seleccionamos un pequeño conjunto de restricciones que estaban basadas sobre la tarjeta que seleccionamos. Cuando completamos los pasos del asistente BSB, las restricciones fueron automáticamente generadas. El UCF está ubicado en el directorio `<project name>\tesis\data`.

A continuación se muestran los pasos para generar el `Netlist` y `Bitstream`:

1. Seleccionar `Project > Add Source` en el navegador de proyecto ISE, e ir al subdirectorio tesis/data.
2. Seleccionar el archivo `tesis.ucf`. Dar click en OK, dar OK nuevamente en ventana de confirmación "`Adding Source Files …` "
   Si todo es correcto veremos algo similar a lo mostrado en la figura 6.16.
3. Dar doble click en `Generate Programming File` y observar el progreso de la implementación del diseño en la consola de Windows. Esto lo podemos observar en la figura 6.18.

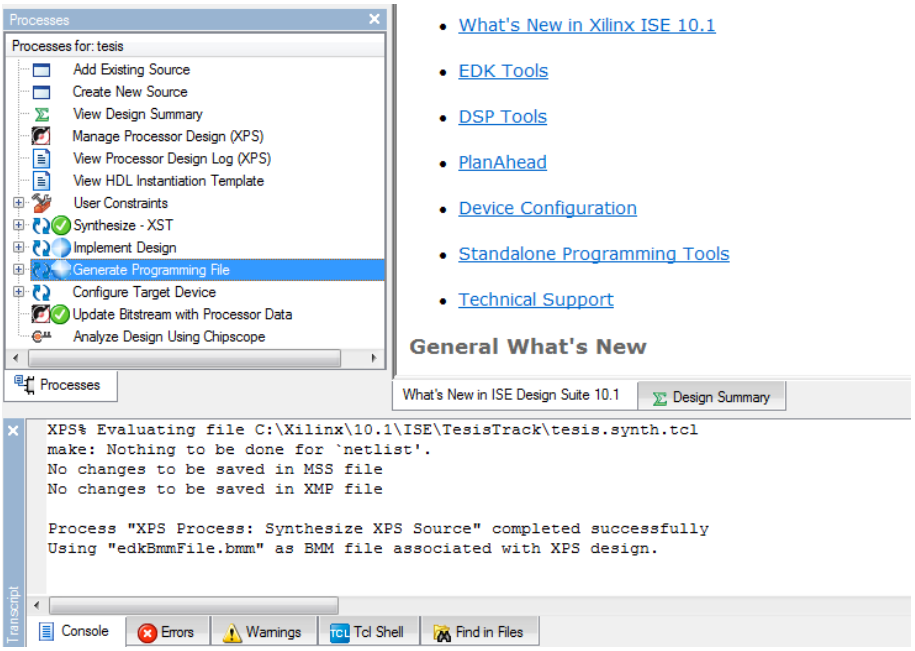

FIGURA 6.18 Proceso para obtener el Netlist y Bitstream.

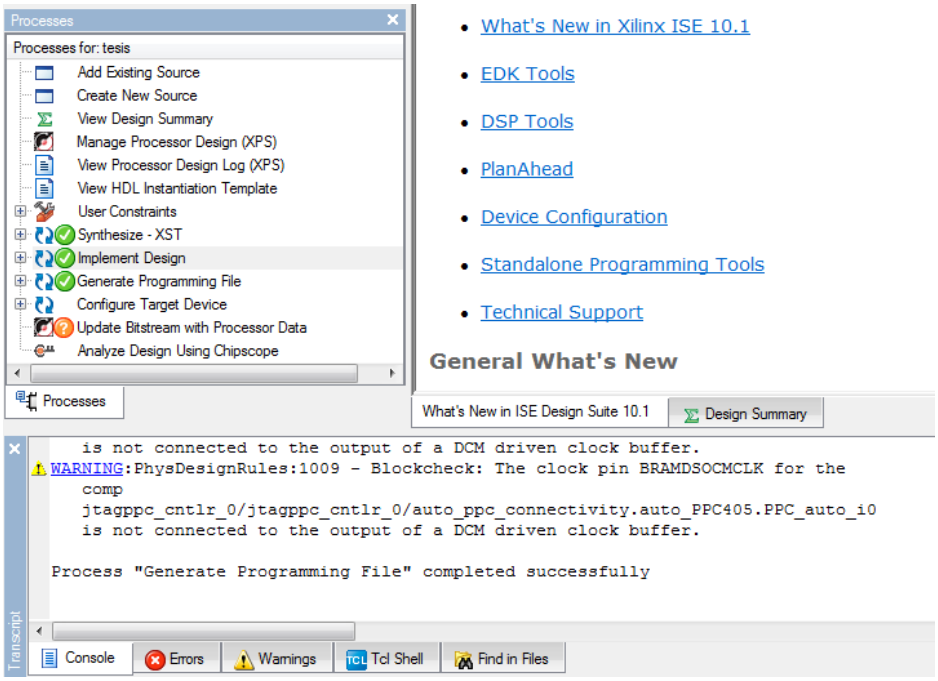

FIGURA 6.19 Proceso exitoso de la obtención del Netlist y Bitstream.

Si todo es correcto veremos algo similar a lo mostrado en la figura 6.19, y con esto terminamos la primera etapa de nuestro diseño que es generar el primer `Bitstream`, porque cuando se compile nuestra aplicación en lenguaje C generaremos una actualización de este `Bitstream` obtenido en los pasos mencionados anteriormente.

Ahora configuraremos el software del sistema dentro de la plataforma de estudio de Xilinx XPS y ejecutar nuestra aplicación. Para configurar la parte de software, dentro del XPS vamos a `Software > Software Platform Settings` en donde se abrirá una ventana con tres opciones como se muestra en la figura 6.20. La primera opción `Software Platform` permite modificar algunos parámetros de diseño, así como las librerías que se deseen incluir (xilnet, xilmfs, xilfile, etc.) y se desea utilizar algún sistema operativo. Aquí modificaremos el parámetro `CORE_CLOCK_FREQ_HZ` y ponemos 100000000 que es la frecuencia de reloj del sistema que se está implementando, en la opción de `OS & Library Settings` en la parte de `OS` seleccionamos `xilkernel`. Estas modificaciones hechas las podemos observar en la figura 6.20.

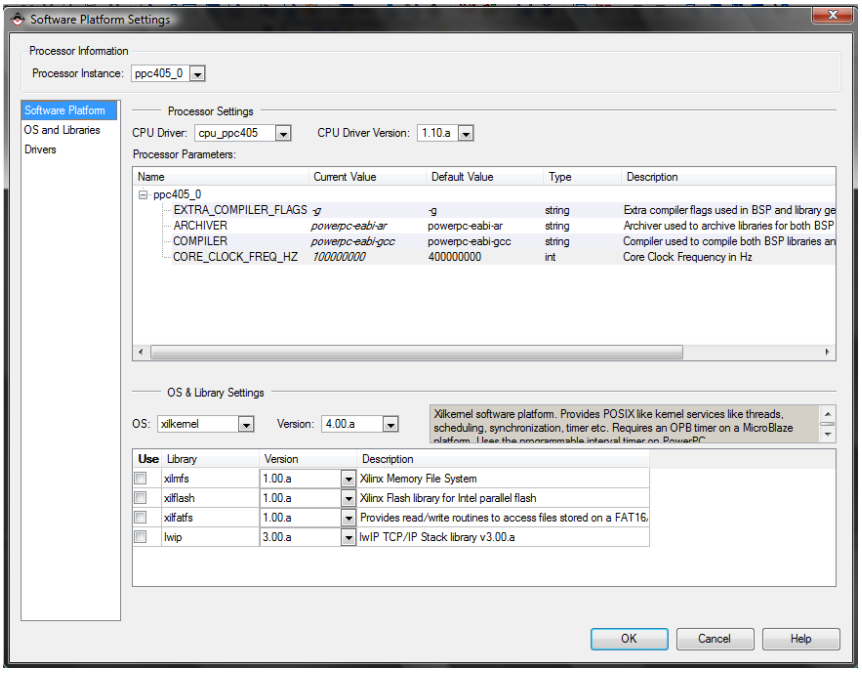

FIGURA 6.20 Modificación de los parámetros de diseño

En la siguiente sección, `OS and Libraries` se pueden modificar parámetros del sistema operativos que se vaya a utilizar. Aquí, dentro de xilkernel hacemos la modificación del parámetro `stdout` a RS232 y `stdin` a RS232, quedando como se muestra en la figura 6.21.

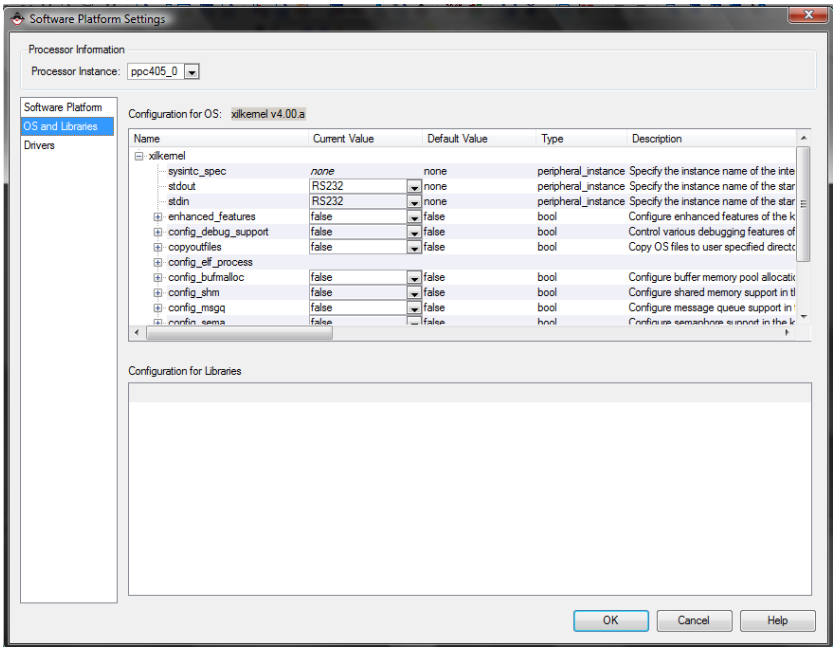

FIGURA 6.21 Modificación de los parámetros del sistema operativo

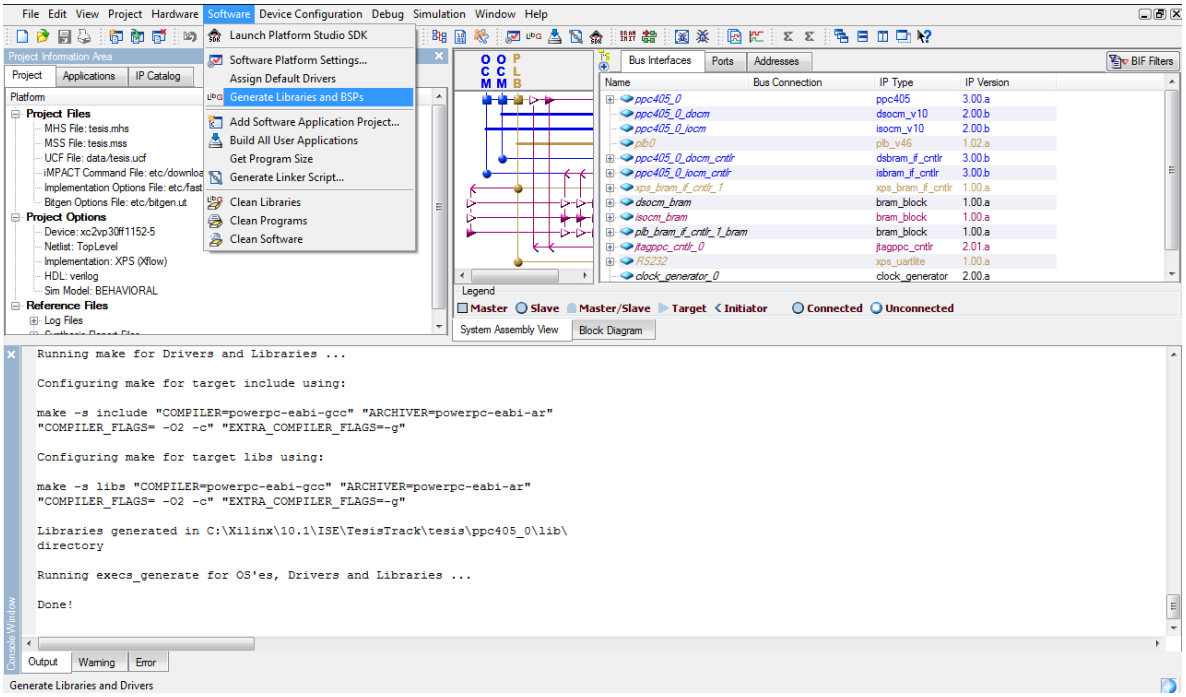

FIGURA 6.22 Generar las librerías de diseño.

Si las librerías no han sido compiladas es necesario ir a `Edit > Preferences` y dentro de preferencias damos click en `Compile`.

Para generar las librerías (previamente compiladas) en el XPS damos click en `Software > Generate Libraries and BSPs` y después de unos segundos si todos es correcto veremos lo mostrado en la figura 6.22 la cual nos dice que las librerías fueron compiladas correctamente y están listas para usarse en nuestra aplicación.

Una vez que ya tenemos las librerías listas lo único que nos queda es compilar nuestra aplicación e incluirla en la memoria del sistema para que se ejecute cuando se descargue a la placa. Para compilar la aplicación simplemente damos click derecho sobre la rama principal del proyecto de software que ya hemos cargado anteriormente cuando creamos el proyecto y damos click en `Mark to Initialize BRAMs` después volvemos a dar click derecho dentro la aplicación y seleccionamos `Build Project`.

Estos pasos se muestran en la figura 6.23.

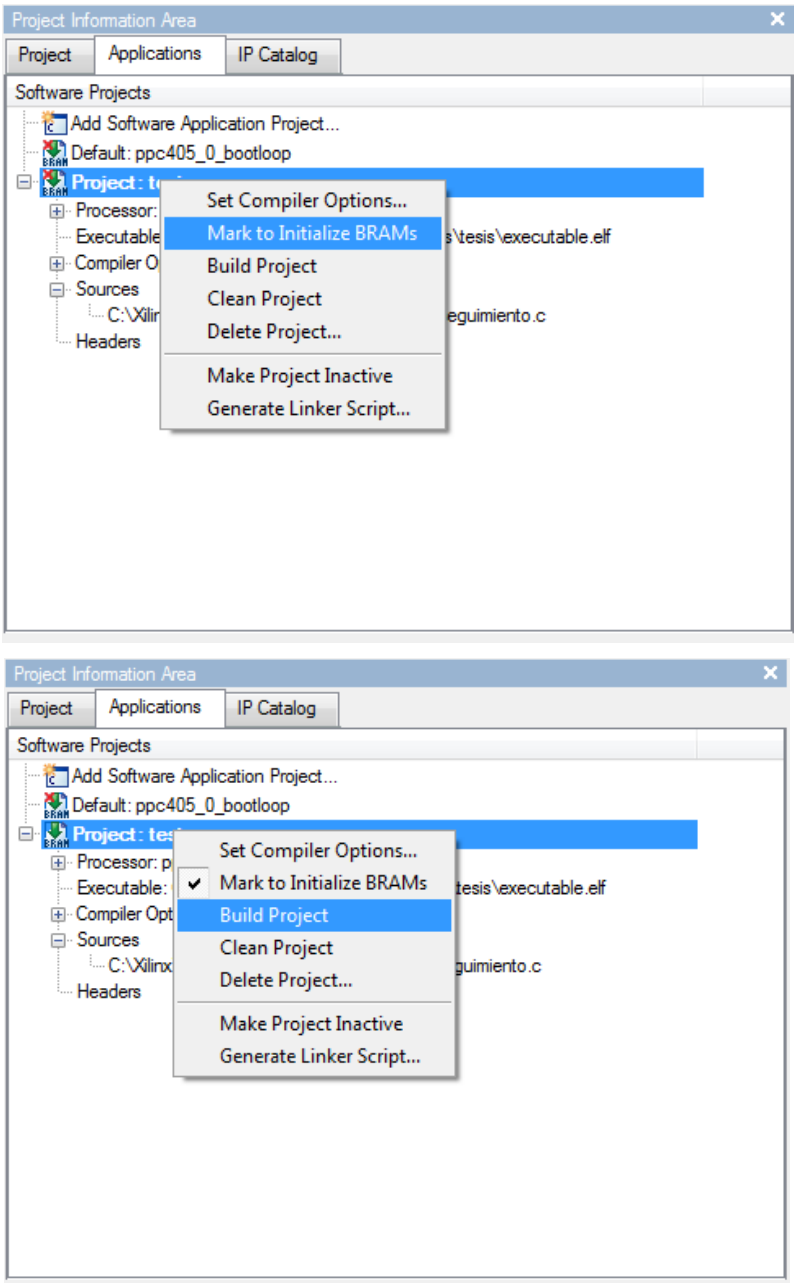

FIGURA 6.23 Pasos a seguir para compilar el proyecto y cargarlo en la memoria

Una vez seguidos los pasos de la figura 6.23 y si todo es correcto tendremos ya casi listo nuestro diseño embebido y en la consola de Windows veremos lo mostrado en la figura 6.24 en donde nos indica que nuestro archivo ejecutable con extensión `*.elf` ha sido generado exitosamente y lo único que faltaría es actualizar el `Bitstream` generado anteriormente y así obtener el `Bitstream` final que se descargara a nuestro XtremeDSP.

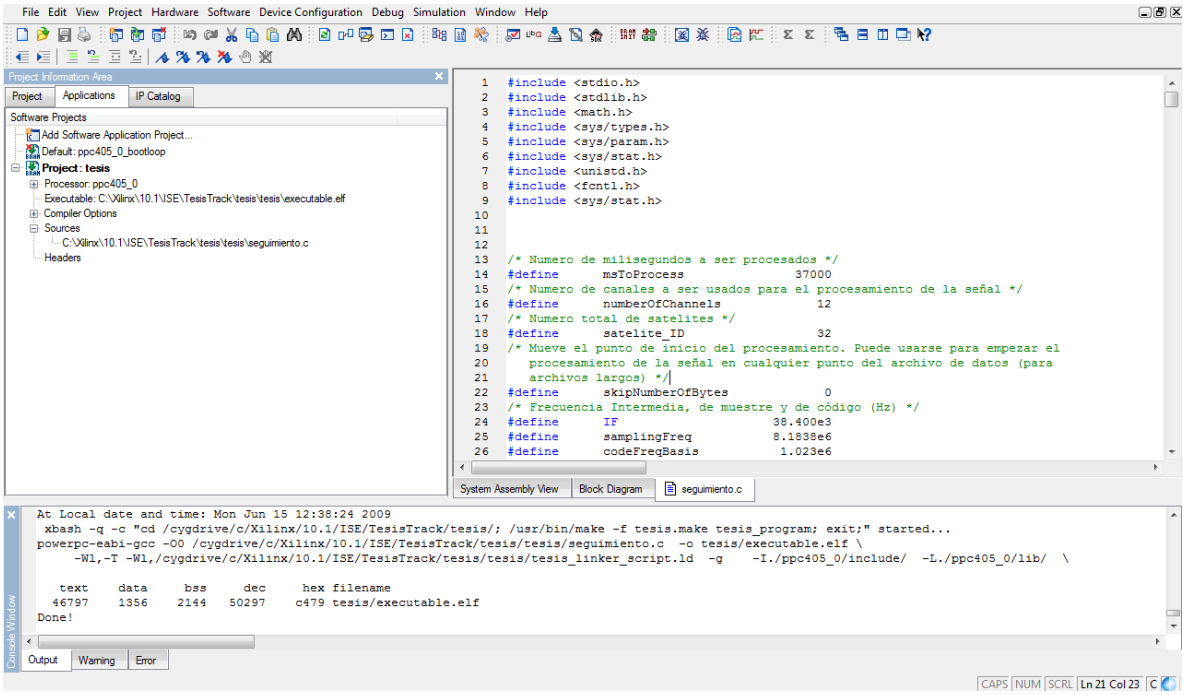

FIGURA 6.24 Obtención del archivo tesis/executable.elf

Una vez que ya tenemos el archivo ejecutable con extensión `*.elf` lo único que queda por hacer es actualizar el bitstream para esto nos vamos a la ventana del navegador de proyecto ISE y damos doble click en `Update Bitstream with Processor Data`, como se muestra en la figura 6.25

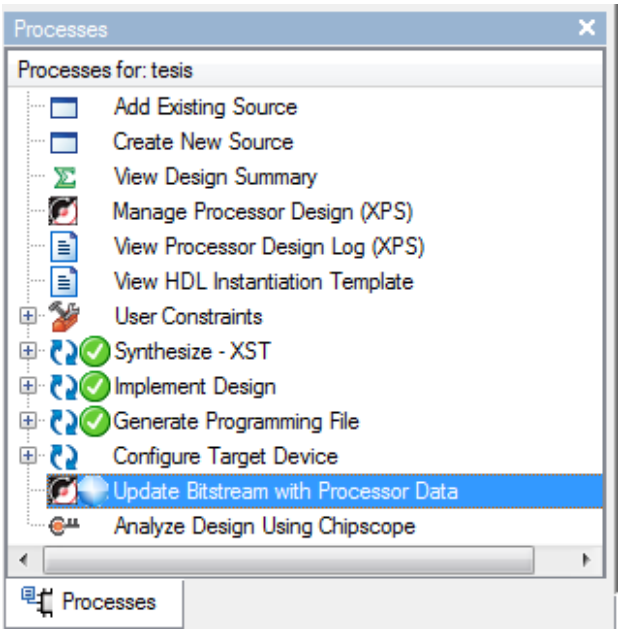

FIGURA 6.25 Actualización del `bitstream`.

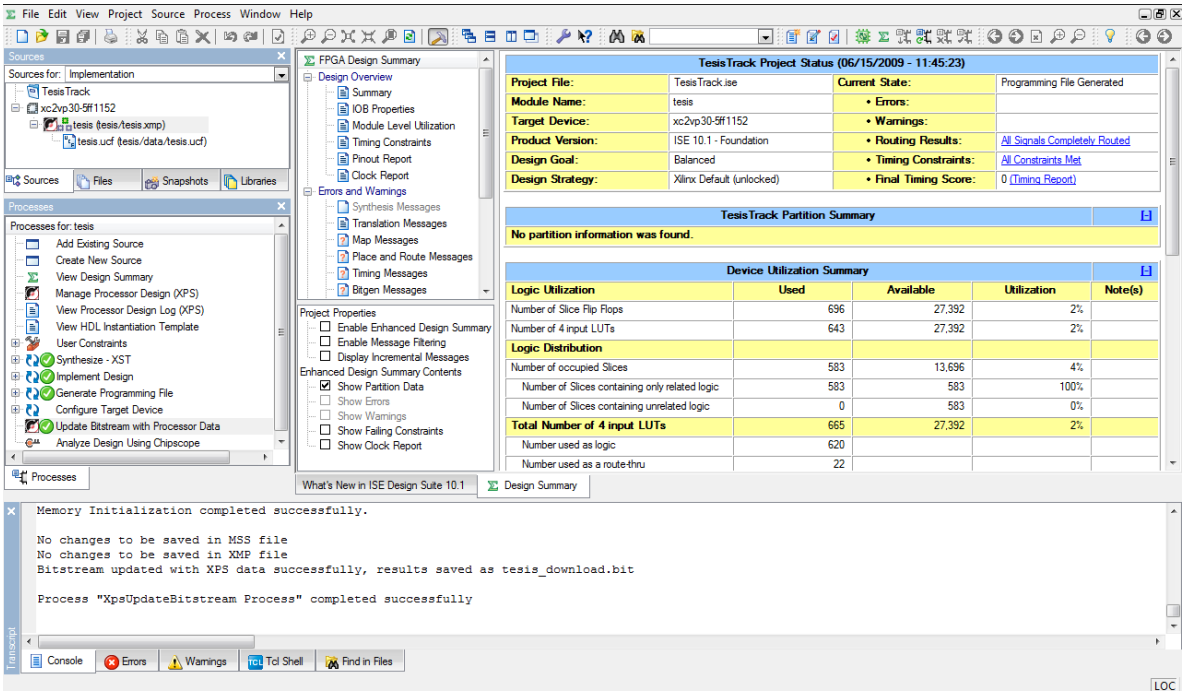

FIGURA 6.26 La actualización del `bitstream` se ha realizado satisfactoriamente y tenemos listo nuestro diseño para su implementación en el kit de desarrollo de Nallatech.

Si todo se ha ejecutado correctamente veremos lo mostrado en la figura 6.26 lo cual indica que nuestro diseño embebido ha sido realizado exitosamente y ya lo tenemos listo para implementarlo en nuestro kit de desarrollo.

La figura 6.27 muestra un diagrama de flujo muy completo de todos los pasos a seguir en cada una de las herramientas de software para realizar un diseño embebido. Este diagrama es la base para cualquier implementación.

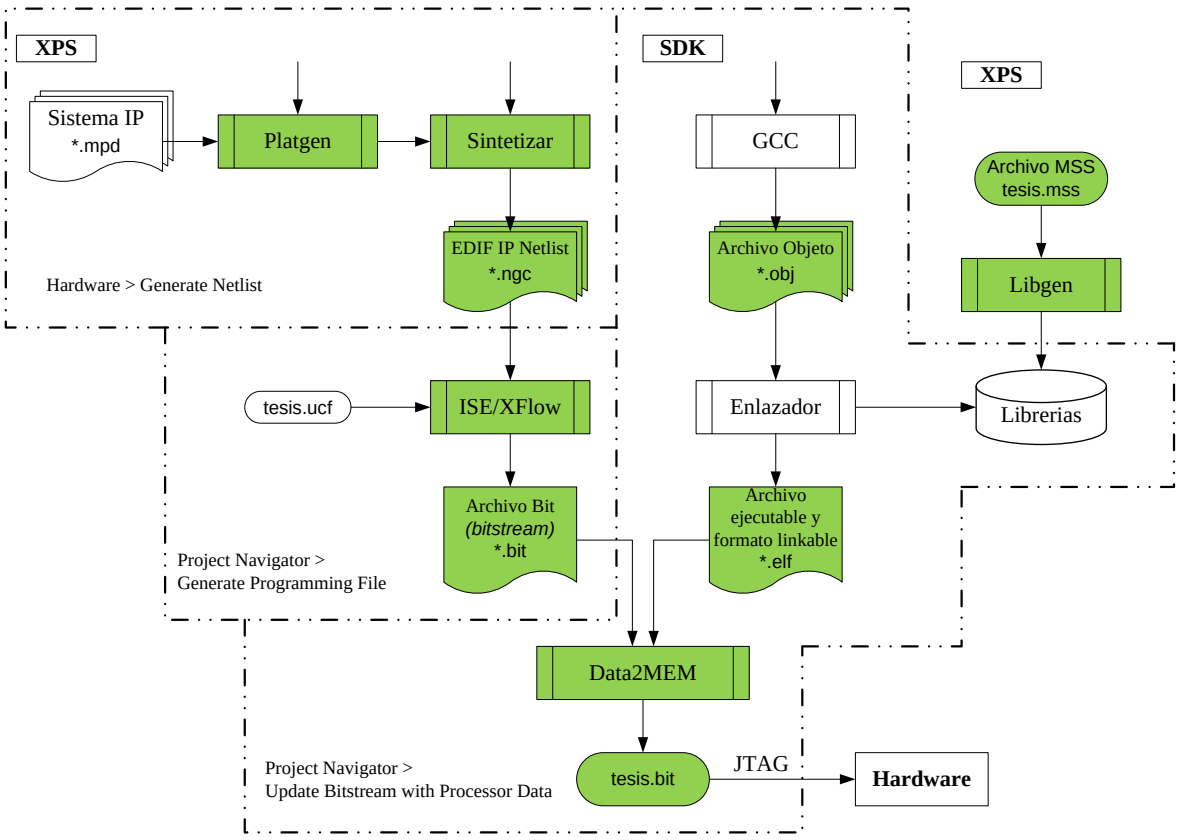

FIGURA 6.27 Elementos y etapas principales del XPS y EDK para la configuración del diseño sobre FPGA.

Hasta esta etapa lo único que hemos hecho es desarrollar todo el proyecto y obtener el archivo final de nuestra tesis que lleva por nombre `tesis.bit` con ayuda de las herramientas de Xilinx. En la siguiente sección se presenta la última etapa de nuestra tesis; que es, la implementación o descarga del archivo `tesis.bit` (obtenido en las secciones anteriores) sobre el FPGA.

6.2.6 Implementación del algoritmo de seguimiento en el kit de desarrollo XtremeDSP Pro. Descarga del archivo `tesis.bit` en el FPGA.

Para poder descargar e implementar el archivo final que contiene el algoritmo de seguimiento de la señal GPS y algunos parámetros importantes del diseño embebido hacemos uso de la herramienta de desarrollo que viene en los discos que acompañan a nuestro kit de desarrollo; la herramienta que vamos a ocupar se llama FUSE (Edición para Windows). El motivo del porqué usar esta herramienta está en que nuestro kit de desarrollo no tiene manera de descargar nuestra aplicación en el FPGA porque no cuenta con JTAG y

no se puede hacer uso de herramientas propias de Xilinx como es el iMPACT; y en cambio la herramienta FUSE tiene una manera muy sencilla de descargar los archivos con extensión `*.bit` sobre el dispositivo. Veamos cómo se hace esto.

6.2.6.1 Descripción del software FUSE.

El software FUSE es una interface de usuario de alto nivel para interconectarse con los módulos y tarjetas madre DIME y DIME-II de Nallatech. El software FUSE es una aplicación basada en Java que permite al usuario una interfaz muy fácil con múltiples tarjetas, configurar FPGAs, y transferencias apply2 DMA. La aplicación también permite al usuario el control de tarjetas a través del lenguaje propio de Nallatech – DIMEScript.

6.2.6.2 Pasos previos antes de la descarga del archivo `tesis.bit`.

Como primer punto, es importante señalar que tenemos que tener ya instalado el software FUSE que viene en el disco con nombre *FUSE (Windows Edition)* e instalar los *drivers* de la tarjeta con ayuda del disco *XtremeDSP Development Kit Pro*; una vez ya teniendo listos estos primeros pasos instalamos la aplicación que viene en este último disco la cual nos va ayudar a cargar por completo todos los componentes de la tarjeta y ocupar por completo la tarjeta. Es importante señalar que si no se llega a instalar esta última aplicación cuando queramos abrir la tarjeta desde el software FUSE, la aplicación nos arrojará un error diciéndonos que no es posible abrir la tarjeta.

Como una nota, cabe señalar que los *drivers* de la tarjeta únicamente están disponibles para Windows XP Profesional, Windows 2000, Windows Millennium Edition, Windows NT Service Pack 4 y Linux para kernel v2.4.

6.2.6.3 Como abrir una tarjeta.

Esta sección describe el proceso de uso del sistema FUSE para encontrar tarjetas sobre una interface especifica. Seguir los pasos siguientes para abrir una tarjeta:

1. Dar click en `Inicio > Programas > FUSE > Software > FUSE probe`. Cuando la interfaz de usuario es cargada no hay tarjetas abiertas, como se muestra en la figura 6.28.

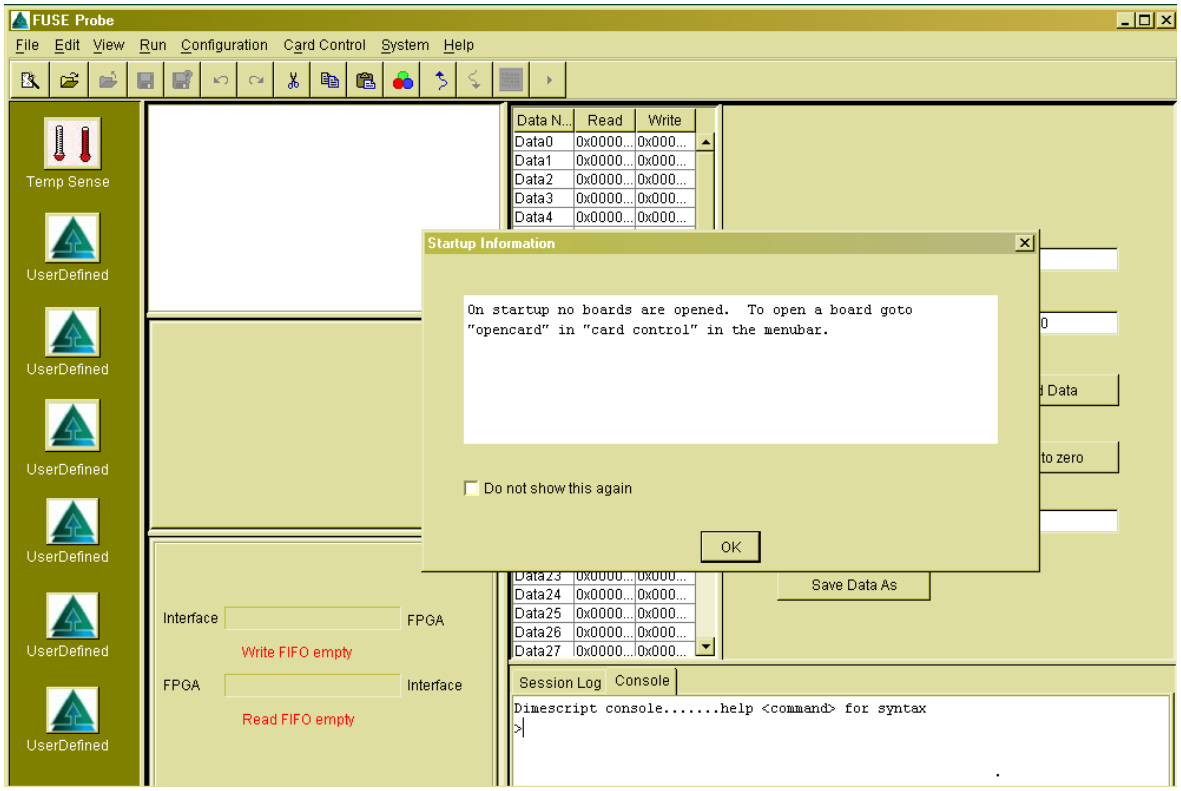

FIGURA 6.28 No hay tarjetas cargadas cuando se abre la interfaz de usuario del software FUSE

2. Entonces damos click en OK. En la ventana del programa FUSE Probe mostrada en el figura 6.29, seleccionamos `Card Control > Open Card` en el menú de la parte de arriba de la ventana.
3. La ventana de `Locate Card` es mostrada en la figura 6.30. seleccionamos el tipo de interface en el recuadro `Interface`, el tipo de tarjeta en el recuadro `Card Type`. El paso 5 explica el propósito de estos recuadros en más detalle.
4. Damos click en el boton `Advanced` para expandir la caja de dialogo y ver la lista completa de las opciones de la tarjeta.
5. En la caja de dialogo hay dos principales menús. El primero es el tipo de interface, que proporciona las opciones para PCI, USB, Canal Citrix Virtual o TCPIP. El segundo menú es el tipo de tarjeta que permite seleccionar un tipo específico de tarjeta o una búsqueda de todas las tarjetas. Aquí describimos como abrir la tarjeta BenONE sobre el bus USB. Seleccionamos la interfaz USB y `All Card` en el tipo de tarjeta. Después damos click en el botón `Locate Cards`.

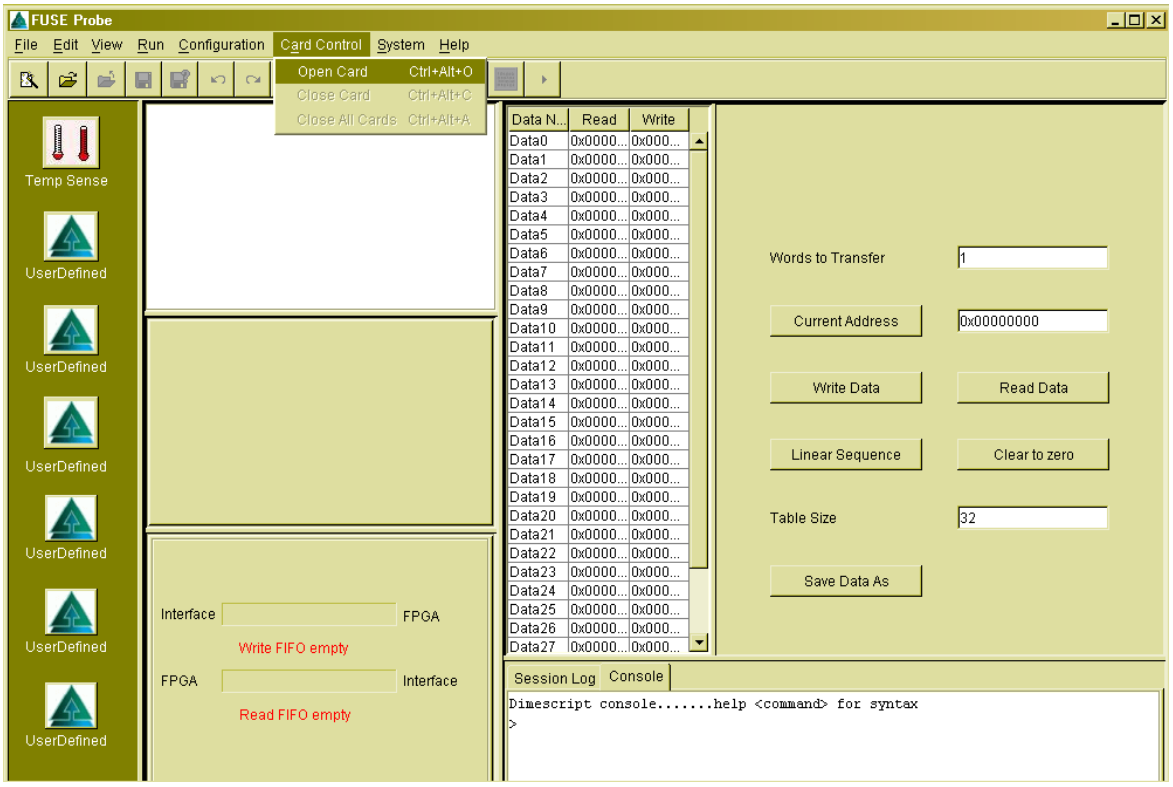

FIGURA 6.29 Como abrir una tarjeta dentro del software FUSE

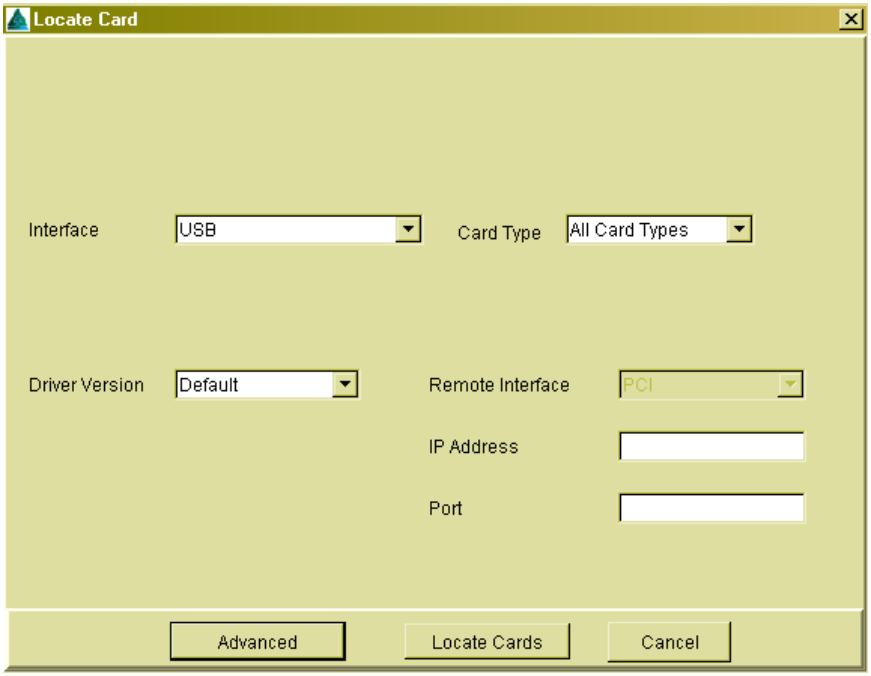

FIGURA 6.30 Ventana mostrada para localizar tarjetas.

6. Cuando se ha localizado la tarjeta, otra ventana es desplegada mostrando las tarjetas encontradas. Esto se puede ver en la figura 6.31.

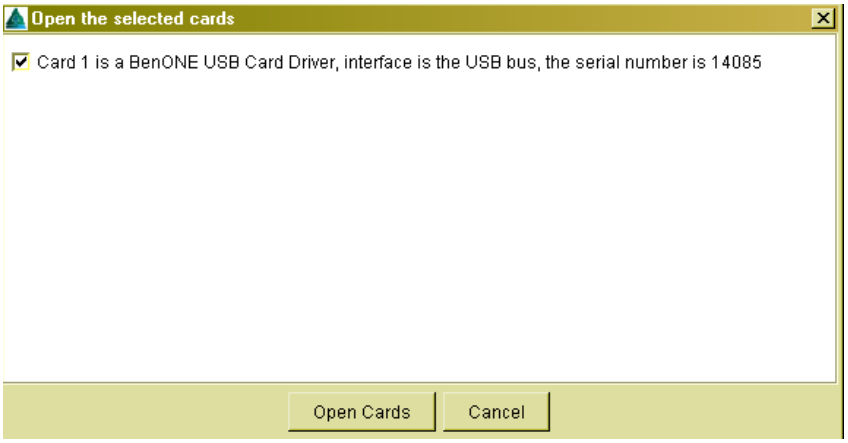

FIGURA 6.31 Ventana mostrada para abrir tarjetas.

7. La figura 6.31 muestra que una tarjeta BenONE fue encontrada. Para abrir la tarjeta, solo basta con seleccionar la tarjeta y dar click en el botón `Open Cards`. La tarjeta que es abierta se mostrada en la figura 6.32.

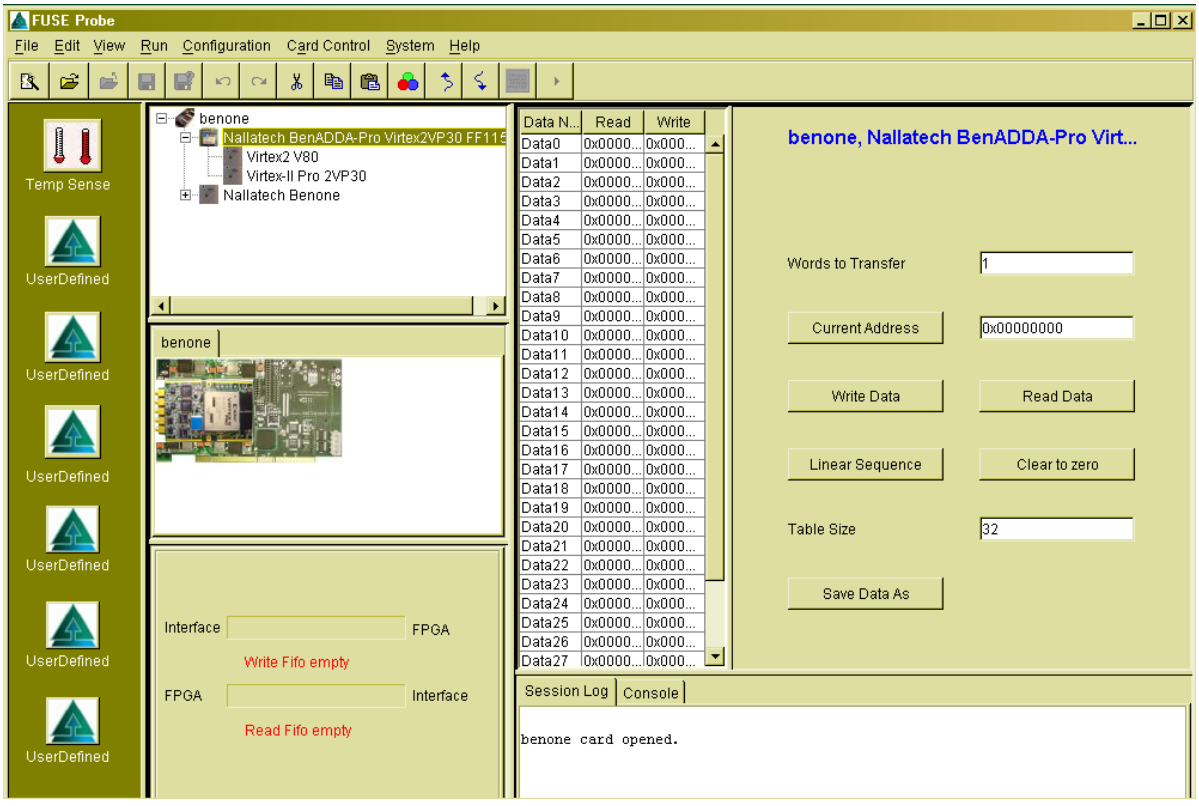

FIGURA 6.32 Tarjeta BenONE abierta.

8. Para cerrar la tarjeta, vamos al menú `Card Control` y seleccionamos `Close Card`.

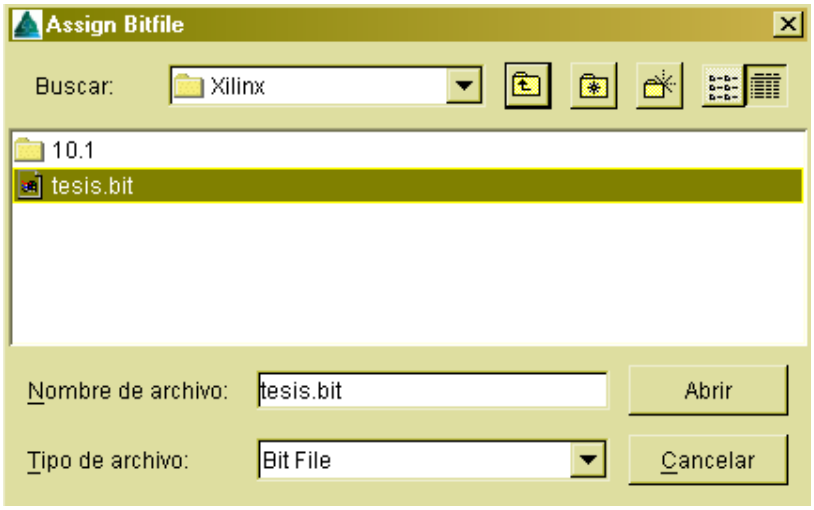

FIGURA 6.33 Selección del archivo `tesis.bit`.

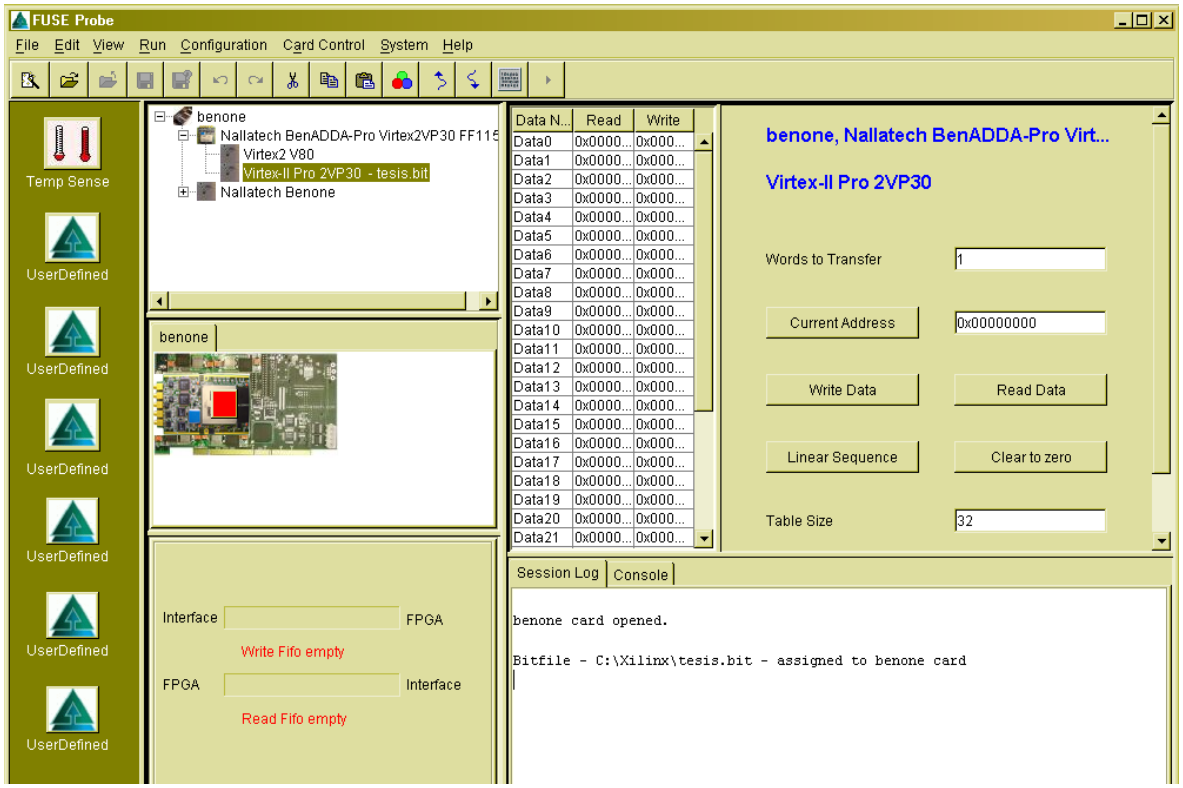

FIGURA 6.34 Asignación del archivo `tesis.bit` en el FPGA.

6.2.6.4 Descarga del archivo `tesis.bit` sobre él FPGA.

Esta sección describe cómo usar el software FUSE para descargar el archivo con extensión `*.bit`. Después de que la tarjeta fue abierta como se mostro en la sección anterior, se tiene que mostrar lo visto en la figura 6.32. Para asignar el archivo `*.bit`,

damos click sobre el dispositivo al que deseamos asignar el *bitfile* – en nuestro caso el Virtex-II Pro 2VP30. Damos click en `Configuration` sobre la barra de menú y entonces seleccionamos `Assign Bitfile`, como se muestra en la figura 6.33.

Entonces seleccionamos `tesis.bit` y damos click en `Abrir`. El archivo `*.bit` es ahora asignado sobre la tarjeta Virtex-II Pro 2VP30 que fue seleccionada anteriormente. Cuando el *bitfile* es asignado la `session log` es actualizada y muestra que el *bitfile* fue asignado, como se muestra en la figura 6.34.

Es importante que el correcto *bitfile* sea descargado para el dispositivo específico. El FPGA sobre el Nallatech BenADDA-Pro Virtex2VP30 FF1152; por lo tanto el *bitfile* ha sido compilado específicamente por un Virtex-II Pro 2VP30.

Para configurar el dispositivo seleccionamos el FPGA en el árbol desplegado, entonces en el menú nos vamos a `Configuration > Configure Device`. El archivo `tesis.bit` es ahora cargado sobre el dispositivo FPGA y se verá lo mostrado en la figura 6.35.

Después que exitosamente se hizo la configuración del FPGA, se debe llevar a cabo una restauración del FPGA. Dentro del diagrama de árbol hay cuatro pestañas. Dar click en la pestaña `Resets`, ahora damos click sobre el botón `Interface Reset`. Completamos la restauración dando click en la opción `FPGA Reset` y nuevamente le damos click para que quede en su estado normal (on – off).

Después de estos pasos el diseño `tesis.bit` ahora estará ejecutándose sobre la tarjeta BenONE.

Y con estos últimos pasos terminamos con la implementación del algoritmo de seguimiento de la señal GPS sobre dispositivos lógicos programables (FPGA).

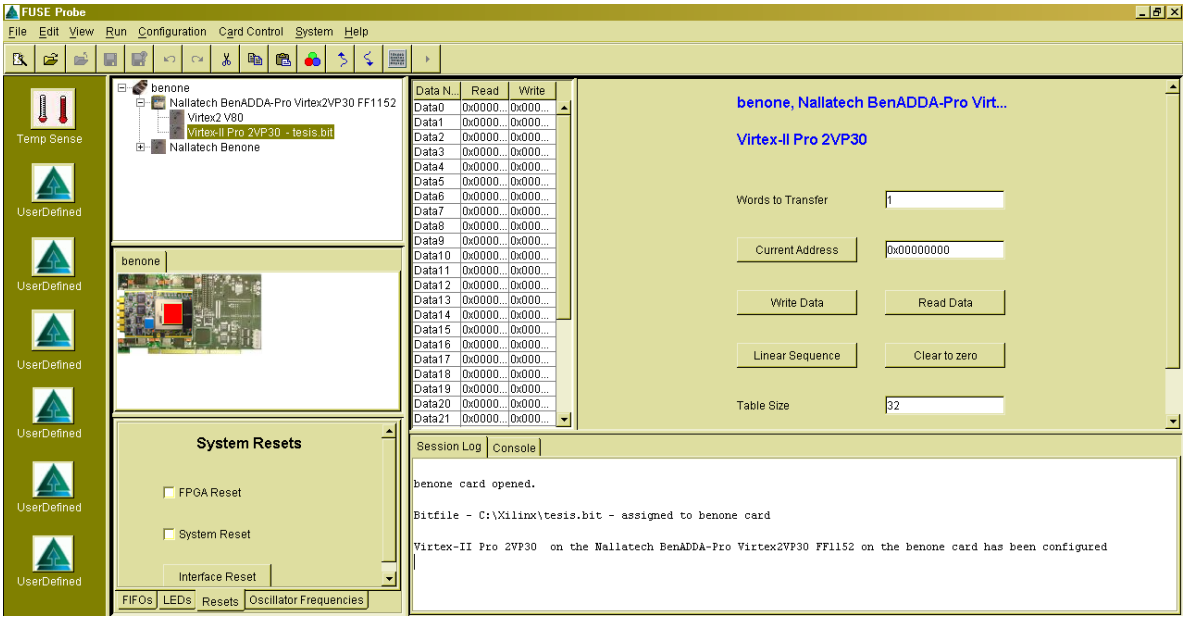

FIGURA 6.35 FPGA configurado.

Conclusiones

El seguimiento de la señal GPS debe proporcionar una excelente sincronización para reducir la desalineación residual después de la adquisición.

El desarrollo de un receptor GPS basado en software contiene enormes perspectivas con diferentes áreas y aplicaciones. En primer lugar, establece el desarrollo de un conocimiento profundo de la estructura de la señal GPS y algoritmo de procesamiento de la señal. En segundo lugar, por definición, el receptor basado en software ya está preparado está preparado para futuros cambios en la estructura de la señal GPS.

La plataforma de desarrollo del FPGA soportó la implementación del sistema, otorgándonos además una gran flexibilidad en su manera de programación; por lo que el proyecto adquiere características cognitivas.

Los objetivos propuestos para este trabajo de tesis fueron cubiertos considerando en todo momento las características de la solución propuesta.

Trabajo a futuro

Después de encontrar los datos de navegación, el preámbulo de cada una de las subtramas y los parámetros de efemérides para cada uno de los satélites adquiridos; la siguiente etapa y para continuar con el proyecto sería evaluar el algoritmo de cálculo de

pseudorango y cálculo de posición para tener un receptor GPS completo dentro de una plataforma de Radio Definido por Software.

Implementar la generación de códigos C/A en las compuertas del FPGA para liberar carga al microprocesador PowerPC 405.

Desarrollo de una interface de comunicación entre la terminal de entrada (dispositivo GN3Sv2) y el kit de desarrollo XtremeDSP Pro para capturar las tramas GPS directamente en el FPGA y no depender de que forzosamente se requiera tener una computadora para la captura de tramas.

Referencias

Bibliográficas:

[1] Akos, D., *A Software Radio Approach to Global Navigation Satellite System Receiver Design*, Ohio University, Athens, OH. 1997.

[2] Balanis, Constantine A., *Antenna Theory: Analysis and Design*, John Wiley & Sons, Inc., Segunda edición, New York, NY., 1996.

[3] Best, Roland E., *Phase-Locked Loops: Design, Simulation and Applications*. Mc Graw Hill, Quinta edición, New York, NY., 2003.

[4] Borre, K., Akos, D., Bertelsen, N., Rinder, P., Holdt, J. S., *A Software-Defined GPS and Galileo Receiver. A Single-Frequency Approach*. Spring Street, New York, NY., 2000.

[5] Ceballos, F., *Enciclopedia del Lenguaje C*, Alfaomega Ra-ma, México, D.F., 1997

[6] Chung, B.-Y., Chien, C., Samueli, H. & Jain, R., *Performance analysis of an all-digital BPSK direct-sequence spread-spectrum IF receiver architecture*. IEEE Journal on Selected Areas in Communications, 11(7): 1096 – 1107.

[7] Dixon, R.C., *Spread Spectrum Systems*. John Wiley & Sons, Segunda edición, New York, NY, 1984.

[8] Gardner, F., *Phaselock Techniques*. John Wiley & Sons, Tercera edición, Palo Alto, California, CAL., 2005.

[9] Gold, Robert. *Optimal binary sequences for spread spectrum multiplexing*, IEEE Transactions on Information Theory, 13(4): 614-621, 1967.

[10] Golomb, S. *Shift Register Sequences*. Aegean Park Press, Laguna Hills, CA., 1982.

[11] Gromov, K., Akos, D., Pullen, S., Enge, P. & Parkinson, B. *GIDL: Generalized Interface Detection and Localization System*, En el 13th International Technical Meeting of the Satellite Division of the Institute of Navigation, Salt Lake City, UT, 2000, páginas 447-457.

[12] Haykin, S., *Communications Systems*, John Wiley & Sons, Cuarta edición, New York, NY., 2000, página 95.

[13] ICD-GPS-200, *Interface control document. ICD-GPS-200*. Arinc Research Corporation, 11770 Warner Ave., Suite 210, Fountain Vallet, CA, 1991.

[14] IS-GPS-200D, *Navstar GPS Space Segment/Navigation User Interfaces*. Arinc Engineering Services, 2004

[15] Joyanes, L., Zahonero, I., *Programación en C: Metodología, algoritmos y estructura de datos*. Mc Graw Hill/Interamericana de España, S. A. U., Segunda edición, 2005

[16] Kaplan, Elliot D. & Hegarty, Christopher J., *Understanding GPS, Principles and Applications*. Artech House, Segunda edición, Boston, MA. 2006. Capítulo 4 y 5.

[17] Kerninghan, B., Ritchie, D., *El Lenguaje de Programación C*. Pearson Educación, Segunda edición, AT&T Bell Laboratories, Murray Hill, New Jersey, 1991.

[18] Márquez, F., *UNIX: Programación avanzada*. Alfaomega Ra-ma, Tercera edición, México, D.F., 2004

[19] Mitra, S., *Procesamiento de Señales Digitales: Un enfoque basado en Computadora*. Mc Graw Hill, Tercera edición, University of California, Santa Barbara, 2007.

[20] Moore, H., *Matlab para ingenieros*. Pearson, Prentice Hall, Primera edición, Salt Lake City, Utah, 2007

[21] Oppenheim, A. & Schäfer, R. *Discrete-Time Signal Processing*. Prentice Hall, Englewood Cliffs, NJ., 1999.

[22] Parkinson, Bradford W. & Spilker Jr., James J., *Global Positioning System: Theory and Applications*, Volumen I de *Progress in Astronautics and Aeronautics*. Instituto Americano de Aeronáuticos y Astronautas, Inc., Washington, DC. 1996, página 335, 371.

[23] Proakis, J., *Digital Communications*, Mc Graw Hill, Cuarta edición, New York, 2000.

[24] Shanmugan, K. Sam & Breipohl, A.M. *Random Signals: Detection, Estimation and Data Analysis*. John Wiley & Sons, New York, NY., 1988

[25] SPS. *Global Positioning System Standard Positioning Service Signal Specification*, U.S. Department of Defense, 1995.

[26] Tsui, J., *Fundamentals of Global Positioning System Receivers: A Software Approach*, John Wiley & Sons, New York, NY. 2000.

[27] Ziemer, Rodger E. & Peterson, Roger L., *Digital Communications and Spread Spectrum Systems*, MacMillan, New York, NY., 1985.

Manuales:

[28] Anonimo. *Application Note: Selecting an A/D converter*. Texa Instruments. 2000

[29] *Clock Module Reference Core*, Xilinx, Reference Design, 2003.

[30] *Data2MEM Users Guide*, Xilinx, UG658, 2009.

[31] Ditzler, G., *Reference Manual for Xilinx Platform Studio, the PowerPC, MicroBlaze & the XUP V2P*.

[32] *EDK Concepts, Tools and Techniques. A Hands-On Guide to Effective Embedded System Design*, Xilinx, XTP013, 2008.

[33] *Embedded System Tools Reference Manual, Embedded Development Kit*, Xilinx.

[34] *OS and Libraries Document Collection*, Xilinx, 2008.

[35] *PowerPC Processor Reference Guide*, Xilinx, UG011, 2007

[36] *PowerPC 405 Processor Block Reference Guide. Embedded Development Kit*, Xilinx, UG018, 2008.

[37] *The PowerPC 405™ Core*, IBM Microelectronics Division, Research Triangle Park, NC 27709, 1998.

[38] *Virtex-II Pro Platform FPGA User Guide*, Xilinx, UG012, 2003.

[39] *Virtex-II Pro and Virtex-II Pro X Platform FPGAs: Complete Data Sheet*, Xilinx, DS083, 2007.

[40] *XtremeDSP Development Kit Pro User Guide*, Nallatech, NT107-0246, 2004

Internet

[41] Artículos sobre GPS
http://www.insidegnss.com/aboutgps

[42] Centro GPS de Danish, Universidad de Aalborg
http://gps.aau.dk/

[43] Documentación acerca de EDK y de todas las herramientas de desarrollo de Xilinx
http://www.xilinx.com/ise/embedded/edk_docs.htm

[44] Navegación satelital.
http://www.globalsecurity.org/space/systems/nav.htm

[45] Página oficial de la NASA donde podemos ver el seguimiento de los satélites en tiempo real
http://science.nasa.gov/realtime/

[46] Receptor GNSS basado en software.
http://www.gpsworld.com/gpsworld/Innovation/The-Software-GNSS-Receiver/ArticleStandard/Article/detail/141119#

[47] Seguimiento de satélites GPS y GLONASS en tiempo real de la FAA.
http://edu-observatory.org/gps/tracking.html

[48] Sitio oficial GNSS de la Universidad de Colorado, aquí encontramos detalles del GN3S
http://ccar.colorado.edu/gnss/