



INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE INGENIERÍA MECÁNICA Y ELÉCTRICA

SECCIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

*“DISEÑO E IMPLEMENTACIÓN DE UN MEDIDOR FASORIAL SINCRONO
NORMALIZADO CON EL ESTANDAR IEEE C37.118”.*

T E S I S

QUE PARA OBTENER EL GRADO DE:

MAESTRO EN CIENCIAS CON ESPECIALIDAD

EN INGENIERÍA ELÉCTRICA

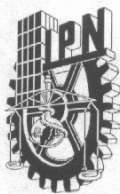
PRESENTA:

ING. BLANCA VERÓNICA HERNÁNDEZ GÓMEZ.

Dr. Raúl Ángel Cortés Mateos
Director de Tesis.

MÉXICO D.F. 2009





SIP-14

INSTITUTO POLITECNICO NACIONAL
SECRETARIA DE INVESTIGACION Y POSGRADO

ACTA DE REVISION DE TESIS

En la Ciudad de México, D. F. siendo las 13:00 horas del día 3 del mes de Marzo 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. ZAC para examinar la tesis de grado titulada:

**“DISEÑO E IMPLEMENTACIÓN DE UN MEDIDOR FASORIAL SÍNCRONO NORMALIZADO
CON EL ESTANDAR IEEE C37.118 ”**

Presentada por el alumno:

HERNÁNDEZ

Apellido paterno

GÓMEZ

materno

BLANCA VERÓNICA

nombre(s)

Con registro:

A	0	7	0	4	0	4
---	---	---	---	---	---	---

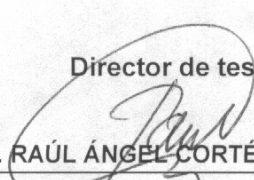
aspirante al grado de:

MAESTRO EN CIENCIAS CON ESPECIALIDAD EN INGENIERÍA ELÉCTRICA

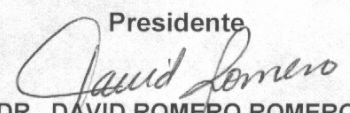
Después de intercambiar opiniones los miembros de la Comisión manifestaron **SU APROBACION DE LA TESIS**, en virtud de que satisface los requisitos señalados por las disposiciones reglamentarias vigentes.

LA COMISIÓN REVISORA

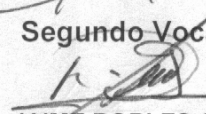
Director de tesis


DR. RAÚL ÁNGEL CORTÉS MATEOS

Presidente


DR. DAVID ROMERO ROMERO

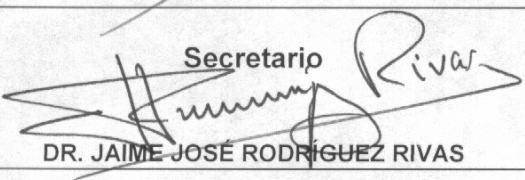
Segundo Vocal


DR. JAIME ROBLES GARCÍA

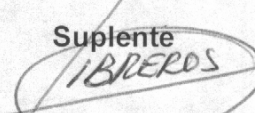
Tercer Vocal


DR. DAVID SEBASTIAN BALTAZAR

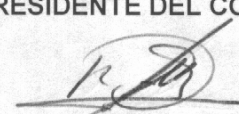
Secretario


DR. JAIME JOSÉ RODRÍGUEZ RIVAS

Suplente


DR. DOMITILO LIBREROS

EL PRESIDENTE DEL COLEGIO


DR. JAIME ROBLES GARCÍA





INSTITUTO POLITECNICO NACIONAL
COORDINACION GENERAL DE POSGRADO E INVESTIGACION

CARTA SESION DE DERECHOS

En la Ciudad de México, Distrito Federal, el día 23 del mes Junio del año 2009 el (la) que suscribe **Ing. Blanca Verónica Hernández Gómez** alumno(a) del Programa de Maestría en Ciencias con especialidad en Ingeniería Eléctrica con número de registro A070405 adscrito a la Sección de Estudios de Posgrado e Investigación de la ESIME Unidad Zacatenco, manifiesta que es autor(a) intelectual del presente Trabajo de Tesis bajo la dirección del **Dr. Raúl Ángel Cortes Mateos** y cede los derechos del trabajo intitulado: **“Diseño e implementación de un medidor fasorial síncrono normalizado con el estándar C37.118”**, al Instituto Politécnico Nacional para su difusión, con fines académicos y de investigación.

Los usuarios de la información no deben reproducir el contenido textual, graficas o datos del trabajo sin el permiso expreso del autor y/o director del trabajo. Este puede ser obtenido escribiendo a la siguiente dirección: hergo2@hotmail.com; rcortes@ipn.mx.

Si el permiso se otorga, el usuario deberá dar el agradecimiento correspondiente y citar la fuente del mismo.

Blanca Verónica Hernández Gómez

Nombre y firma

Resumen

En este trabajo se presenta la implementación de una unidad de medición fasorial (PMU) sincronizado mediante un modulo receptor del sistema satelital de posicionamiento global (GPS) y se verifica que cumpla con las normas del estándar IEEE C37.118 utilizado por los PMU comerciales, instalados en CFE.

El PMU implementado tiene la capacidad de medir los fasores de voltaje y corriente de un sistema trifásico y la frecuencia del sistema. Los fasores calculados, el estampado de tiempo y la frecuencia calculada, son enviados al sistema de control mediante el estándar IEEE C37.118.

Se analizan diferentes algoritmos para el cálculo de fasores a partir de una señal discreta, se describen los algoritmos matemáticos y se justifica la utilización de la Transformada Discreta de Fourier para la estimación de fasores en el PMU estandarizado.

Se reportan los resultados, en tiempo real, de las pruebas realizadas al PMU conforme lo indica el estándar. La respuesta del algoritmo de la Transformada Discreta de Fourier (DFT), la respuesta del algoritmo para determinar la frecuencia ante señales de entrada no ideales y la sincronización con un pulso por segundo (PPS) que envía el GPS.

Finalmente, después de realizar las pruebas que indica el estándar se llega a la conclusión de que el PMU implementado si cumple con el estándar IEEE C37.118.

Abstract

This paper presents the implementation of a phasor measurement unit (PMU) synchronized by a global positioning system (GPS) in power system applications and verified that comply with the norms of the standard IEEE C37.118 used by commercial PMU, installed in CFE.

The implemented PMU has the ability to measure voltage and current phasor a three-phase system and the system frequency. The phasor calculated, timetag and frequency are sent in real-time to the control system using the IEEE standard C37.118.

Different algorithms for calculating phasor are analyzed, and justified the use of Discrete Fourier Transform to estimating phasors in PMU.

The results are reported, in real time, of the tests realized to the PMU as indicates it the standard. The answer of the algorithm of the Discrete Fourier Transform (DFT), the answer of the algorithm to determine the frequency and the synchronization with a pulse per second (PPS) that sends the GPS

Finally, after testing indicated by standard is concluded that the PMU implemented if it complies with the standard IEEE C37.118.

TABLA DE CONTENIDO

Resumen.....	i
ÍNDICE DE FIGURAS.....	vii
ÍNDICE DE TABLAS	xi
GLOSARIO	xii
CAPÍTULO 1.-Introducción	1
1.1 Definición del problema.....	1
1.2 Objetivo	2
1.3 Justificación.....	2
1.4 Estado del arte.....	3
1.5 Aportaciones.....	5
1.6 Alcances y Limitaciones.....	6
1.7 Aplicaciones.....	6
1.8 Estructura del trabajo de tesis.....	8
CAPÍTULO 2.-Algoritmos para Medición de Fasores.....	11
2.1 Introducción.....	11
2.2 Conceptos básicos.....	13
2.2.1 Muestreo digital	13
2.2.2 Teorema del muestreo.....	14
2.3 Algoritmos trigonométricos.....	15
2.4 Algoritmos de mínimos cuadrados	19
2.4.1 Algoritmo de mínimos cuadrados con 3 muestras	20
2.4.2 Algoritmo de mínimos cuadrados con 11 muestras y con más componentes en el modelo	25
2.5 Transformada discreta de Fourier.....	29
2.5.1 Funciones base de la Transformada Discreta de Fourier.....	31
2.5.2 DFT por correlación.....	33
2.6 Algoritmos para la medición de frecuencia	36
2.6.1 Algoritmo de la DFT para cálculo de la frecuencia.....	36
2.6.2 Transformada Rápida de Fourier para el cálculo de la frecuencia.....	39
2.6.2.1 Algoritmo de la FFT	40
2.6.3 Cálculo de la frecuencia por cruce de cero	42
CAPÍTULO 3.- Diseño De Software.....	46
3.1 Ambiente de desarrollo integral (IDE) del microcontrolador	46
3.1.1 Software del microcontrolador de 32 bits V2 ColdFire.....	47

3.1.2	Crear nuevo proyecto para V2 ColdFire.....	48
3.2	Diagrama esquemático del PMU.....	50
3.3	Sistema de Posicionamiento Global.....	54
3.4	Medición.	56
3.4.1	Programa principal.....	59
3.4.2	Estimación de componentes simétricas.	64
3.4.3	ADC.	66
3.4.4	Frecuencímetro.	68
3.4.5	Temporizador para el periodo de muestreo.	72
3.5	Comunicaciones.....	73
3.5.1	Programa principal de comunicaciones.	75
3.5.2	Comprobación de redundancia cíclica.	80
 CAPÍTULO 4.- Pruebas del PMU Implementado para verificar que cumple con el		
estandar IEEE C37.118		83
4.1	Introducción.....	83
4.2	Medición de frecuencia.....	84
4.2.1	Fasores de una señal con frecuencia nominal.	90
4.2.2	Fasores de una señal con frecuencia fuera de la nominal.	92
4.3	Respuesta en el tiempo del PMU	97
4.4	Límites de precisión.....	98
4.4.1	Cambio de magnitud en la entrada del PMU.....	100
4.4.2	Cambio del ángulo de fase.....	104
4.4.3	Distorsión armónica.	106
4.4.4	Señal de Interferencia.....	111
4.4.5	Cambios en la frecuencia.....	113
4.5	Comunicación.....	114
 CAPÍTULO 5.- Conclusiones y Recomendaciones para trabajos futuros.		118
5.1	Conclusiones.....	118
5.2	Recomendaciones para trabajos futuros.	119
 Bibliografía.....		120
 APÉNDICE A.-UML (Lenguaje unificado de modelado)		125
A.1	Diagramas UML	125
A.2	Diagramas de Clase.....	126
A.3	Diagrama de objetos.	127
A.4	Diagramas de caso de uso	127
A.5	Diagrama de secuencias.	128
A.6	Diagrama de actividades.	129
A.7	Paquetes.....	130
A.8	Envío de Mensajes.....	131

APÉNDICE B.- Hardware.....	132
B.1 Introducción.....	132
B.2 68K ColdFire microprocesador M52233DEMO.....	132
B.2.1 Diagrama a Bloques.	134
B.2.1.1 ADC.....	136
B.2.1.2 Temporizador.	137
B.2.1.3 Pines de entrada y salida de propósito general (GPIO).....	138
B.2.1.4 Interfaz de comunicación serial.....	138
B.3 Receptor GPS.....	139
B.3.1 Antena del GPS.....	140
B.3.2 Módulo receptor.	140
B.3.3 Pulso Por Segundo.....	142
B.4 Adecuador de señal	142
B.4.1 Especificaciones Eléctricas.....	143
B.4.2 Acondicionamiento de señales de voltaje de CA.....	143
B.4.2.1 Análisis en CA	144
B.4.2.2 Análisis en CD:.....	144
B.5.3 Acondicionamiento de señales de corriente alterna.	145
B.5.3.1 Análisis en CA:	146
B.5.3.2 Análisis en CD.....	147
APÉNDICE C.- Estándar IEEE C37.118.....	149
C.1 Introducción	149
C.1.2 Descripción.....	150
C.1.3 Alcance	150
C.1.4 Propósito	151
C.2 Necesidades para este estándar	151
C.3 Definiciones:.....	152
C.4.- Medición de sincrofasor.....	153
C.4.1.- Medición para el estampado de tiempo para el sincrofasor.	153
C.4.2.- Sistema de tiempo de sincronización	154
C.5.- Requerimientos de las mediciones del sincrofasor y el cumplimiento de la verificación	154
C.5.1.- Estimación del sincrofasor.....	154
C.5.1.1 Reportando las tramas	155
C.5.1.2 Reportando tiempos.	155
C.5.1.4.- Respuesta en el tiempo del PMU	155
C.5.2.- Límites de precisión	156

C.5.3.- Cumplimiento de verificación.....	156
C.6.- Formatos de mensaje del sincrofasor	157
C.6.1.- Estructura del mensaje.....	158
C.6.1.1.- Mensaje total	159
C.6.1.2 Tiempo y calidad de tiempo.....	159
C.6.1.3.- Ejemplos de salto de segundo para el estampado.....	160
C.6.2 Trama de datos.....	161
C.6.2.1 Explicación de la palabra STAT en la trama de datos.	161
C.6.3 Configuración de trama.....	164
C.6.4 Encabezado.....	164
C.6.5 Trama de comandos.....	165
APÉNDICE D. Precisión y Exactitud.....	181
D.1 Exactitud	181
D.2 Precisión.....	181
D.3 Resolución.....	181
APÉNDICE E.- Código implementado.....	182
E.1 Programa Principal.	182
E.2 Eventos.....	184
E.3 Funciones.	188

ÍNDICE DE FIGURAS

CAPÍTULO 1

Figura 1.- Conjunto de PMU's interconectados	5
--	---

CAPÍTULO 2

Figura 2.1.a.-Corriente alterna sinusoidal.....	12
Figura 2.1.b.- Representación de fasores de la corriente alterna.....	12
Figura 2.2.- 3 muestras de la señal de entrada para el algoritmo.....	15
Figura 2.3.- Diagrama unificar en PSCAD/EMTC.....	16
Figura 2.4.- Corriente de entrada.	17
Figura 2.5 a).- Magnitud estimada.	17
Figura 2.5 b).- Ángulo estimado	17
Figura 2.6.- Señal de entrada.....	18
Figura 2.7 a).- Magnitud estimada.	19
Figura 2.7 b).- Ángulo estimado	19
Figura 2.8.- Voltaje de entrada.	24
Figura 2.9 a).- Magnitud estimada.	24
Figura 2.9 b).- Ángulo estimado	24
Figura 2.10 a).- Magnitud estimada.....	25
Figura 2.10 b).- Ángulo estimado	25
Figura 2.11.- Cinco muestras de la señal de entrada.....	26
Figura 2.12.- Señal de entrada.....	28
Figura 2.13 a).- Magnitud estimada.....	28
Figura 2.13 b).- Ángulo estimado.	28
Figura 2.14 a).- Magnitud estimada.....	29
Figura 2.14 b).- Ángulo estimado.	29
Figura 2.15 a).- Magnitud estimada.....	35
Figura 2.15 b).- Ángulo estimado.	35
Figura 2.16.-Diagrama de flujo de la DFT.....	37
Figura 2.17.-Señal de entrada para la FFT	38
Figura 2.18.-Respuesta de la DFT.....	38
Figura 2.19.- La FFT de descomposición. Un punto N señal se descompone en N señales conteniendo cada uno un solo punto ^[27]	39

Figura 2.20.-Diagrama de flujo de la FFT.....	41
Figura 2.21.-Respuesta de la FFT	42
Figura 2.22.- Medición de frecuencia.....	43
Figura 2.23.-Ecuación de la recta para calcular la fracción de tiempo.	44

CAPÍTULO 3

Figura 3.1.-Programa Freescale CodeWarrior.	49
Figura 3.2.-Selección de la tarjeta en uso y medio de programación.	49
Figura 3.3.-Selección de processor expert.	49
Figura 3.4.-Selección de memoria.....	50
Figura 3.5 Diagrama general de un PMU.....	51
Figura 3.6.- Conector J1 de la tarjeta de desarrollo.....	52
Figura 3.7 Diagrama conceptual del hardware/software a implementar.....	53
Figura 3.8 – Diagrama de subsistemas del PMU	54
Figura 3.9.- Diagrama de secuencias del subsistema GPS.....	54
Figura 3.10.- Diagrama de flujo la interrupción externa.	55
Figura 3.11.- Diagrama de flujo de la Recepción del estampado.	55
Figura 3.12.- Diagrama de secuencias de mediciones.	57
Figura 3.13.- Diagrama de flujo del programa principal.	59
Figura 3.14 a).- Diagrama del cálculo del imaginario.	60
Figura 3.14 b).- Diagrama de flujo para calcular el real.....	60
Figura 3.15.- Diagrama de flujo para calcular la raíz.....	61
Figura 3.16.- Calculo del ángulo de un fasor.	61
Figura 3.17.- Calculo del ángulo en los 4 cuadrantes.	62
Figura 3.18.- Diagrama da flujo para el cálculo del ángulo.	63
Figura 3.19.- Diagrama da flujo para el cálculo de componentes simétricas.....	66
Figura 3.20.- Diagrama de flujo del ADC.....	67
Figura 3.21.- Cálculo de la frecuencia cada 8 muestras	70
Figura 3.22.- Cálculo de la frecuencia cada 8 muestras con una señal contaminada de armónicos	70
Figura 3.23.- Diagrama de flujo del frecuencímetro.....	71
Figura 3.24.- Diagrama de flujo de la interrupción “TimerInt” para la frecuencia de muestreo de conversión del ADC.....	72
Figura 3.25.- Diagrama de secuencias de comunicaciones.....	74
Figura 3.26.- Diagrama de flujo del evento que recibe.	78
Figura 3.27.- Diagrama del programa principal de comunicaciones.....	79
Figura 3.28.-Funcionamiento del CRC.	81
Figura 3.29.- Diagrama de flujo del calculo del CRC.....	81
Figura 3.30.- Diagrama de flujo de la función transmite.....	82

CAPÍTULO 4

Figura 4.1.- Representación fasorial de una onda sinusoidal.	84
Figura 4.2.- Cálculo de la frecuencia para una señal sinusoidal pura.....	86
Figura 4.3.- Cálculo de la frecuencia para una señal sinusoidal pura.....	87
Figura 4.4.- Cálculo de los fasores a una frecuencia nominal.	91
Figura 4.5.- Señal con un periodo de T observado en instante que son múltiplos de T0 donde T0 no es un múltiplo entero de T.	92
Figura 4.6.- Cálculo de los fasores a una frecuencia por debajo de la nominal.	93
Figura 4.7.-Rotación del fasor a una frecuencia de 55Hz.	95
Figura 4.8.- fasor a una frecuencia de 61Hz.	96
Figura 4.9.- fasor a una frecuencia de 65Hz.	97
Figura 4.10.-Señal de entrada.....	98
Figura 4.11.-Fasor de magnitud.	98
Figura 4.12.-Fasor cuando hay un cambio de magnitud.	101
Figura 4.13.-TVE debido a la magnitud nivel 0.	102
Figura 4.14.-TVE debido a la magnitud nivel 1.	103
Figura 4.15.-Error debido al ángulo de fase.	105
Figura 4.16.-Señal con 1% de cualquier armónico.	106
Figura 4.17.-Señales con 10% de cualquier armónico y sus fasores.	109
Figura 4.18.-Señales con 1% de una señal de interferencia.	111
Figura 4.19.-Señales con 10% de una señal de interferencia.	112
Figura 4.20.-Conexión de Parámetros.	115
Figura 4.21.-Seleccionando Protocolo.	116
Figura 4.22.-Protocolo e ID del PMU.	116
Figura 4.23.-Datos del PMU.	117
Figura 4.24.-Mediciones en tiempo real.....	117

APÉNDICE A

Figura A.1.-Diagrama de secuencias UML	129
Figura A.2.-Diagrama de actividades UML.	130
Figura A.3.-El paquete UML le permite agrupar los elementos de un diagrama.	130
Figura A.4.- Notas en UML.....	130

APÉNDICE B

Figura B.1.-Tarjeta de desarrollo M52233DEMO.....	132
Figura B.2.- Microprocesador de 80 pines	134
Figura B.3.-Diagrama a bloques del MCF5223x.	134
Figura B.4.-Diagrama a bloques del MCF52233.	135

Figura B.5.- Antena del GPS	140
Figura B.6 Módulo Receptor GPS	141
Figura B.7 Numeración de los pines del conector J4.....	141
Figura B.8.- Circuito acondicionador de señal de VAC.....	143
Figura B.9.- Circuito resultante con fuente de CA en corto	145
Figura B.10.- Circuito acondicionador de señal de I AC	146
Figura B.11.- Circuito resultante con fuente de CD en corto.....	146
Figura B.12.- Circuito resultante con fuente de CA en corto.....	147

ÍNDICE DE TABLAS

CAPÍTULO 3

Tabla 3.1. Significado de los principales campos del renglón 1 de la trama NMEA.	56
Tabla 3.2 Signo del real e imaginario para el calculo del ángulo	62
Tabla 3.3 Organización de la trama de comandos	75
Tabla 3.4 Comandos enviados al PMU	76
Tabla 3.5 Mensaje de configuración del PMU	77

CAPÍTULO 4

Tabla 4.1 Cálculo de la frecuencia con una señal sinusoidal pura	85
Tabla 4.2 Cálculo de la frecuencia con una señal sinusoidal pura más 20% del séptimo y onceavo armónico	86
Tabla 4.3 Cálculo de la frecuencia con una señal sinusoidal con ruido	88
Tabla 4.4 Cálculo de la frecuencia cada 4 muestras	88
Tabla 4.5 Cálculo de la frecuencia cada 8 muestras	89
Tabla 4.6 Cálculo de la frecuencia cada 16 muestras.....	89
Tabla 4.7 Reportando mediciones 10 veces por segundo.....	94
Tabla 4.8.- Límites de error admisibles para el cumplimiento de los niveles 0-1 ..	99
Tabla 4.9.- TVE del cambio en magnitud para el nivel 0.....	102
Tabla 4.10.- TVE del cambio en magnitud para el nivel 1	104
Tabla 4.11.- TVE del cambio ángulo en el nivel 0 y 1	105
Tabla 4.12.- TVE de una señal con distorsión armónica del 1% para el nivel 0 .	107
Tabla 4.13.- TVE de una señal con distorsión armónica del 10% para el nivel 1	110
Tabla 4.14.- TVE con 1%de una señal de interferencia para el nivel 0.....	112
Tabla 4.15.- TVE con 10%de una señal de interferencia para el nivel 1	113
Tabla 4.16.- TVE con 10%de una señal de interferencia para el nivel 1	113
Tabla 4.17.- TVE con 10%de una señal de interferencia para el nivel 1	114

APÉNDICE B

Tabla B.1.- Características básicas del microcontrolador.....	133
Tabla B.2 descripción del conector j4.....	141

GLOSARIO

A.	Amperes.
ADC	Convertidor Analógico a Digital.
ALU	Unidad aritmética lógica.
AN0-PAN0	Canal 1 del ADC utilizado para medición del voltaje fase A.
AN1-PAN1	Canal 2 del ADC utilizado para medición del voltaje fase B.
AN2-PAN2	Canal 3 del ADC utilizado para medición del voltaje fase C.
AN3-PAN3	Canal 1 del ADC utilizado para medición de corriente fase A.
AN4-PAN4	Canal 2 del ADC utilizado para medición de corriente fase B.
AN5-PAN5	Canal 3 del ADC utilizado para medición de corriente fase C.
BCD	Código binario decimal
CA	Corriente alterna.
CD	Corriente directa
CENAM	Centro Nacional de Metrología
CNM	Centro Nacional de Metrología.
CMOS/TTL	Tecnologías utilizadas para crear circuitos integrados.
Cos	Coseno.
CRC	Comprobación de redundancia cíclica.
CRC-CCITT	Comprobación de redundancia cíclica de 16-bits
CUT	Tiempo Universal Coordinado.
c_k	Forma cosenoidal.
c_n	Coeficientes complejos de Fourier.
DFT	Transformada Discreta de Fourier.
DSP	Procesador de señales digitales
DC	Concentrador de Datos

DoD	Departamento de defensa de US.
DoT	Departamento de transporte de US
f	Frecuencia base.
f_{act}	Frecuencia actual o medida.
f_{max}	Frecuencia máxima a muestrear.
FFT	Transformada Rápida de Fourier.
GMT	Tiempo promedio del observatorio de Greenwich.
GPIO	Pines de entrada y/o salida de propósito general.
GPS	Sistema de Posicionamiento Global.
Hz	Hertz.
i	Coeficiente para puntos de la señal seno y coseno.
I	Magnitud de corriente.
IDE	Ambiente de desarrollo integral.
IDCODE	Código de identificación del PMU
INMARSAT	Sistema Internacional de Telecomunicaciones Marítimas por satélite
IRIG-B	Grupo Interamericano gama de instrumentación formato de código de tiempo B
$\text{Im } \bar{X}$	Amplitud escalada de la forma seno.
$\text{Im } X[]$	Parte imaginaria del dominio de la frecuencia.
IPR	Unidad de generación Registro de Prioridad de Interrupción
ITCN	módulo de control de interrupciones ()
j	Número complejo.
JTAG	Puerto para prueba del núcleo.
J4	Conector del módulo receptor GPS.
k	Coeficiente para cada una de las frecuencias.
k_n	Coeficientes auxiliares en obtención de la serie de Fourier.

M	Magnitud en forma polar.
N	Número de muestras del muestreo digital.
NAVSTAR	Sistema de navegación satelital.
NMEA	Protocolo de comunicación utilizado en el envío de datos al módulo receptor GPS.
NTP	Protocolo de tiempo de red.
PC	Computadora personal.
PDC	Concentrador de datos fasoriales
PMU	Unidad de medición Fasorial.
pps	Pulso por segundo.
PSCAD/EMTC	Diseño Asistido por computador de Sistemas de Potencia
p. u.	Por unidad.
RAM	Memoria de acceso aleatorio.
$\text{Re } X[]$	Parte real del dominio de la frecuencia.
$\text{Re } \bar{X}[k]$	Amplitud escalada de la forma coseno.
rms	Raíz media cuadrática
RS-232	Estándar que norma la comunicación serial.
SC	Comunicación Serial.
SC0	Puerto Serial 0 del Microcontrolador DSP.
SC1	Puerto Serial 1 del Microcontrolador DSP.
SCADA	Supervisory Control and Data Acquisition
SEL	Schweitzer Engineering Laboratories
sen	Seno.
SOC	Segundo de siglo
s_k	Forma senoidal.
t	Tiempo

TC3	Pin del microcontrolador DSP que se utiliza para recibir el pulso por segundo.
THD	Distorsión armónica.
TUC	Tiempo Universal Coordinado.
TVE	Vector total de error.
UART	Transmisor-Receptor Asíncrono Universal
UML	Lenguaje unificado de modelado.
UT	Tiempo Universal.
UTC	Tiempo Universal Coordinado.
V_p	Voltaje poco.
V	Magnitud de voltaje.
v	volts.
WAMS	Wide Area Monitoring Systems
$x(t)$	Série temporal continua.
$X[]$	Dominio de la frecuencia.
$x[i]$	Señal resultante de la inversa de la transformada discreta de Fourier.
x_r	Serie temporal discreta.
$x_0, x_1, x_3, \dots$	Componentes de la serie temporal discreta.
Δs	Intervalo de muestreo.
Δf	Frecuencia de muestreo.
θ	Ángulo de fase.
μs	micro segundos
ω	Frecuencia angular

CAPÍTULO 1.-Introducción

1.1 Definición del problema

La distribución de los fasores de tensión e intensidad a través de la red es una de las informaciones más importantes de que dispone el operador para conocer el margen de operación estable[1].

La mayor parte de las herramientas existentes actualmente estiman dicha distribución a partir de medidas no sincronizadas realizadas en diversos puntos de la red. La medición de los fasores de tensión en tiempo real y no simplemente estimado mejora la capacidad de respuesta de los operadores y permite el desarrollo de sistemas de prevención de inestabilidades y de respuesta automática más rápida y eficaz[1].

Numerosos esfuerzos se han realizado para afrontar el problema de estimación de estado considerando el caso de sistemas no estacionarios [2], sistemas con componentes armónicas [3], sistemas no balanceados [4] y la inclusión de mediciones fasoriales de corriente y voltaje. De todas estas mejoras, la medición fasorial ha sido la de mayor impacto en los sistemas de registro de datos y control de supervisión (SCADA supervisory control and data acquisition) comerciales, gracias a la incipiente masificación y desarrollo la tecnología PMU.

Los PMU permiten obtener mediciones fasoriales de voltaje y corriente sincronizadamente en distintos puntos de la red y transmitir la información de los fasores al centro de control, donde pueden ser comparadas, evaluadas y procesadas [5].

La capacidad de proveer mediciones sincronizadas en la red trae consigo muchas ventajas. La principal es que el comportamiento dinámico del sistema puede ser observado, lo cual puede ser utilizado para un modelado preciso de la carga frente a perturbaciones de voltaje y frecuencia. Esto ha abierto un nuevo campo para la investigación de nuevos algoritmos de control dinámico de sistemas.

1.2 Objetivo

Diseñar e implementar un medidor fasorial síncrono capaz de proporcionar mediciones fasoriales de voltaje y corriente sincronizado vía satélite mediante un módulo receptor GPS y transmitir la información de los fasores a un centro de control. La transmisión de información del PMU se realiza mediante el protocolo del estándar IEEE C37.118.

1.3 Justificación.

Las aplicaciones de monitoreo y protección de sistemas requiere el intercambio de información entre equipos y aplicaciones, no solo procedente de diferentes fabricantes sino instalados en redes de transmisión de diferentes compañías. En cada caso los procedimientos utilizados para el cálculo, registro y transmisión de los fasores de voltaje y corriente se realizan de diferente forma[1].

Por esa razón es necesario disponer de un estándar que proporcione no solo un método y una fuente común de sincronismo para el fasor de referencia sino también un protocolo común de comunicaciones para el intercambio de

información, así como el criterio para el establecimiento y comparación de precisión y verificación de conformidad [6].

La medición de fasores sincronizados tiene un gran potencial de aplicaciones en sistemas eléctricos de potencia, por mencionar algunos[7, 8]:

- Medición de frecuencia y magnitud de Voltajes y corrientes
- Estimación de estado
- Predicción de inestabilidad.
- Control y monitorización
- Monitorear los ángulos de voltaje y corriente

Estas aplicaciones requieren que el PMU se comuniquen con un equipo maestro y, por lo cual, el enlace de comunicación se ha convertido en una preocupación importante [7]. El problema consiste en implementar un medidor fasorial sincronizado vía satélite y normalizado con el estándar IEEE C37.118

1.4 Estado del arte.

En 1983 Phadke, Thorp y Adamiak [9, 10] describen las bases teóricas y el proceso computacional para el cálculo de fasores en tiempo real. La Transformada Discreta de Fourier se utiliza para la obtención de fasores.

En 1988 en el Tecnológico de Virginia se construye la primera Unidad de Medición Fasorial. Esta unidad se sincroniza mediante el Sistema satelital de

Posicionamiento Global. Esta PMU empieza a ser comercializada por la compañía Macrodyne [9].

En 1993 Phadke describe el PMU y además menciona las diferentes formas de sincronización de las mismas, destacando la forma de sincronización del Sistema de Posicionamiento Global, que tiene una exactitud de $1\ \mu\text{s}$ [10, 11].

En 1994 se utilizan los PMU's sincronizados vía satélite para la obtención de fasores en la red la compañía de Potencia Georgia [12].

En ese mismo año se utilizan los PMU's sincronizados vía satélite para confirmar la ecuación del flujo de potencia en líneas de transmisión [13].

Las aplicaciones de los PMU's sincronizados vía satélite se extendió desde entonces, utilizándose para estimación de estado, predicción de estabilidad de líneas, control y monitoreo [14, 15].

En el año 2000, Yutaka Ota y Hideki Fujita utilizan unidades de medición fasorial para llevar a cabo la evaluación de la estabilidad de un sistema de potencia, para llevar a cabo las mediciones se utiliza un microcontrolador DSP y además se menciona que la fuente de sincronización es el pulso por segundo que proporciona un receptor del Sistema de Posicionamiento Global [16].

En el 2004 se desarrolla en México un PMU sincronizado vía satélite por Rafael San Vicente, en su aplicación utiliza un microcontrolador DSP de Texas instruments y se describe el proceso completo de medición [17, 18, 19].

En el 2005 se desarrolla en México un PMU sincronizado vía satélite por el M. en C: Carlos Cuvas Castillo, en su aplicación utiliza un microcontrolador DSP de freescale y se describe el proceso completo de medición [20].

Actualmente la compañía SEL cuenta con dispositivos de medición de fasores sincronizados, instalados en el sistema eléctrico de potencia de los Estados Unidos de Norteamérica, Monterrey y ciudad de México [21].

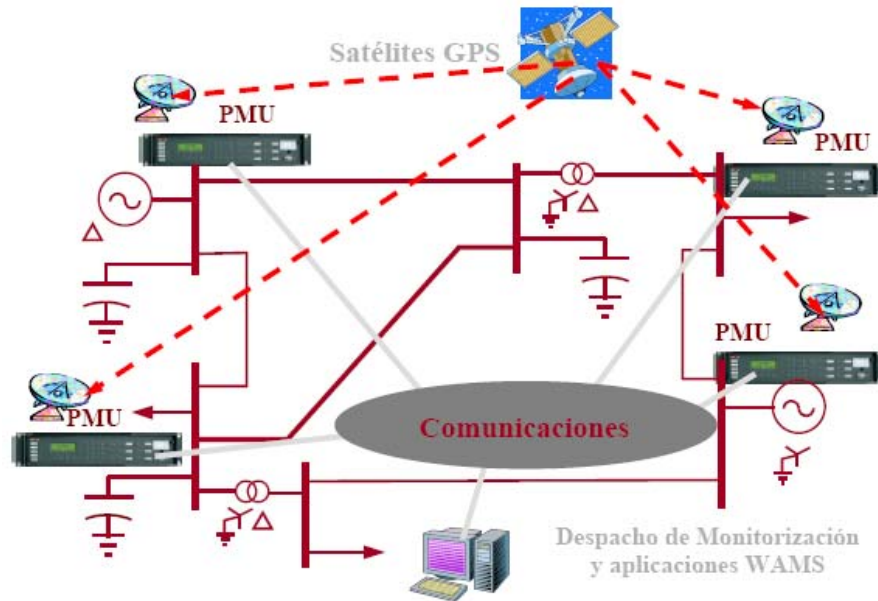


Figura 1.- Conjunto de PMU's interconectados

1.5 Aportaciones.

- Proporcionar al laboratorio de Sistemas Digitales de la SEPI-ESIME-ZAC una unidad de medición fasorial que realice mediciones fasoriales sincronizadas mediante un GPS estandarizado con el protocolo IEEE C37.118 para realizar futuras investigaciones.
- Proporcionar diagramas de flujo, plantillas y algoritmos en lenguaje C, mediante el lenguaje de modelado unificado (UML) que puedan ser modificados y/o adecuados [22] a una aplicación específica de:

- o Procesamiento de señales digitales en tiempo real.
- o Sincronización mediante un pulso por segundo generado por un GPS.

1.6 Alcances y Limitaciones.

El PMU implementado será capaz de realizar mediciones fasoriales de voltaje y de corriente trifásicas, además; realizara los cálculos de las componentes simétricas, las cuales son útiles para tratar con circuitos trifásicos desbalanceados (cuando se tiene una frecuencia diferente a 60 Hertz).

El rango de mediciones de voltaje será de 12.7V a 152.4V, con lo que se tiene del 10 al 120% de 127V y que cumple con el estándar.

La sincronización se realizara mediante el PPS que proporciona el GPS. El GPS proporciona también el UTC el cual se utiliza para el estampado de tiempo de las mediciones. Se enviaran de 10 a 30 tramas por segundo, según lo indicado por el estándar IEEE C37.118.

1.7 Aplicaciones.

Las compañías eléctricas tiene que enfrentarse a problemas debido a la evolución de los sistemas eléctricos de potencia, hay dos que destacan especialmente: la previsión de que las redes eléctricas funcionarán más cerca de su capacidad máxima y la necesidad de una supervisión mejor y más precisa de las redes eléctricas. Determinar los fasores a lo largo del sistema de distribución de energía en un momento dado puede solucionar estos problemas [46].

Las unidades de medición fasorial sincronizada vía satélite, encuentran su principal aplicación en los sistemas de distribución de energía [23, 24 ,25], y se pueden agrupar de la siguiente forma:

- Supervisión de sistemas de distribución de energía
- Protección avanzada de redes
- Esquemas de control avanzado

Actualmente las compañías eléctricas, para supervisión de sistemas de distribución de energía utilizan la estimación de estado del sistema de distribución a partir de las mediciones en tiempo real

Hasta hace poco, los conjuntos de medición disponibles no contenían mediciones de ángulo de fase. Con la llegada del PMU esto cambió [26]. Las mediciones sincronizadas de fasores abren una posibilidad totalmente nueva para sintetizar el proceso de estimación de estado [25].

Las compañías eléctricas estarán en condiciones de realizar, en tiempo real, avanzados análisis de los imprevistos estáticos y dinámicos que se producen sus redes.

Otro grupo de aplicaciones de la medición sincronizada de fasores se caracteriza por aumentar la eficacia de la protección de los sistemas de distribución de energía, es decir, la protección de los sistemas y equipos, y la preparación de esquemas de medidas correctoras. Por ejemplo, la protección tradicional de líneas se basa en realizar mediciones de ciertas magnitudes del sistema en un extremo de la línea para determinar si se ha producido una avería. En el caso de las líneas críticas, las mediciones se sincronizan mediante algún mecanismo que proporcione esquemas de protección diferencial para la detección de averías. Se considera que la protección

diferencial es la forma de protección más fiable. En el futuro, las unidades PMU podrían ser utilizadas para suministrar protección diferencial [23].

En un estudio sobre la protección con relés de pérdida de sincronismo adaptativo [27] se presenta un ejemplo de las mediciones de fasores utilizadas para la protección. Este trabajo ha demostrado que es posible diseñar relés de pérdida de sincronismo, mejorados, utilizando la medición en tiempo real de ángulos de fasores en los puntos clave de la red y aplicando los conceptos del análisis de estabilidad transitoria.. Una instalación de características similares es el 'Plan de Defensa' de Electricité de France (EDF) [28]. En este esquema se comparan las mediciones de fasores en diferentes regiones de Francia; cuando se sobrepasa un límite predefinido se emite la orden de desconexión a la subestación apropiada, quedando aislada la parte de la red que podría provocar el colapso total del sistema de distribución de energía.

1.8 Estructura del trabajo de tesis.

En este capítulo 1, se aborda el problema a resolver; exponiendo el objetivo, justificación, estado del arte, aportaciones y estructura de la tesis. Se da una breve reseña general de lo que se conoce, de los elementos más importantes empleados en la solución del problema; se plantea el alcance y limitaciones; se describe el problema a resolver; se presentan los antecedentes y lo que se conoce en la actualidad del tema. Se mencionan las aplicaciones más importantes del PMU y finalmente, se describe la estructura del trabajo por capítulos.

En el capítulo 2 se analizan diferentes algoritmos para el cálculo de fasores a partir de una señal discreta, se describe los algoritmos matemáticos y se implementan en MATLAB; se prueban con una señal con sobrecarga trifásica simulada con el programa de diseño de sistemas de potencia asistido por computadora (PSCAD/EMTC) que genera un archivo con los datos de la señal discreta; y en otro caso se genera una señal contaminada de armónicos y ruido, se analizan los fasores estimados por cada algoritmo y se justifica la utilización de la Transformada Discreta de Fourier para la estimación de fasores en el PMU estandarizado.

En el capítulo 3 se presenta el diagrama a bloques de un PMU y el diagrama conceptual a implementar, se trabaja con diagramas UML (Lenguaje unificado de modelado) para expresar de manera clara el funcionamiento de cada bloque del PMU. Se describe el ambiente de desarrollo integral (IDE) del microcontrolador y el procedimiento de inicialización de periféricos mediante técnicas de objetos encapsulados “beans”.

En el capítulo 4 se analiza el PMU normalizado e implementado para verificar que cumple con el estándar IEEE C37.118.

Se reportan los resultados, en tiempo real, de las pruebas realizadas al PMU conforme lo indica el estándar. La respuesta del algoritmo de la DFT, la respuesta del algoritmo para determinar la frecuencia ante señales de entrada no ideales y la sincronización con el Pulso por segundo que envía el GPS.

En el capítulo 5 se dan las conclusiones de la implementación y las recomendaciones para trabajos futuros con esta línea de estudio.

En la parte final del trabajo se muestran los apéndices; en el apéndice A se da un resumen lo que es la programación con lenguaje unificado de modelado, en el apéndice B se muestra el hardware del PMU, en el apéndice C se incluye la programación realizada en el microcontrolador.

En el apéndice B.-Se presenta el hardware utilizado para la implementación. Los principales elementos utilizados en el medidor fasorial sincronizado. Tarjeta de desarrollo M52233DEMO con un microcontrolador de 32 bits ColdFire V2 de Freescale, el GPS y la tarjeta de adecuación de señales de voltaje y corriente.

En el apéndice C se muestra un breve resumen del contenido del estándar IEEE C37.118.

En el apéndice D se muestra una breve definición de exactitud y precisión.

En el apéndice E se presenta el programa implementado que consta de 3 archivos esencialmente, el primero es el programa principal llamado PMU, el segundo es eventos, este contiene el algoritmo de las interrupciones utilizadas en el proyecto y el tercer archivo es de las funciones ocupadas en el proyecto.

CAPÍTULO 2.-Algoritmos para Medición de Fasores.

En este capítulo se analizan los diferentes métodos utilizados en sistemas eléctricos de potencia para calcular fasores de voltaje y corriente, los cuales portan amplitud y fase de una señal. Justificando el uso de la Transformada Discreta de Fourier para la implementación del medidor fasorial que debe cumplir con los requerimientos de las mediciones del síncrofasor (cláusula 5) del estándar IEEE C37.118.

2.1 Introducción

La medición fasorial es muy importante para monitorear y diagnosticar sistemas eléctricos de potencia. Los fasores representan la amplitud y la fase de una señal senoidal en estado estable (frecuencia, amplitud y fase constantes) [16].

Los cálculos de magnitud y fase corresponden a la transformación de coordenadas rectangulares a polares, cuya representación permite la visualización de los desplazamientos de los ángulos de fase.

Las compañías eléctricas tienen que enfrentarse con problemas complejos en una situación de constante evolución de los negocios. Entre estas cuestiones hay dos que destacan especialmente: la previsión de que las redes eléctricas funcionan más cerca de su capacidad máxima y la necesidad de una supervisión más precisa de las redes eléctricas.

La posibilidad de determinar los fasores a lo largo del sistema de distribución o transmisión de energía en un momento dado abre una puerta a la solución de estos problemas, es por ello que es de suma importancia analizar y estudiar las técnicas de estimación de fasores.

Para un mejor análisis de la corriente alterna representada por la figura 2.1.a, expresada como $A \sin(\omega t + \varphi)$ puede transformarse a un sistema de coordenadas polares como se muestra en la figura 2.1.b.

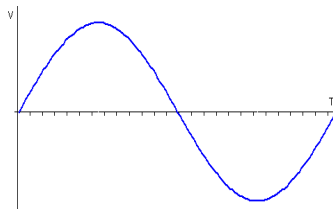


Figura 2.1.a.-Corriente alterna sinusoidal.

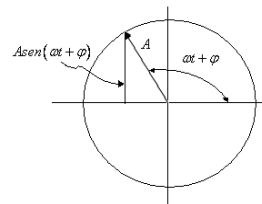


Figura 2.1.b.- Representación de fasores de la corriente alterna.

El radio gira como un vector rotativo con frecuencia ω describiendo un círculo. La longitud del radio representa la amplitud, en este caso de la tensión. La componente vertical tiene el valor $A \sin(\omega t + \varphi)$ en la curva sinusoidal de la corriente alterna (CA). Este radio giratorio es el denominado fasor.

Para calcular los fasores, los algoritmos actuales procesan segmentos de señal (de voltaje o corriente) de duración igual a un múltiplo de ciclo fundamental. A continuación se mencionan los algoritmos estudiados para la estimación de fasores:

- Algoritmos no recursivos.
 - ✓ Algoritmos trigonométricos
 - Miki & Mikano
 - Mann & Morrison

- Rockefeller & Udren
- Gilbert & Shovlin
- ✓ Algoritmos de mínimos cuadrados
- ✓ Algoritmos de correlación
 - Algoritmos de la DFT
 - Funciones pares e impares
 - Algoritmo de la FFT

Los algoritmos recursivos no se estudian ya que estos pueden ir acumulando un error en el cálculo de los fasores.

2.2 Conceptos básicos.

Para implementar cualquier algoritmo de estimación fasorial es necesario estudiar dos conceptos básicos en procesamiento de señales digitales, muestreo digital y el teorema de muestreo.

2.2.1 Muestreo digital

El muestreo digital es un proceso de adquisición de datos, a intervalos de tiempo regulares, consistente en la obtención del valor que toma la señal original en un momento dado [29].

El parámetro fundamental del muestreo digital es el intervalo de muestreo Δ_s , o su equivalente frecuencia de muestreo $1/\Delta_s$ Hz, cuanto menor sea Δ_s , mayor número de valores se tienen de la señal, y viceversa.

El resultado de dicho muestreo es la obtención de una serie discreta ordenada $\{x_r\} = \{x_0, x_1, x_3, \dots, x_r, \dots\}$, en la que el índice r indica la posición de orden temporal del valor x_r . Así, el valor de la señal original, en el tiempo $t = \Delta r$, $x(t)$, se representa por x_r . A la señal continua de origen la llamamos serie temporal continua, mientras que a la serie obtenida por el muestreo la llamamos serie temporal discreta.

2.2.2 Teorema del muestreo.

Según el teorema de muestreo de Nyquist-Shannon, para poder reconstruir con exactitud la forma de una onda es necesario que la frecuencia de muestreo (Δf) sea como mínimo el doble de la máxima frecuencia (f_{max}) a muestrear [29].

El teorema de Nyquist indica el valor mínimo necesario para reconstruir la señal original sin embargo mientras más muestras se tengan, se podrá reconstruir mejor la señal [29]. Por lo tanto, para que el muestreo sea correcto, deberemos escoger la frecuencia de muestreo de tal forma que:

$$\Delta f = 2 \cdot f_{max} \quad (2.1)$$

Además se filtra la señal de entrada para eliminar las frecuencias que no pueden ser detectadas mediante el proceso del muestreo, para evitar ver frecuencias no existentes (alias). Es conveniente que la frecuencia de muestreo sea muy superior al doble de la frecuencia máxima de la señal puesto que ningún filtro pasa-bajos puede eliminar completamente las frecuencias superiores a la frecuencia de corte.

2.3 Algoritmos trigonométricos

El proceso de estimación fasorial de los algoritmos trigonométricos se hace a partir de señales con duración de fracciones de ciclo, solo capturan de 2 a 3 muestras por ciclo [30].

El algoritmo de Miky & Mikano trabaja con 2 muestras de la señal de entrada, la primera v_0 en el instante t_0 y la segunda v_{+1} en el instante $t+1$ con un intervalo de muestreo Δt constante. El algoritmo de Mann & Morrison, Rockefeller & Udren y el de Gilvert & Shovlin trabaja con 3 muestras, la primera v_0 en el instante t_0 , la segunda v_{-1} en el instante $t-1$ y la tercera v_{+1} en el instante $t+1$ con un Δt como se muestra en la figura 2.2 [30].

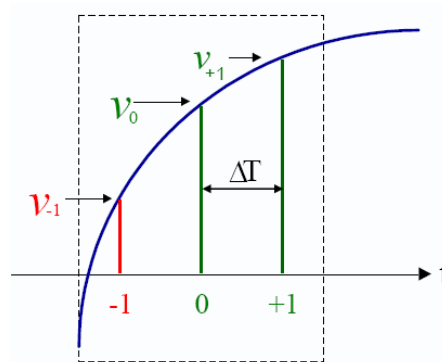


Figura 2.2.- 3 muestras de la señal de entrada para el algoritmo.

Las consideraciones que se deben de tomar para estos algoritmos es que la señal de entrada es sinusoidal a una frecuencia fundamental y no debe existir variación en la señal de entrada. [30]:

Los algoritmos trigonométricos presentan las siguientes caracterizas [30]:

Ventana corta de muestreo (menos de la mitad de un ciclo)

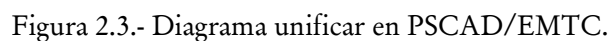
Respuesta transitoria rápida

Mínimo número de cálculos

$$V_p = \sqrt{v_0^2 + \left[\frac{v_0 \cos(\omega \Delta T) - v_{-1}}{\sin(\omega \Delta T)} \right]^2} \quad (2.2)$$

$$\tan(\theta) = \frac{v_0 \sin(\omega \Delta T)}{v_0 \cos(\omega \Delta T) - v_{-1}} \quad (2.3)$$

El PSCAD/EMTC genera un archivo de datos con los valores de voltaje y



Se analiza la respuesta del algoritmo con una señal de corriente de la fase A. En la figura 2.4 se muestra la señal de entrada simulada en el PSCAD/EMTC.

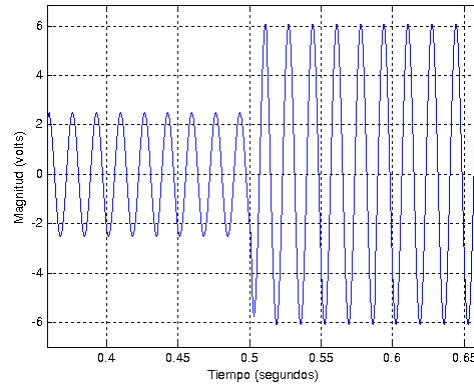


Figura 2.4.- Corriente de entrada.

Como se puede observar en la figura 2.4 la amplitud antes de la sobrecarga es de 2.5 Amperes y después de la sobrecarga es de 6 Amperes, la corriente se incremento en 3,5 Amper después de la sobrecarga.

En la figura 2.5 se muestra la estimación en magnitud para la Ifase A y se puede observar que existe un pico en el momento de la sobrecarga llega hasta 8.5 Amperes, este se estabiliza muy rápidamente en 6 Amperes; el tiempo que tarda en llegar a este valor es de 1.38 ms que es lo que se tarda en tomar 2 muestras.

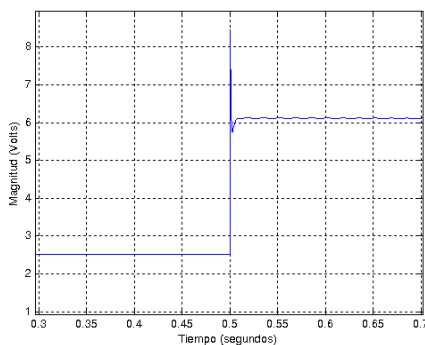


Figura 2.5 a).- Magnitud estimada.

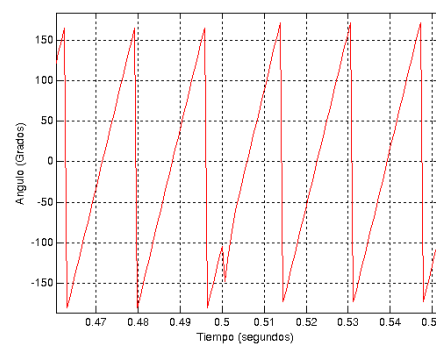


Figura 2.5 b).- Ángulo estimado

Ahora la señal de entrada es una senoidal con una frecuencia fundamental de 60 Hz agregando una señal de segundo, tercero, cuarto y quinto armónico, su ecuación es la 2.4 y La grafica de la señal de entrada se muestra en la figura 2.6; esta señal es utilizada para el análisis de todos los algoritmos mencionados aquí.

$$V(t) = V_p \sin(\omega \Delta t) + V_s \sin(2 * \omega \Delta t) + V_t \sin(3 * \omega \Delta t) + V_c \sin(4 * \omega \Delta t) + V_q \sin(5 * \omega \Delta t) \quad (2.4)$$

Donde:

$$V_p=10, V_s=5, V_t=5, V_c=5 \text{ y } V_q=5$$

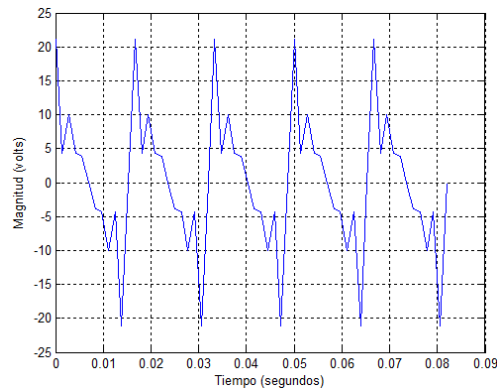


Figura 2.6.- Señal de entrada.

Se muestra en la figuras 2.6 que la señal de entrada esta completamente deforme ya no es puramente senoidal, con la característica que es periódica. La amplitud máxima es de aproximadamente 22 unidades.

En la figura 2.7 se muestra la estimación en magnitud y se puede observar que el fasor en magnitud es erróneo al igual que la estimación de ángulo; esto debido a que los filtros de los algoritmos trigonométricos no eliminan los armónicos.

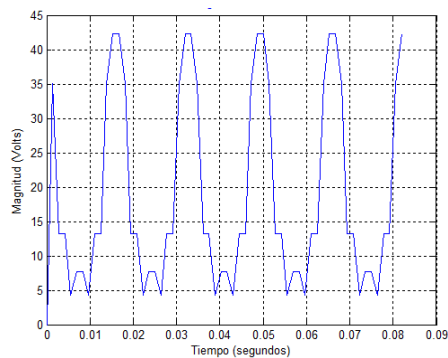


Figura 2.7 a).- Magnitud estimada.

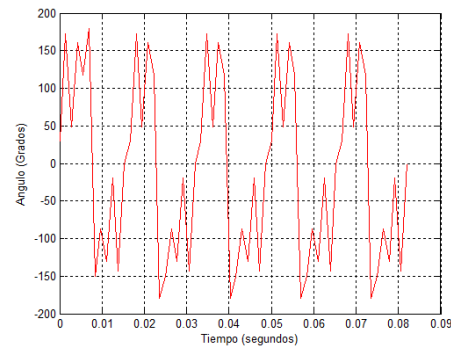


Figura 2.7 b).- Ángulo estimado

Anteriormente en términos generales el sistema eléctrico de potencia se tenía una característica lineal, y se podía trabajar con voltajes y corrientes que en esencia eran de tipo senoide, sin embargo en la actualidad y debido al crecimiento de la electrónica de estado sólido, se han agregado una gran cantidad de cargas no lineales a las redes eléctricas.

Los algoritmos trigonométricos tienen ventajas al calcular fasores, por ejemplo cuando se tiene un transitorio, estos responden rápidamente y se requieren de pocos cálculos; sin embargo cuentan con deficiencias significativas para implementarlos en medidores fasoriales.

Finalmente estos algoritmos no eliminan los armónicos y el ruido afecta su desempeño, es por esa razón que no son adecuados para implementarlos en el PMU y no se analizan en este trabajo.

2.4 Algoritmos de mínimos cuadrados

Mínimos cuadrados es una técnica de optimización matemática que, dada una serie de mediciones, encuentra una función que se aproxime a los datos. Minimiza la

suma de cuadrados de las diferencias ordenadas entre los puntos generados por la función y los correspondientes en los datos.

2.4.1 Algoritmo de mínimos cuadrados con 3 muestras

Para implementar el algoritmo de mínimos cuadrados, primero se establece como esta compuesta la señal de entrada, que se muestra en la ecuación 2.5 [31].

$$v(t) = V_p \sin(\omega t + \theta) \quad (2.5)$$

Desarrollando la ecuación 2.5 con la función trigonométrica de la ecuación 2.6 se tiene la ecuación 2.7:

$$\sin(x + y) = \sin(x)\cos(y) + \cos(x)\sin(y) \quad (2.6)$$

$$v(t) = V_p \{ \sin(\theta)\cos(\omega t) + \cos(\theta)\sin(\omega t) \} \quad (2.7)$$

Se consideran 3 muestras tomadas en intervalos regulares de v_{-1} , v_0 , v_{+1} de la señal, las muestras pueden ser expresadas como tres ecuaciones a continuación descritas [29]:

$$v_{-1}(t) = V_p \{ \sin(\theta)\cos(-\omega\Delta T) + \cos(\theta)\sin(-\omega\Delta T) \} \quad (2.8)$$

$$v_0(t) = V_p \{ \sin(\theta)\cos(0) + \cos(\theta)\sin(0) \} \quad (2.9)$$

$$v_{+1}(t) = V_p \{ \sin(\theta)\cos(\omega\Delta T) + \cos(\theta)\sin(\omega\Delta T) \} \quad (2.10)$$

La frecuencia de la señal de entrada es conocida y definida por el usuario, en este caso se trabaja con una frecuencia fundamental de 60 Hz, la frecuencia de muestreo depende de la frecuencia fundamental y de el número de muestras por ciclo que se deseen, para este trabajo se determinan 12 muestras por ciclo; ahora podemos calcular la frecuencia de muestreo.

- Frecuencia nominal 60Hz.
- Número de muestras por ciclo 12.

De los datos anteriores se calcula la frecuencia de muestreo con la ecuación (2.11).

$$f_m = f_{nominal} * Nm \quad (2.11)$$

$$f_m = 60 * 12 = 720$$

Con la frecuencia de muestreo se calcula la frecuencia angular con la ecuación (2.12)

$$\omega = 2 * \pi * f \quad (2.12)$$

Y el intervalo Δt , para reducir las ecuaciones de las muestras de entrada.

$$\Delta t = \frac{1}{f_m} = \frac{1}{720}$$

$$\omega \Delta t = 2 * \pi * f * \left(\frac{1}{f_m} \right)$$

$$\omega \Delta t = 2 * \pi * 60 * \left(\frac{1}{720} \right)$$

$$\omega \Delta t = \frac{\pi}{6} \text{ rad} = 30^\circ$$

Sustituimos el valor de $\omega\Delta t$ en las ecuaciones de la señal de entrada para calcular las constantes y se tiene:

$$v_{-1}(t) = V_p \left\{ \sin(\theta) \cos\left(-\frac{\pi}{6}\right) + \cos(\theta) \operatorname{sen}\left(-\frac{\pi}{6}\right) \right\}$$

$$v_0(t) = V_p \left\{ \sin(\theta) \cos(0) + \cos(\theta) \operatorname{sen}(0) \right\}$$

$$v_{+1}(t) = V_p \left\{ \sin(\theta) \cos\left(\frac{\pi}{6}\right) + \cos(\theta) \operatorname{sen}\left(\frac{\pi}{6}\right) \right\}$$

Estas ecuaciones pueden ser expresadas en función de dos variables desconocidas V_p y θ . Las tres ecuaciones anteriores pueden ser expresadas en forma matricial, como se muestra en la ecuación 2.13:

$$\begin{bmatrix} -\frac{1}{2} & \frac{\sqrt{3}}{2} \\ 0 & 1 \\ \frac{1}{2} & \frac{\sqrt{3}}{2} \end{bmatrix} \cdot \begin{bmatrix} V_p \cos(\theta) \\ V_p \operatorname{sen}(\theta) \end{bmatrix} = \begin{bmatrix} v_{-1} \\ v_0 \\ v_{+1} \end{bmatrix} \quad (2.13)$$

A partir de la matriz A se obtiene la seudo inversa de la siguiente manera:

$$[A]^{-1L} = \left[\left[[A]^T [A] \right]^{-1} [A]^T \right] \begin{bmatrix} 2 & 0 \\ 0 & \frac{2}{5} \end{bmatrix} \cdot \begin{bmatrix} -\frac{1}{2} & 0 & \frac{1}{2} \\ \frac{\sqrt{3}}{2} & 1 & \frac{\sqrt{3}}{2} \end{bmatrix} = \begin{bmatrix} -1 & 0 & 1 \\ \frac{\sqrt{3}}{5} & \frac{2}{5} & \frac{\sqrt{3}}{5} \end{bmatrix}$$

Las componentes de la parte real e imaginaria del fasor que representa a la onda de voltaje, son obtenidos usando las ecuaciones 2.14:

$$\begin{bmatrix} V_p \cos(\theta) \\ V_p \sin(\theta) \end{bmatrix} = \begin{bmatrix} -1 & 0 & 1 \\ \frac{\sqrt{3}}{5} & \frac{2}{5} & \frac{\sqrt{3}}{5} \end{bmatrix} \cdot \begin{bmatrix} v_{-1} \\ v_0 \\ v_{+1} \end{bmatrix} \quad (2.14)$$

Después que la siguiente muestra es recibida, se hace un cambio y tomando la muestra v_{+1} , como muestra actual, como se muestra a continuación.

$$v_{+1} = \text{muestra actual}$$

$$v_0 = v_{+1}$$

$$v_{-1} = v_0$$

Las ecuaciones del filtro real e imaginario se muestran en las ecuaciones 2.15 y 2.16 respectivamente.

$$V_p \cos(\theta) = v_{+1} - v_{-1} \quad (2.15)$$

$$V_p \sin(\theta) = \frac{\sqrt{3}}{5} v_{-1} + \frac{2}{5} v_0 + \frac{\sqrt{3}}{5} v_{+1} \quad (2.16)$$

Para estimar el fasor en magnitud se eleva al cuadrado la parte real y se suma la parte imaginaria elevada el cuadrado y la resultante de esta operación se le saca la raíz cuadrada obteniendo así la magnitud del fasor. Para calcular el ángulo solo es necesario dividir la parte imaginaria entre la real y sacar la tangente.

Se analiza ahora la estimación del fasor para una señal de entrada proporcionada por el software PSCAD/EMTC, el circuito empleado se muestra en la figura 2.3 y se utiliza ahora el voltaje de la fase A.

La señal del voltaje de entrada en tiempo continuo se muestra en la figura 2.8.

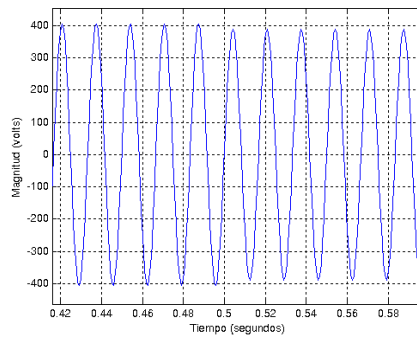


Figura 2.8.- Voltaje de entrada.

En este caso la señal de entrada es el voltaje de la fase A y con una resistencia de sobrecarga cero, como se puede observar en la figura el $V_{\text{faseA}} = 400$ Volts antes de que ocurra la sobrecarga en el instante $t = 0.5$ segundos y después de la sobrecarga el $V_{\text{faseA}} = 390$ Volts

La magnitud del fasor estimado se muestra en la figura 2.9 a y en la figura 2.9 b se muestra este mismo fasor amplificado en el momento de la sobrecarga.

En el momento de la sobrecarga, el fasor estimado tiene un pico negativo que llega hasta los 360 Volts y después estabilizarse en 390 Volts.

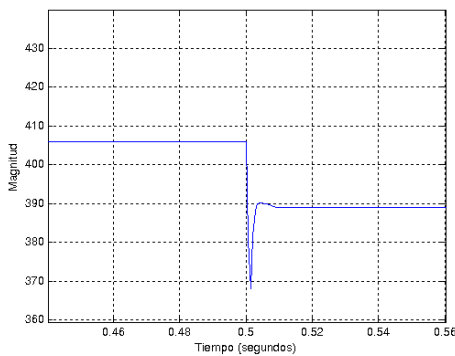


Figura 2.9 a).- Magnitud estimada.

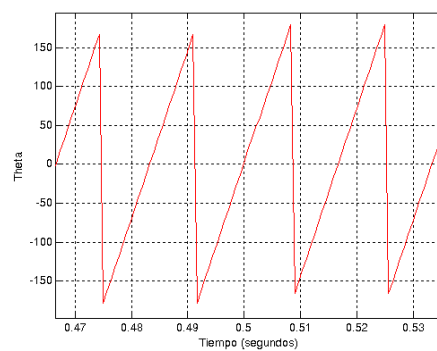


Figura 2.9 b).- Ángulo estimado

Se analiza ahora la estimación del fasor con una señal contaminada de armónicos como la que se muestra en la figura 2.6.

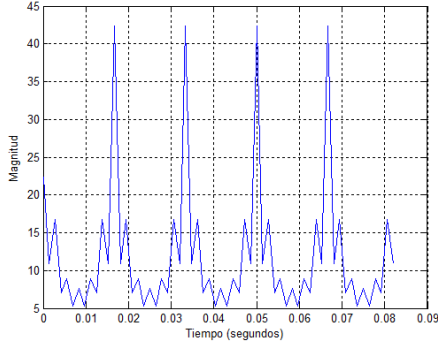


Figura 2.10 a).- Magnitud estimada.

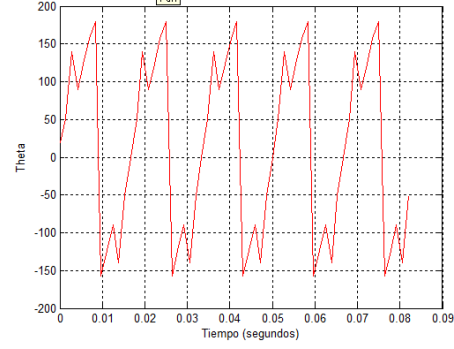


Figura 2.10 b).- Ángulo estimado

Como se puede observar en la figura 2.10, el fasor estimado es erróneo con el algoritmo de mínimos cuadrados con 3 muestras, esto debido a que para la implementación de este algoritmo no toma en consideración ningún armónico.

2.4.2 Algoritmo de mínimos cuadrados con 11 muestras y con más componentes en el modelo

Si la señal de entrada esta compuesta con frecuencias de 60Hz segunda, tercera y cuarta armónica y la componente decreciente de corriente directa (CD). La onda de entrada puede ser expresada como:

$$i(t) = I_{p1} \sin(\omega t + \theta_1) + I_{p2} \sin(2\omega t + \theta_2) + I_{p3} \sin(3\omega t + \theta_3) + I_{p4} \sin(4\omega t + \theta_4) + I_0 e^{-\frac{t}{\tau}} \quad (2.17)$$

Desarrollando la ecuación 2.17 con la función trigonométrica de la ecuación 2.6 tenemos la ecuación 2.18:

$$i(t) = I_{p1} (\sin(\theta_1) \cos(\omega t)) + I_{p1} (\cos(\theta_1) \sin(\omega t)) + I_{p2} (\sin(\theta_2) \cos(2\omega t)) + I_{p2} (\cos(\theta_2) \sin(2\omega t)) + I_{p3} (\sin(\theta_3) \cos(3\omega t)) + I_{p3} (\cos(\theta_3) \sin(3\omega t)) + I_{p4} (\sin(\theta_4) \cos(4\omega t)) + I_{p4} (\cos(\theta_4) \sin(4\omega t)) + I_0 e^{-\frac{t}{\tau}} \quad (2.18)$$

La ecuación 2.13 puede ser expandida utilizando los primeros dos términos de la serie de Taylor para el término exponencial, se obtiene la ecuación 2.13. Se consideran 11 muestras tomadas en intervalos regulares de i_{-5} , i_{-4} , i_{-3} , i_{-2} , i_{-1} , i_0 , i_{+1} , i_{+2} , i_{+3} , i_{+4} , i_{+5} , en la figura 2.11 se muestran los intervalos de muestreo.

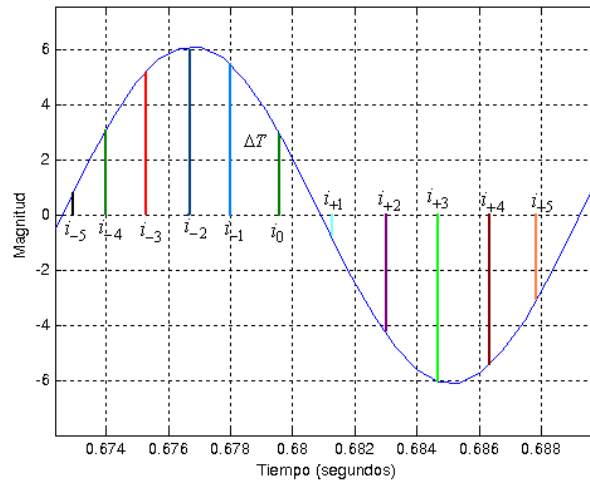


Figura 2.11.- Cinco muestras de la señal de entrada

Al igual que en el caso anterior se tiene una frecuencia de muestreo de 720 Hz, con este dato se calcula la frecuencia angular y se sustituya en la ecuación 1.11, además se sustituyen los intervalos de muestreo teniendo 11 ecuaciones. Estas once ecuaciones lineales que contienen 11 variables desconocidas pueden ser expresadas en forma matricial y así formar la matriz A:

$$\begin{bmatrix} -0.5 & -0.866 & 0.866 & 0.5 & -1 & 0 & 0.866 & -0.5 & 1 & -5 \\ -0.866 & -0.5 & 0.866 & -0.5 & 0 & 1 & -0.866 & -0.5 & 1 & -4 \\ -1 & 0 & 0 & -1 & 1 & 0 & 0 & 1 & 1 & -3 \\ -0.866 & 0.5 & -0.866 & -0.5 & 0 & -1 & -0.866 & -0.5 & 1 & -2 \\ -0.5 & 0.866 & -0.866 & 0.5 & -1 & 0 & -0.866 & -0.5 & 1 & -1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0.5 & 0.866 & 0.866 & 0.5 & 1 & 0 & 0.866 & -0.5 & 1 & 1 \\ 0.866 & 0.5 & 0.866 & -0.5 & 0 & -1 & -0.866 & -0.5 & 1 & 2 \\ 1 & 0 & 0 & -1 & -1 & 0 & 0 & 1 & 1 & 3 \\ 0.866 & -0.5 & -0.866 & 0.5 & 0 & 1 & 0.866 & -0.5 & 1 & 4 \\ 0.5 & -0.866 & -0.866 & 0.5 & 1 & 0 & -0.866 & -0.5 & 1 & 5 \end{bmatrix} \cdot \begin{bmatrix} I_{p1} \cos(\theta_1) \\ I_{p1} \sin(\theta_1) \\ I_{p2} \cos(\theta_2) \\ I_{p2} \sin(\theta_2) \\ I_{p3} \cos(\theta_3) \\ I_{p3} \sin(\theta_3) \\ I_{p4} \cos(\theta_4) \\ I_{p4} \sin(\theta_4) \\ I_0 \\ \frac{I_0}{720\tau} \end{bmatrix} = \begin{bmatrix} i_{-5} \\ i_{-4} \\ i_{-3} \\ i_{-2} \\ i_{-1} \\ i_0 \\ i_{-1} \\ i_{+2} \\ i_{+3} \\ i_{+4} \\ i_{+5} \end{bmatrix}$$

Se despejan las variables desconocidas y de la ecuación resultante se estiman los fasores. Las componentes de la parte real e imaginaria del fasor que representa a la onda de voltaje, son obtenidos usando los primeros dos renglones de la siguiente ecuación.

$$\begin{bmatrix} I_{p1} \cos(\theta_1) \\ I_{p1} \sin(\theta_1) \\ I_{p2} \cos(\theta_2) \\ I_{p2} \sin(\theta_2) \\ I_{p3} \cos(\theta_3) \\ I_{p3} \sin(\theta_3) \\ I_{p4} \cos(\theta_4) \\ I_{p4} \sin(\theta_4) \\ I_0 \\ \frac{I_0}{720\tau} \end{bmatrix} = \begin{bmatrix} 1.0774 & -2.1547 & 2.1547 & -2.1547 & 1.0774 & 0.0000 & -1.0774 & 2.1547 & -2.1547 & 2.1547 & -1.0774 \\ -0.2961 & 0.0278 & -0.0556 & 0.0833 & 0.1850 & 0.1111 & 0.1850 & 0.0833 & -0.0556 & 0.0278 & -0.2961 \\ -0.3943 & 1.0774 & -1.0774 & 0.7887 & -0.6830 & 0 & 0.6830 & -0.7887 & 1.0774 & -1.0774 & 0.3943 \\ 0.2351 & -0.1944 & -0.1111 & -0.0833 & 0.0427 & 0.2222 & 0.0427 & -0.0833 & -0.1111 & -0.1944 & 0.2351 \\ 0.1443 & -0.5387 & 0.7887 & -0.5387 & 0.1443 & 0.0000 & -0.1443 & 0.5387 & -0.7887 & 0.5387 & -0.1443 \\ -0.1518 & 0.2778 & -0.0556 & -0.1667 & 0.0407 & 0.1111 & 0.0407 & -0.1667 & -0.0556 & 0.2778 & -0.1518 \\ -0.0352 & 0.1667 & -0.3591 & 0.4553 & -0.3239 & -0.0000 & 0.3239 & -0.4553 & 0.3591 & -0.1667 & 0.0352 \\ 0.0684 & -0.1944 & 0.2222 & -0.0833 & -0.1240 & 0.2222 & -0.1240 & -0.0833 & 0.2222 & -0.1944 & 0.0684 \\ 0.1592 & 0.0278 & 0.1111 & 0.0833 & 0.0630 & 0.1111 & 0.0630 & 0.0833 & 0.1111 & 0.0278 & 0.1592 \\ -0.3110 & 0.5387 & -0.6220 & 0.5387 & -0.3110 & -0.0000 & 0.3110 & -0.5387 & 0.6220 & -0.5387 & 0.3110 \end{bmatrix} \cdot \begin{bmatrix} i_{-5} \\ i_{-4} \\ i_{-3} \\ i_{-2} \\ i_{-1} \\ i_0 \\ i_{-1} \\ i_{+2} \\ i_{+3} \\ i_{+4} \\ i_{+5} \end{bmatrix}$$

Se analiza ahora la estimación del fasor para una señal de entrada tipo senoidal con una frecuencia de 60Hz y se le suma una señal con un cuarto armónico, es generada a través de una función y su ecuación es la 2.19

$$V(t) = 10 \sin(\omega \Delta t) + 5_s \sin(4 * \omega \Delta t) \quad (2.19)$$

La señal de entrada se muestra en la figura 2.12

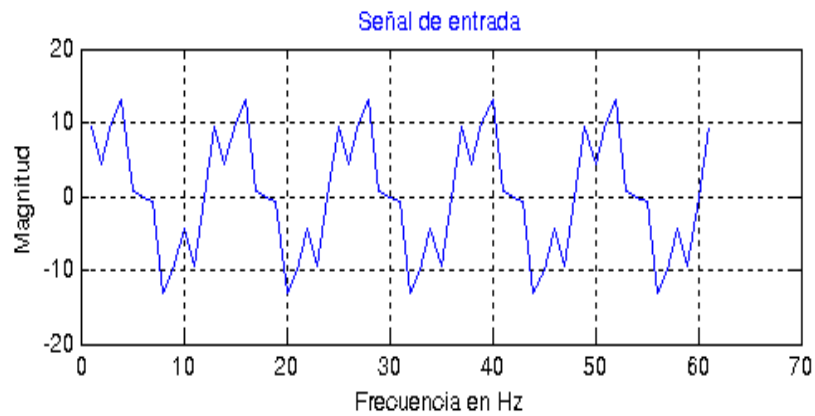


Figura 2.12.- Señal de entrada.

El Fazor estimado en magnitud para el algoritmo de Mínimos cuadrados para la señal de entrada senoidal más cuarto armónico se muestra en la figura 2.13 a) y en la 2.13 b) se muestra el fazor estimado en ángulo.

Como se puede observar en las figuras 2.13, el fazor en magnitud y ángulo tiene una pequeña distorsión al principio sin embargo se corrige en un mínimo tiempo a pesar que la señal de entrada contiene cuarto armónico, esto es debido que esta componente se incluyo en el modelo del algoritmo.

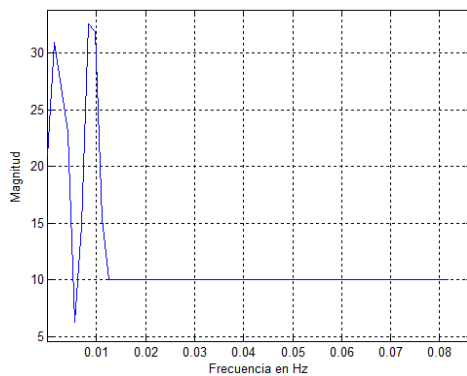


Figura 2.13 a).- Magnitud estimada.

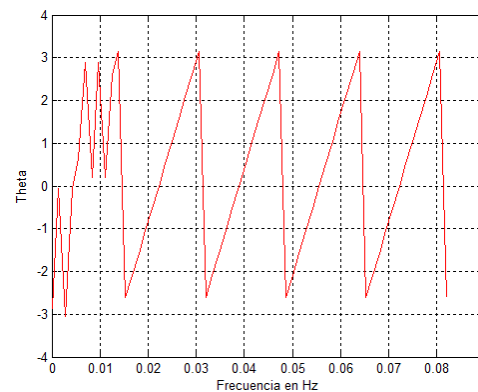


Figura 2.13 b).- Ángulo estimado.

Se analizan los fasores con la señal de prueba de la figura 2.6. Como se puede observar en la figura 2.14 a) la magnitud estimada es errónea, esto es debido a que el

algoritmo no contempla todos los armónicos de la frecuencia fundamental y aunque en la implementación del algoritmo se toman en cuenta los primeros 3 armónicos al instante de introducir uno diferente a estos la estimación del fasor ya no responde adecuadamente.

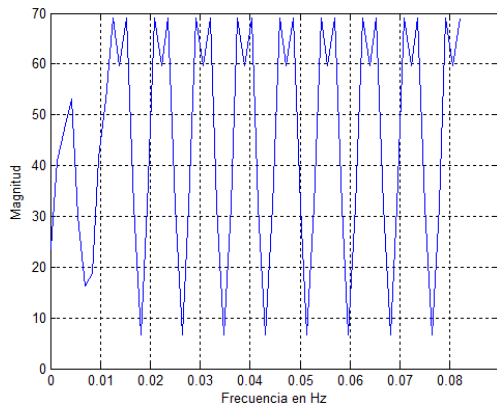


Figura 2.14 a).- Magnitud estimada.

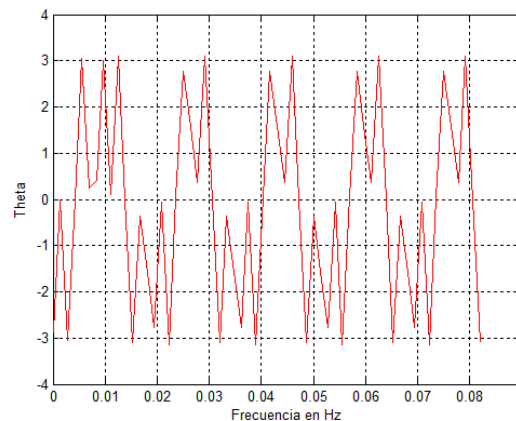


Figura 2.14 b).- Ángulo estimado.

Al igual que los algoritmos trigonométricos, los mínimos cuadrados tienen una respuesta rápida ante transitorios y requieren de muy pocos cálculos, sin embargo la presencia de armónicos en la señal de entrada tiene efectos negativos en el algoritmo, aunque se pueden agregar más elementos a el algoritmo sin embargo aria más robusto el algoritmo y seria necesario saber perfectamente de que armónicos se compone la señal de entrada y como se sabe en el sistema eléctrico de potencia no se conocen estos elementos ya que la señal es muy variante.

2.5 Transformada discreta de Fourier.

La Transformada Discreta de Fourier era conocida en teoría desde hace muchos años, pero solamente con la llegada de la computadora digital fue llevada a la práctica [20]. Ni la Serie de Fourier, ni la Transformada de Fourier se prestan

fácilmente para cálculos en computadoras digitales. Para vencer este impedimento se desarrolló la Transformada Discreta de Fourier [29].

El análisis en frecuencia de una señal implica su descomposición en sus componentes espectrales, tanto en magnitud como ángulo de fase, el espectro es una característica única de una señal y proporciona la información suficiente para poder definirla completamente, ejemplos de esta información son; frecuencia fundamental, magnitud a la frecuencia fundamental, magnitud de sus componentes armónicas, ángulos de fase, además de indicar la presencia de otras señales o simplemente ruido, el análisis de Fourier es una herramienta útil para obtener y analizar el espectro de una señal

La DFT opera con una señal muestreada, a partir de ésta se genera un espectro en el dominio de la frecuencia. El espectro que resulta es una aproximación de la Serie de Fourier, en el sentido que se pierde información entre las muestras de la forma de onda [20].

El análisis DFT es una aproximación del espectro de la señal analógica original. Su magnitud se ve influenciada por el intervalo de muestreo, mientras que su fase depende de los instantes de muestreo. Las formas seno y coseno usadas en la DFT son comúnmente llamadas funciones base de la DFT. Las funciones base son un conjunto de formas de onda seno y coseno de amplitud unitaria [32].

Se dice que la señal de entrada se encuentra en el dominio del tiempo, es decir, muestras tomadas en un periodo de tiempo establecido, o señal discreta de la onda que será transformada. El término dominio de la frecuencia es usado para describir las amplitudes y fases de las frecuencias que componen la señal de entrada [29].

El dominio de la frecuencia contiene exactamente la misma información que el dominio del tiempo, solo que en diferente forma. Si se conoce uno de los dominios se puede conocer el otro.

Si se cuenta con la señal en el dominio del tiempo, el proceso para conocer el dominio de la frecuencia es llamado descomposición o análisis o Transformada Discreta de Fourier. Si se conoce el dominio de la frecuencia, el cálculo para obtener el dominio del tiempo se conoce como síntesis o Inversa de la Transformada Discreta de Fourier.

El número de muestras en el dominio del tiempo se representa por la variable N , el cual puede tomar cualquier valor entero positivo, pero se eligen 128, 256, 512, etc., esto es debido a que los datos guardados digitalmente en una computadora usan direccionamiento binario, esta es la razón por la cual el algoritmo de la DFT conocido, opera con N muestras que son potencia de dos [29].

El dominio de la frecuencia ($X[k]$) consta de dos partes, cada una es un arreglo de $N/2+1$ localidades, dichas partes son: la parte real de $X[k]$ ($\text{Re } X[k]$), que es el arreglo en donde se encuentran las amplitudes de las formas coseno; y la parte imaginaria de $X[k]$ ($\text{Im } X[k]$) que son los valores de las formas seno.

2.5.1 Funciones base de la Transformada Discreta de Fourier.

Las formas seno y coseno usadas en la DFT son comúnmente llamadas funciones base de la DFT. Las funciones base son un conjunto de formas de onda seno y coseno de amplitud unitaria [29].

Las funciones base son generadas de las ecuaciones 2.20 y 2.21:

$$c_k[i] = \cos(2\pi ki / N) \quad (2.20)$$

$$s_k[i] = \sin(2\pi ki / N) \quad (2.21)$$

En donde:

N = número de muestras

k = coeficiente para cada una de las frecuencias, desde cero hasta $N/2$.

i = coeficiente para los puntos de la señal seno y coseno de referencia, varia de cero hasta $N-1$.

c_k = forma de onda cosenoidal para obtener la amplitud de $\text{Re } X []$.

s_k = forma senoidal para la amplitud de $\text{Im } X []$.

En otras palabras, cada punto N de la señal, se puede crear adicionando $N/ (2 + 1)$ señales coseno y seno. Las amplitudes de las formas coseno y seno están contenidas en los arreglos $\text{Re } \bar{X}[k]$ y $\text{Im } \bar{X}[k]$, respectivamente.

El escalamiento se realiza mediante las siguientes ecuaciones:

$$\text{Re } \bar{X}[k] = \frac{\text{Re } X[k]}{N/2} \quad (2.22)$$

$$\text{Im } \bar{X}[k] = \frac{\text{Im } X[k]}{N/2} \quad (2.23)$$

Excepto por:

$$\text{Re } \bar{X}[0] = \frac{\text{Re } X[0]}{N} \quad (2.24)$$

$$\text{Re } \bar{X}[N/2] = \frac{\text{Re } X[N/2]}{N} \quad (2.25)$$

En donde:

$\text{Re } \bar{X}[k]$ y $\text{Im } \bar{X}[k]$ y contienen las amplitudes de las formas seno y coseno necesarias para la transformada inversa de Fourier.

$\text{Re } \bar{X}[k]$ y $\text{Im } \bar{X}[k]$ contienen la parte real e imaginaria del dominio de la frecuencia.

2.5.2 DFT por correlación.

La correlación es una operación matemática en la cual usa dos señales de entrada para obtener una tercera. La correlación es una técnica óptima para detectar una forma de onda conocida en una que contiene ruido [33].

El algoritmo de la DFT en Tiempo discreto, se muestra en las ecuaciones 2.26 a 2.30.

$$x(k) = \sum_{k=-\infty}^{\infty} a_k e^{-jK\omega_0 n} \quad \text{Ec.(2.26)}$$

$$\text{Re } X = \frac{\sum_{i=0}^{N-1} \text{muestra}(i) * \cos(i)}{N/2} \quad \text{Ec.(2.27)}$$

$$\text{Im } X = \frac{\sum_{i=0}^{N-1} \text{muestra}(i) * \text{sen}(i)}{N/2} \quad \text{Ec.(2.28)}$$

$$\text{Magnitud} = \sqrt{(\text{Im } X)^2 + (\text{Real } X)^2} \quad \text{Ec.(2.29)}$$

$$\text{Fase} = \arctg \frac{\text{Im } X}{\text{Re } X} \quad \text{Ec.(2.30)}$$

En donde:

$x(k)$.- Representa la señal analógica en tiempo discreto.

N .- Es el número de muestras a cuantificar del convertidor analógico digital (ADC).

i .- Índice para efectuar el producto de los elementos uno a uno, de cero hasta $N-1$.

$\text{Im } X$ e $\text{Re } X$.- Componentes rectangulares del vector correspondiente a la descomposición de la señal analógica.

Es decir, cada muestra del dominio de la frecuencia es encontrada multiplicando cada uno de los puntos de señal del dominio del tiempo por la forma coseno o seno (parte real o imaginaria) y acumulando los resultados [27]. Lo primero que se debe definir es cual va a ser la frecuencia nominal y cuantas muestras requerimos por ciclo.

- Frecuencia nominal 60Hz.
- Número de muestras por ciclo 12.

De los datos anteriores es posible calcular la frecuencia de muestreo con la ecuación 2.11 anteriormente dada. Sustituyendo los valores de la frecuencia nominal y el número de muestras se tiene una frecuencia de muestreo de 720Hz.

Para el algoritmo de la Transformada Discreta de Fourier se analiza un caso donde se procesa una señal de entrada que este contaminada con segundo, tercero y cuarto armónico; para comprobar que el fasor estimado no es afectado por los armónicos.

El análisis de la estimación de fasor se realiza con la señal de entrada de la figura 2.6. La transformada discreta de Fourier tiene dos filtros que eliminan completamente los armónicos y como respuesta se obtiene un fasor completamente lineal, y que tarda en estabilizarse alrededor de 16.666 mili segundos.

El fasor estimado en magnitud para una señal contaminada de segundo, tercero y cuarto armónico se muestra en la figura 2.15a) y el fasor estimado del ángulo se presenta en la 2.15 b).

En la figura 2.15 a) podemos comprobar que la transformada discreta de Fourier elimina completamente las armónicas, podemos ver que el fasor se estabiliza alrededor de 16.6666 mili segundos que es el periodo de un ciclo de la señal de entrada. El fasor estimado en magnitud indica que la señal de entrada tiene 10 unidades de magnitud en la señal con frecuencia fundamental.

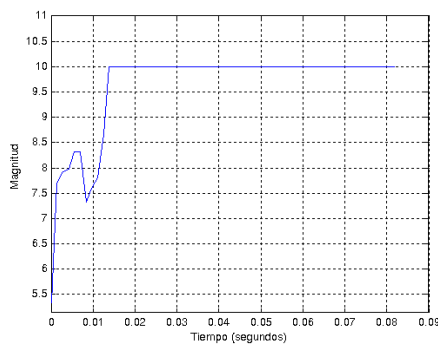


Figura 2.15 a).- Magnitud estimada.

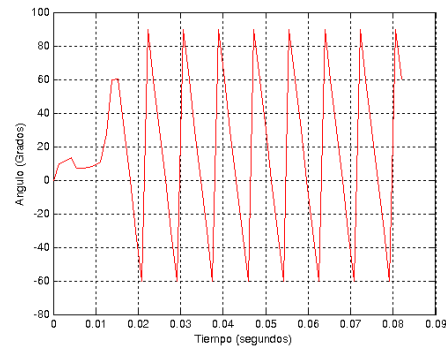


Figura 2.15 b).- Ángulo estimado.

Al igual que en la magnitud, el fasor estimado del ángulo no es afectado por la señal de entrada contaminada de armónicos y tarda en estabilizarse el periodo de un ciclo completo.

Las ventajas de la DFT son que atenúan efectivamente el ruido, atenúan armónicas pares e impares así como las componentes de alta frecuencia y rechazan la componente decreciente; sin embargo también cuenta con desventajas, como respuesta lenta al transitorio y requiere de mayor número de coeficientes.

La DFT es el algoritmo más adecuado para el cálculo de fasores, ya que elimina ruido, armónicas y a pesar de que tarda un ciclo en actualizar las mediciones, este tiempo no afecta la ejecución del programa y cumple con la cláusula 5 del protocolo de comunicación IEEE C37.118

2.6 Algoritmos para la medición de frecuencia

Existen diferentes métodos para calcular la frecuencia de un sistema, en este apartado se hace un análisis de tres diferentes métodos para calcular la frecuencia y elegir el método más preciso.

2.6.1 Algoritmo de la DFT para cálculo de la frecuencia

Con la DFT solo se obtienen los valores reales e imaginarios del fasor resultante, se puede calcular la frecuencia realizando unas iteraciones con diferentes factores Twiddle (señal seno y coseno). El diagrama de flujo para el algoritmo de la DFT se muestra en la figura 2.16.

Para una resolución de una décima (0.1) en la frecuencia es necesario que N (tamaño de DFT) sea 10 veces el valor de la frecuencia de muestreo (f_s), es decir; para obtener una mejor resolución se debe de incrementar el valor de N o disminuir la frecuencia de muestreo, teniendo en cuenta que se requieren 4 muestras por ciclo para reconstruir la señal, es decir; con la restricción del teorema de Nyquits [6].

Para este caso se requiere una resolución de una centésima, si se propone una frecuencia de muestreo de 960, entonces N debe ser igual a 96 000, o sea cien ciclos;

lo que corresponde a 1.66 segundos. Para un instrumento de medición si cumple pero en este caso la norma requiere 10 mediciones por segundo.

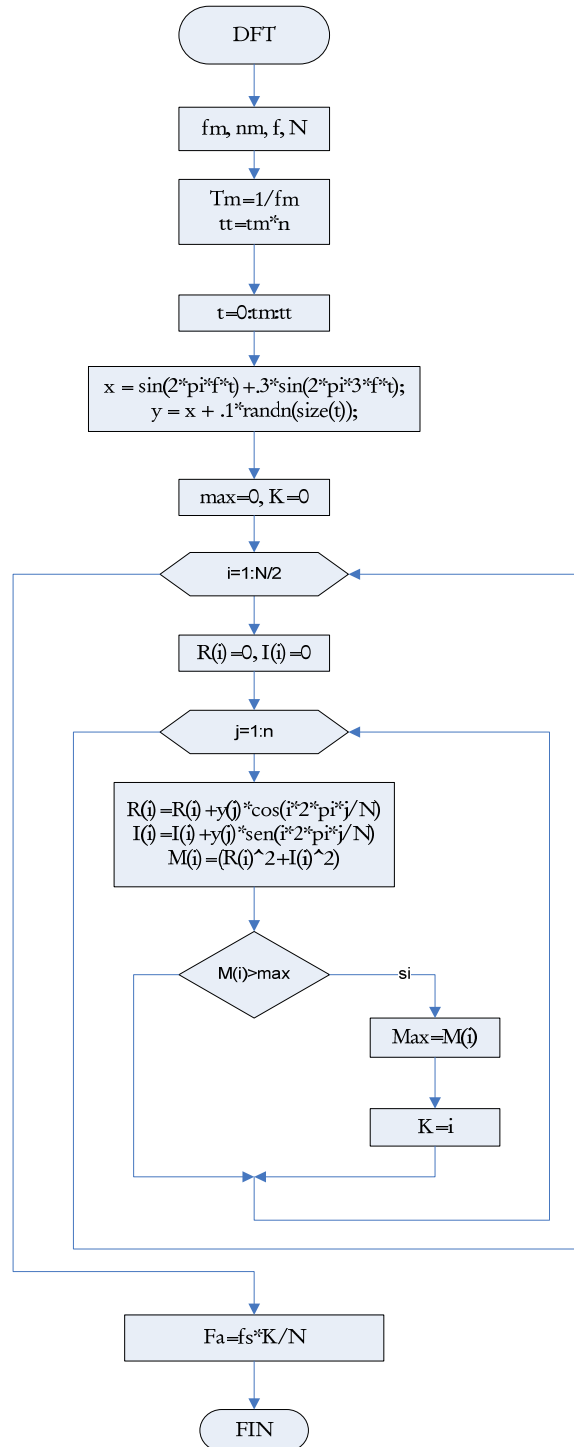


Figura 2.16-Diagrama de flujo de la DFT

La señal de entrada se muestra en la figura 2.17 y esta compuesta por una frecuencia fundamental de 60.4, el tercer armónico de esta frecuencia y ruido.

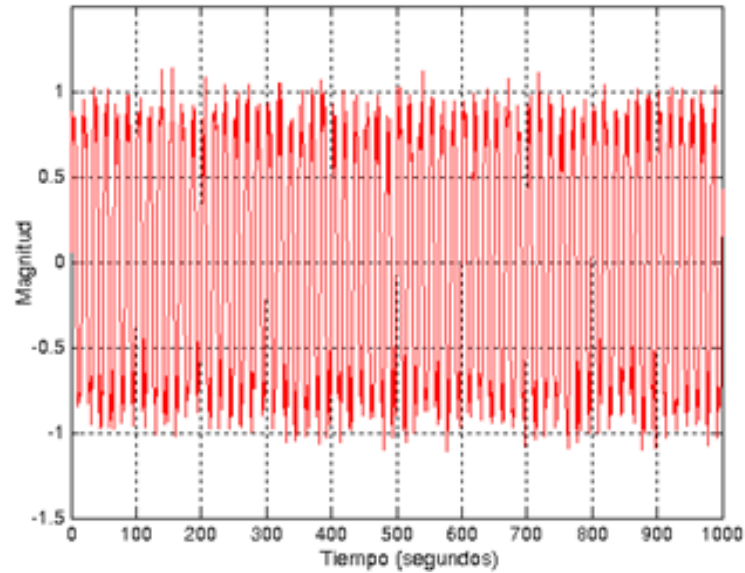


Figura 2.17.-Señal de entrada para la FFT

La respuesta de la DFT en frecuencia es de 60.04, se muestra en la figura 2.18.
Frecuencia (Hz)

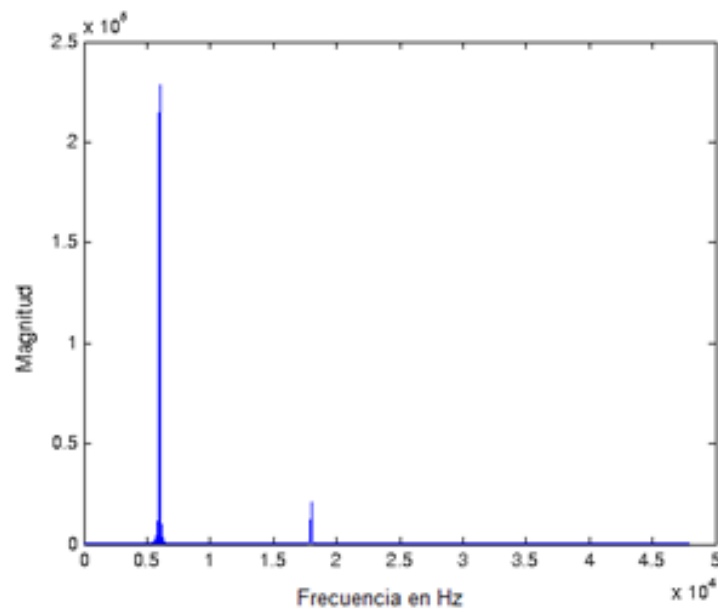


Figura 2.18.-Respuesta de la DFT

2.6.2 Transformada Rápida de Fourier para el cálculo de la frecuencia.

El algoritmo para la Transformada rápida de Fourier (FFT) explota las propiedades de simetría de la exponencial compleja discreta en el tiempo para reducir el número de multiplicaciones. Para evaluar una transformada discreta de Fourier con N muestras el algoritmo de la FFT encuentra su eficiencia cuando N es una potencia de 2. [32].

Figura 2.19 muestra un ejemplo de descomposición del dominio de tiempo utilizados en la FFT. En este ejemplo, una señal de 16 puntos se descompone a través de cuatro etapas separadas. La primera etapa se rompe el punto 16 la señal en dos señales de cada una de ellas de 8 puntos. La segunda etapa los datos se descomponen en cuatro señales de 4 puntos. Este patrón continúa hasta que hay N señales compuestas por un solo punto. Hay $\log_2 N$ etapas necesarias en esta descomposición, es decir, 16 puntos de la señal requiere de 4 etapas, una señal de 127 puntos requiere 7 etapas, una señal de 4096 puntos requiere de 12 etapas, etc. [29]

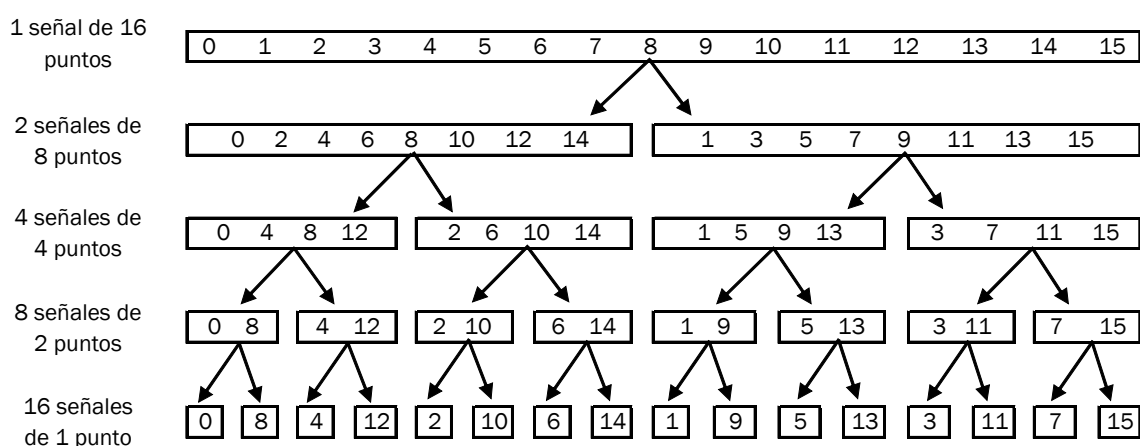


Figura 2.19.- La FFT de descomposición. Un punto N señal se descompone en N señales conteniendo cada uno un solo punto [29].

La descomposición no es más que un reordenamiento de las muestras en la señal de acuerdo con poco orden inverso. El siguiente paso en el algoritmo FFT es encontrar los espectros de frecuencia de 1 punto de las señales en el dominio del tiempo, el espectro de frecuencias de una señal de 1 punto es igual a sí mismo [27].

El último paso en la FFT es combinar los N espectros de frecuencia exacta en el orden inverso que el dominio del tiempo de descomposición se llevó a cabo. En la primera etapa, el 16 de espectros de frecuencias (1 punto cada uno) se sintetizan en 8 espectros de frecuencia (2 puntos cada uno), y así sucesivamente. El último escenario resultado en la salida de la FFT, un punto de 16 espectros de frecuencias [29].

La FFT además de estimar la parte real e imaginaria del fasor resultante, es posible calcular la frecuencia de la señal, esto se hace calculando la magnitud con las partes real e imaginario y realizando un recorrido en el vector resultante para encontrar la posición de la magnitud más grande.

2.6.2.1 Algoritmo de la FFT

En el software de MATLAB ya existe una función que calcula la FFT, esta función tiene como argumento la señal de entrada y el número de DFT que va a realizar. La señal de entrada es la misma que para la DFT, con las mismas condiciones de resolución en la frecuencia. A continuación se presenta el diagrama de flujo en la figura 2.20 para el algoritmo de la FFT.

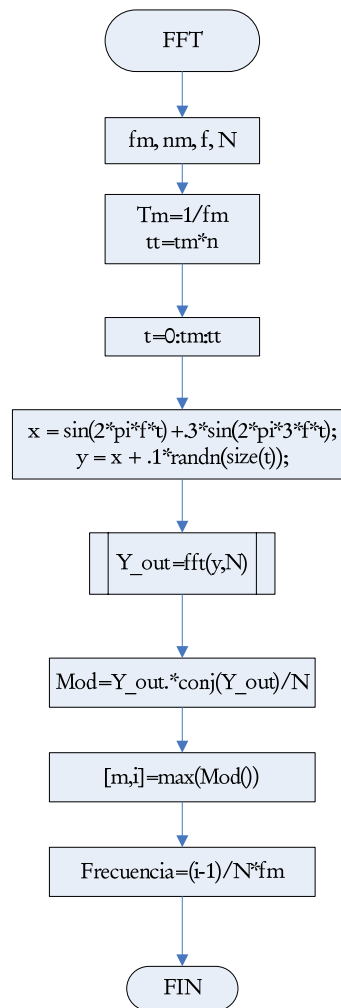


Figura 2.20.-Diagrama de flujo de la FFT

La respuesta de la FFT para calcular la frecuencia es de 60.3999 y se muestra en la figura 2.21.

La respuesta de la DFT y FFT es correcta y precisa, pero existen diferencias muy significativas en el tiempo de ejecución, en MATLAB la DFT tarda aproximadamente 10 minutos en arrojar resultados, mientras que la FFT tarda menos de 10 segundos. Esto se debe a que la DFT realiza 2^N ($2^{96000} = 2.5785e28898$) cálculos y la FFT solo tarda $N \log_2 N$, para este caso N es muy grande del valor de 96000.

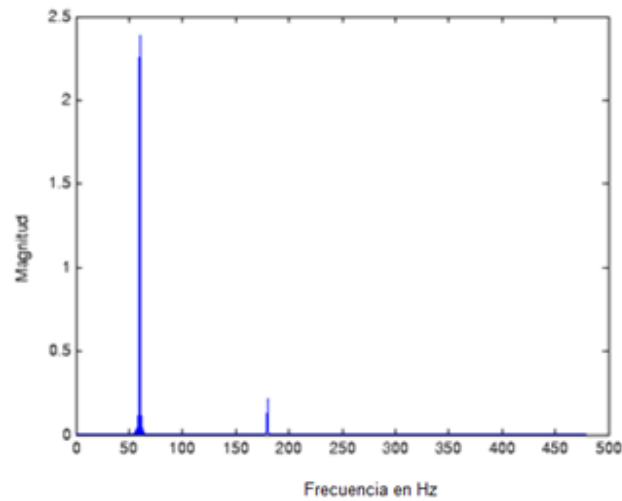


Figura 2.21.-Respuesta de la FFT

Para la implementación de un medidor fasorial estandarizado no es conveniente utilizar la FFT ya que requiere de 100 ciclos para poder re calcular los fasores y la frecuencia del sistema. La DFT es ideal para la implementación de un PMU estandarizado, aunque solo calcula el fasor del sistema y la frecuencia no, este solo requiere de un ciclo para actualizar los datos y se propone calcular la frecuencia con otro algoritmo.

2.6.3 Cálculo de la frecuencia por cruce de cero

El algoritmo del cálculo de la frecuencia por el método de cruce por cero consiste en detectar un cruce por cero con pendiente negativa y a partir de ese instante contar el número de muestras enteras que se obtengan hasta el próximo cruce por cero.

Como el número de muestras que caben en un ciclo no es un número entero, hay que determinar las fracciones de muestra que se obtienen antes y después del cruce por cero, para conocer el número fraccionario de muestras que caben en un ciclo. Una vez conocido el número exacto de muestras que caben en un ciclo, multiplicado por el periodo de muestreo se determina el periodo de la señal y su inverso que es la frecuencia de la señal.

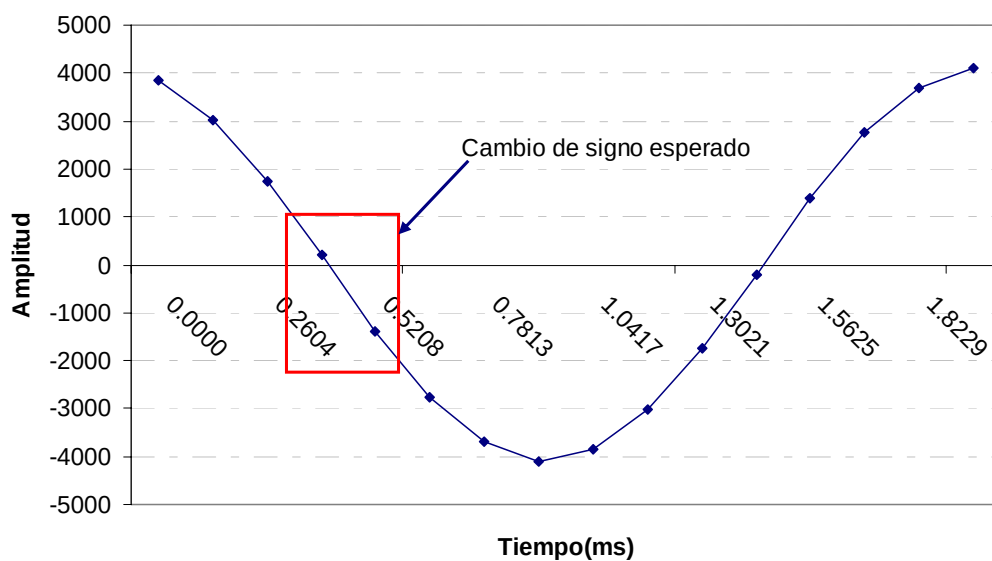


Figura 2.22.- Medición de frecuencia.

Para el cálculo de la fracción de muestra en el instante anterior el cruce y el posterior, se hace uso de la ecuación de la recta, en la figura 2.23 se muestra la grafica que forman la muestra actual y la anterior en el instante que ocurre el cruce por cero.

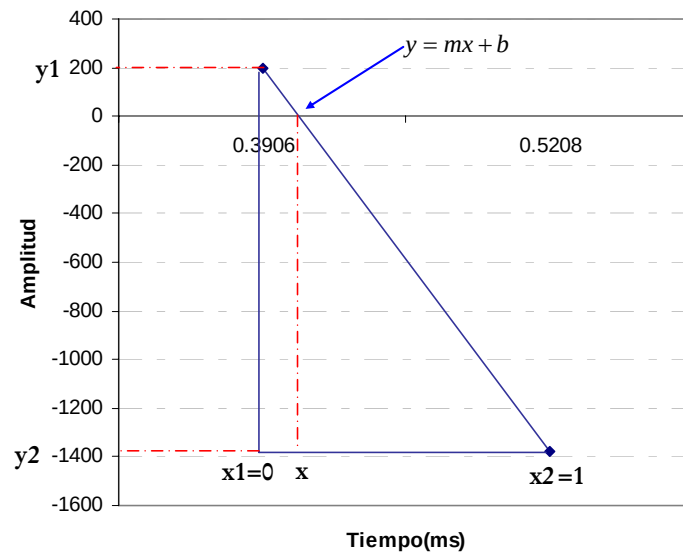


Figura 2.23.-Forma de calcular la fracción de tiempo por medio de una ecuación lineal.

Se desea calcular el intervalo de x_1 a x ; para calcular la pendiente (m) se utiliza la ecuación de la recta,

$$y = mx + b \quad (2.31)$$

$$m = \frac{y_2 - y_1}{x_2 - x_1} \quad (2.32)$$

Donde

y_1 .- muestra anterior

y_2 .- muestra actual

x_1 .-tiempo de muestreo anterior

x_2 .-tiempo de muestreo actual

El intervalo entre x_1 y x_2 corresponde a un periodo de muestreo ($x_2 - x_1 = 1$) y se maneja por unidad (PU).

Si se quiere encontrar el valor de x cuando $y=0$, $b=y_1$ y $x_2-x_1=1$, despejando x ; se tiene la ecuación 2.33:

$$x = \frac{-y1}{y2 - y1} \quad (2.33)$$

En la figura 3.9 se puede observar que $y1$ es positiva y $y2$ es negativa, por lo tanto la ecuación final queda de la siguiente manera.

$$\Delta fa = x = \frac{y1}{y2 + y1} \quad (2.34)$$

Donde:

Δfa .- fracción de muestra anterior al cambio de signo.

El valor de x corresponde a la fracción de muestra anterior al cambio de signo, para calcular la fracción de muestra posterior al cambio de signo es necesario realizar una sencilla operación, donde la suma de la fracción de muestra anterior y posterior es igual a la unidad. Recordando que el periodo de muestreo se toma como la unidad.

$$\Delta fp = 1 - \Delta fa \quad (2.35)$$

Donde:

Δfp .- fracción de muestra posterior al cambio de signo.

Se realiza el programa en MATLAB para mostrar sus resultados y verificar su eficiencia para poder comparar con los métodos anteriores.

El primer resultado que arroja este algoritmo es erróneo debido a que no se conoce en que instante del ciclo empieza a ejecutarse, después de la primera ejecución el resultado es satisfactorio y la frecuencia resultante es la de la señal fundamental, este algoritmo no identifica que armónicos tiene la señal de entrada pero esto no es necesario ya que la DFT lo hace.

Este algoritmo es ideal para la aplicación del PMU, ya que es muy rápido con respecto a los 2 algoritmos vistos anterior

CAPÍTULO 3.- Diseño De Software

En este capítulo se presenta el desarrollo del software de la unidad de medición fasorial sincronizada vía satelital y estandarizada mediante el estándar IEEE C37.118 mediante un Microcontrolador ColdFire V2 de 32 bits.

Se muestran en forma detallada, las etapas que son necesarias para implementar el PMU, como son; los algoritmos de medición de los fasores de voltaje y corriente, frecuencia fundamental, la etapa de sincronización con el GPS y el protocolo de comunicación.

3.1 Ambiente de desarrollo integral (IDE) del microcontrolador

La programación del microcontrolador M52233 utiliza un ambiente de desarrollo integral, llamado “Code Warrior” de Metrowerks. En el que se puede programar tanto en lenguaje ensamblador como en C++. La estructura de programación utilizada es por medio de un programa principal y rutinas [36].

El programa principal realiza llamadas a rutinas y ejecución de instrucciones, las cuales se considera que no son convenientes realizar en las interrupciones de los periféricos, debido al tiempo de procesamiento que lleva efectuarlos.

Las rutinas se clasifican en:

- Interrupciones o eventos
- Funciones o métodos.

Las interrupciones son rutinas ejecutadas por eventos de dispositivos periféricos o internos del microcontrolador, por ejemplo Interrupciones del reloj:

- Por tiempo.
- Por captura o conteo de tiempo.

Interrupción de comunicación serial.

- Por transmisión.
- Por recepción.

Al realizarse la interrupción de algún dispositivo se ejecutan instrucciones convenientes para el mismo o el control de otros dispositivos.

Las funciones son rutinas con una tarea específica, las cuales se pueden llamar desde una interrupción o del programa principal. Estas funciones pueden ser:

- Función Inicialización de dispositivos.
- Función Calcular DFT.
- Función Calcula frecuencia.
- Función para protocolo de comunicación.

3.1.1 Software del microcontrolador de 32 bits V2 ColdFire.

El programa CodeWarrior para microcontroladores V.7.1 “Special Edition” incorpora la herramienta de configuración de “Processor Expert” y permite trabajar gratuitamente con la herramienta hasta 32k de código generado en C [37].

La metodología de programación utilizada es por medio de un programa principal llamado “main” y uno denominado “events”, estando en este último, las diferentes interrupciones y/o eventos que son configuradas por el usuario. El programa principal siempre se está ejecutando dentro de un lazo infinito, el cual es interrumpido cada vez que un evento y/o interrupción es invocado.

Es de importancia mencionar que al implementar un proyecto en este software es necesario configurar el “bean” del CPU, aquí se define que trabaja con cristal externo de 25 MHz, para no tener problemas al implementar las interrupciones de tiempo.

3.1.2 Crear nuevo proyecto para V2 ColdFire.

Una parte importante de la implementación es el conocimiento general del software de CodeWarrior en donde se implementan los algoritmos, A continuación se muestran los pasos que se deben de seguir para implementar un nuevo proyecto.

- La ruta que se debe de seguir para abrir el programa es la siguiente: inicio/todos los programas/Freescale CodeWarrior/CodeWarrior for ColdFire V7.1/CodeWarrior IDE; se abre la ventana de la figura 3.1, en donde se crea un proyecto nuevo.



Figura 3.1.-Programa Freescale CodeWarrior.

- Se abre la ventana de la figura 3.2 en donde se elige la tarjeta con la cual se va a trabajar y la conexión con la cual se programa y se selecciona siguiente.

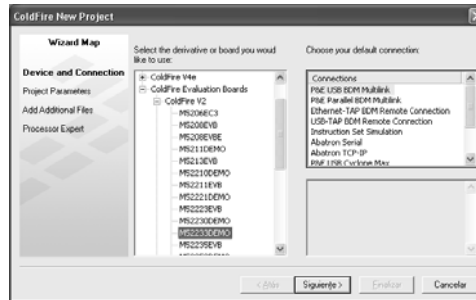


Figura 3.2.-Selección de la tarjeta en uso y medio de programación.

- En la siguiente ventana se selecciona el lenguaje de programación, puede ser C, C++ o ambos, también se asigna nombre al proyecto y se da la ubicación.
- Si ya se cuenta con algún archivo este se añade en la siguiente ventana.
- La siguiente ventana es muy importante, ya que es aquí donde se especifican las opciones de diseño de la aplicación; para la implementación del PMU se trabaja con processor expert, se selecciona como se muestra en la figura 3.3.

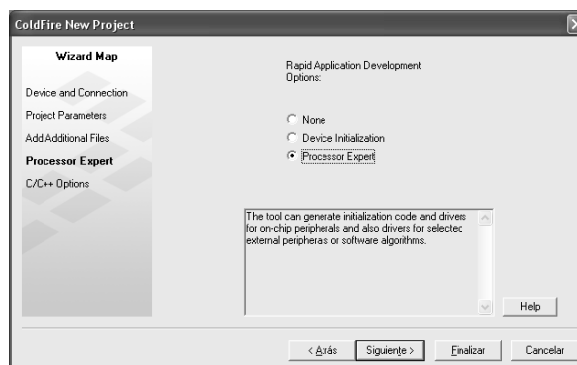


Figura 3.3.-Selección de processor expert.

- En la última ventana se indica si el código se optimiza y se da clic en finalizar.

- Al instante de finalizar aparece otra ventana donde se pregunta con que tipo de memoria se desea trabajar si Flash o RAM, se puede trabajar con las dos (figura 3.4).

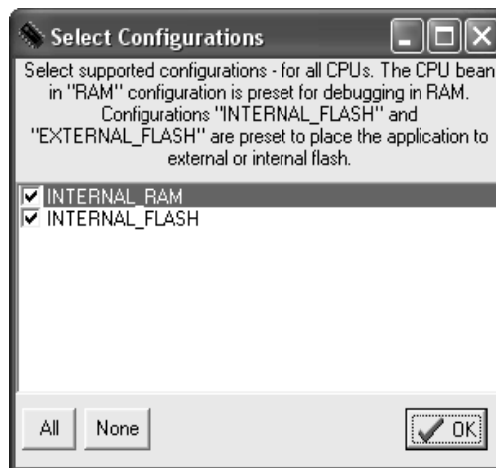


Figura 3.4.-Selección de memoria.

- Finalmente en la siguiente ventana se indica el número de pines del microprocesador a utilizar, para poder inicializar el CPU correctamente y se da un clic en el icono “make” para empezar a programar.

La tarjeta de desarrollo M52233DEMO tiene un circuito de multienlace el cual permite programar y depurar el programa a través del puerto USB desde la PC, además de energizar a la tarjeta mediante ese mismo puerto [37].

3.2 Diagrama esquemático del PMU

En la figura 3.5 se muestra un diagrama a bloques general de un sistema PMU, el cual está compuesto de los elementos físicos básicos para lograr la estandarización.

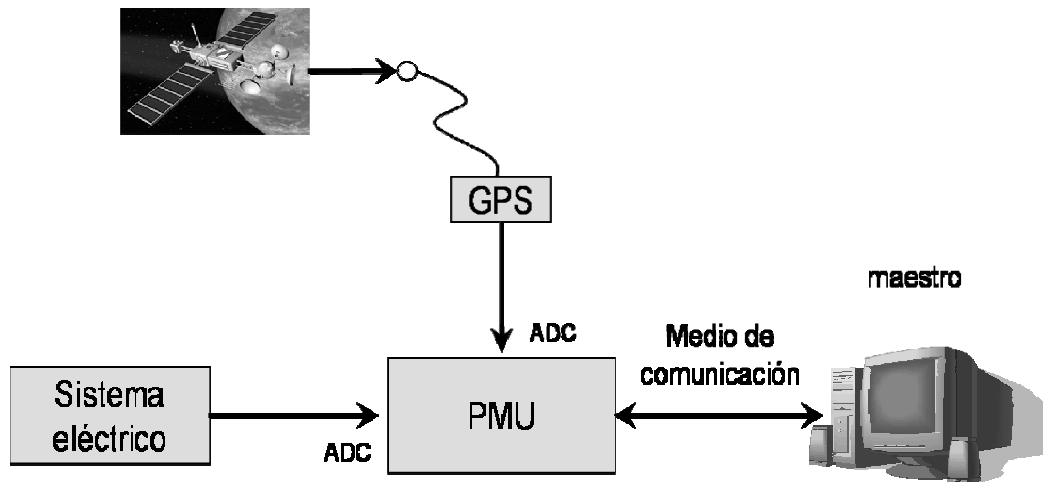


Figura 3.5 Diagrama general de un PMU.

El GPS genera un pulso por segundo que se utiliza para sincronizar al PMU. El pulso por segundo da inicio a la adquisición de datos. Además transmite la trama NMEA, uno de sus datos es el tiempo universal coordinado (UTC), este acompaña a cada fasor indicando el instante exacto de la medición [24].

El PMU consta de un receptor de señales de GPS, un sistema de adquisición de datos y un microprocesador.

La tarjeta de desarrollo MC52233 DEMO, cuenta con un conector J1 en el cual se tiene acceso a los puertos del microcontrolador y su configuración se muestra en la figura 3.8, es importante conocer la configuración del microcontrolador ya que se utilizan los puertos del ADC y un pin de entrada/salida de propósito general para capturar el pulso por segundo proveniente del GPS, así como los puertos de transmisión receptor asíncrono universal (UART) para lograr la comunicación y sus conexiones se muestran en la figura 3.6.

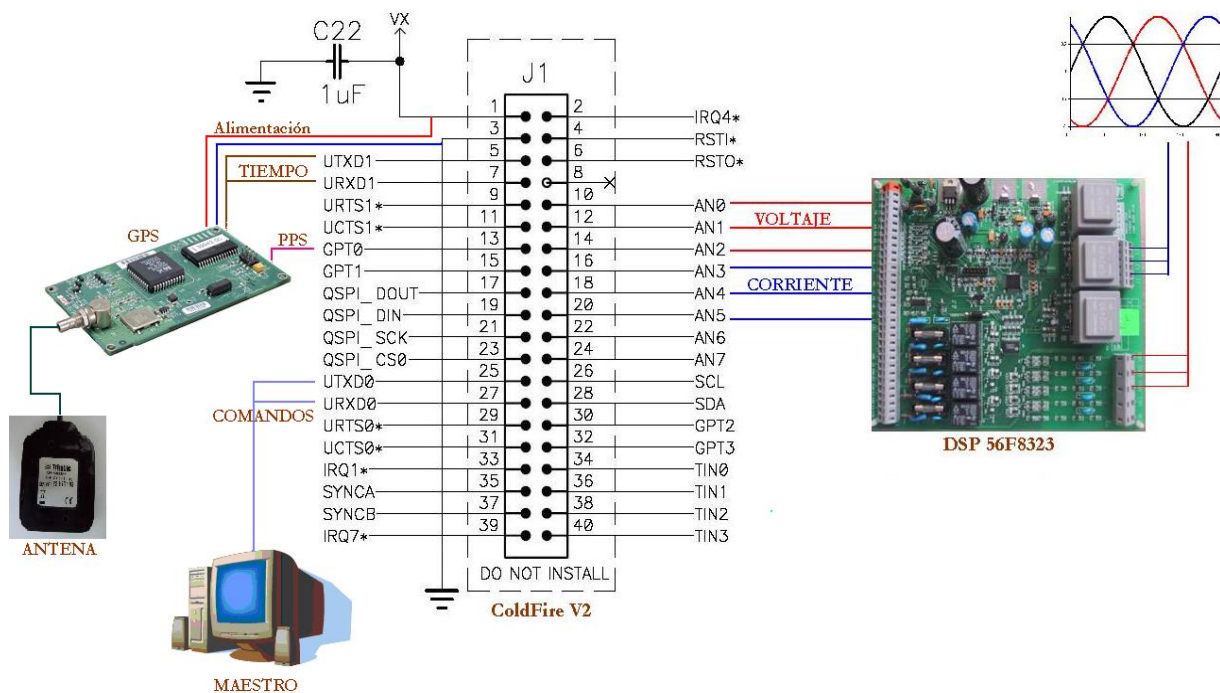


Figura 3.6.- Conector J1 de la tarjeta de desarrollo.

Las señales de voltaje y corriente provenientes del sistema eléctrico, son capturadas por el PMU y mediante el ADC del microcontrolador convertidas a su forma discreta. Mediante la DFT por correlación se calculan los fasores de voltaje y corriente además de calcular la frecuencia de la señal con un algoritmo independiente de la DFT, todo se realiza en tiempo real.

El maestro se encarga de enviar diferentes comandos hacia el PMU y recibir los datos requeridos, esta comunicación se logra a través del RS232 o vía Ethernet.

El diagrama del hardware software del PMU se muestra en la figura 3.7, este describe de forma grafica la operación total del proyecto, así como la interacción del hardware con las rutinas del software [37].

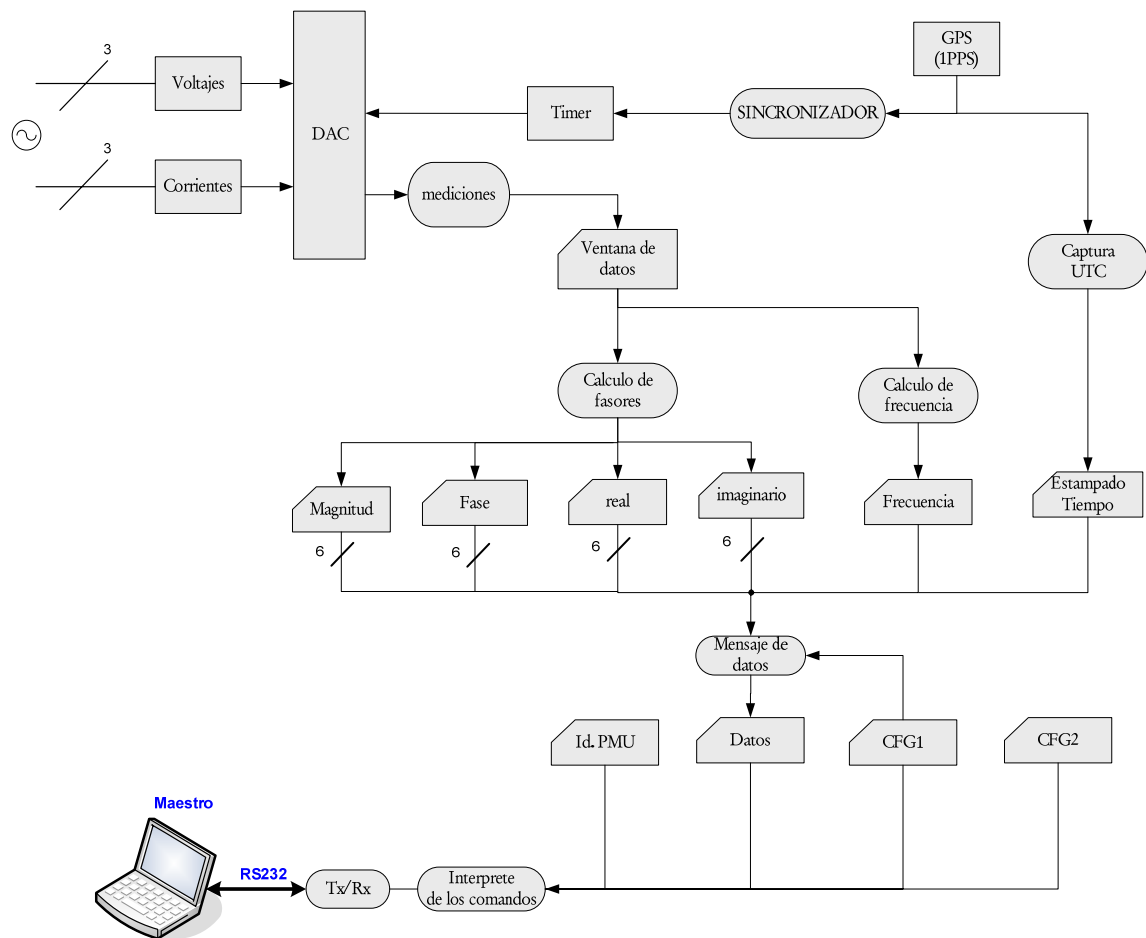


Figura 3.7 Diagrama conceptual del hardware/software a implementar.

Esta es una forma muy útil de abordar el problema, de manera que se tenga un amplio panorama de lo que se implementa. En rectángulos se muestra el hardware, mientras que con óvalos el software a implementar es indicado, y además se muestran en rectángulos cortados los paquetes de datos utilizados por el PMU [20].

En la figura 3.7 se muestra la interacción entre el maestro y el PMU, es por esa razón que la flecha se encuentra en ambos sentidos; y la recepción de datos en el PMU del GPS, en este caso la flecha va el GPS hacia el PMU. En el diagrama de Hardware/Software se identifican tres subsistemas principales: comunicaciones, mediciones y GPS, su diagrama se muestra en la figura 3.8.

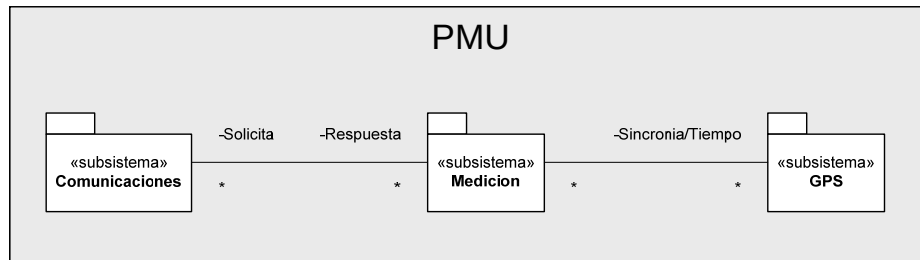


Figura 3.8 – Diagrama de subsistemas del PMU

3.3 Sistema de Posicionamiento Global.

El subsistema del GPS tiene como objetivo principal la sincronización del PMU mediante el pulso por segundo que recibe de un satélite, además de capturar el tiempo exacto de la medición de los fasores. El diagrama de secuencias del subsistema GPS se muestra en la figura 3.9.

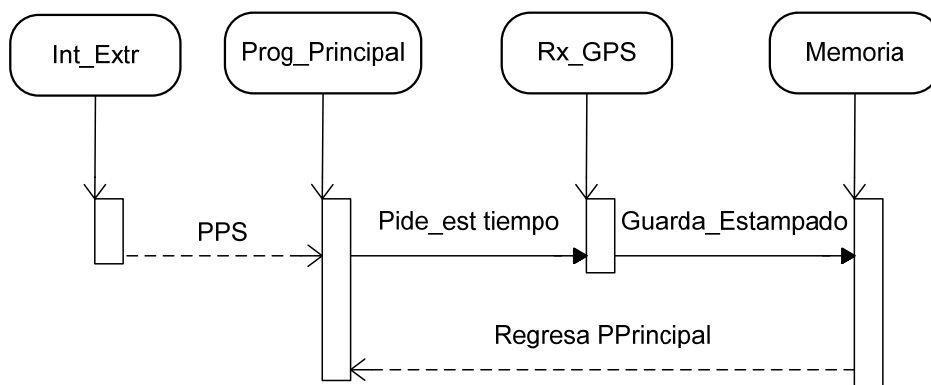


Figura 3.9.- Diagrama de secuencias del subsistema GPS.

Se ocupa un pin de entrada/salida de propósito general de la tarjeta de desarrollo MC52233 DEMO para capturar el pulso por segundo proveniente del GPS. En el software de ColdFire se agrega un “bean” llamada ExtInt, una interrupción externa que genera un evento; su diagrama de flujo se muestra en la figura 3.10.

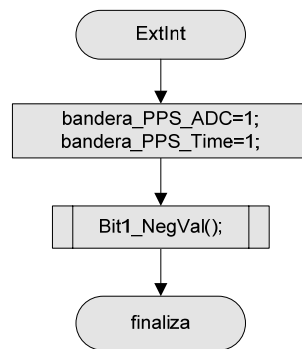


Figura 3.10.- Diagrama de flujo la interrupción externa.

En el evento de la interrupción externa se activan dos banderas, bandera_PPS_ADC para indicarle al ADC en que momento empezar a capturar las señales, es decir; para la sincronización del PMU y bandera_PPS_Time para capturar el estampado de tiempo que acompaña a cada una de las mediciones.

El estampado de tiempo es capturado por un puerto serial en el momento que la bandera para este fin es activada. El diagrama de flujo se visualiza en la figura 3.11. La configuración del puerto es: 4800 baudios, 8 bits, sin bit de paridad y un bit de paro.

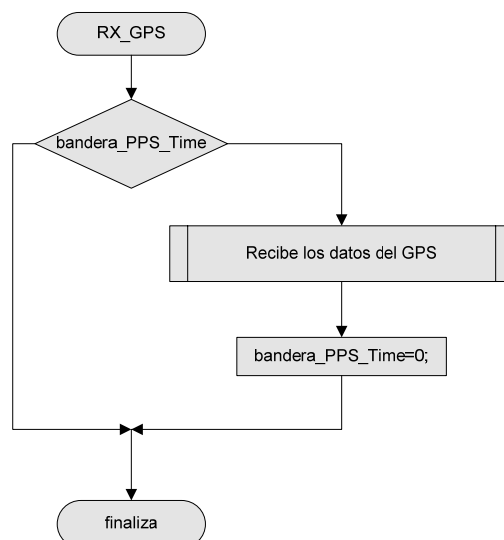


Figura 3.11.- Diagrama de flujo de la Recepción del estampado.

En la interrupción, se obtienen los primeros 16 caracteres, de los cuales se obtiene el UTC en el formato hh.mm.ss.d.

La bandera PPS_time es la que indica si se puede guardar el mensaje en el arreglo Buffer_Unix_Time, el cual tiene una longitud de 16 caracteres. La bandera “PPS_time” se deshabilita cuando ya se acabaron de recibir los 16 caracteres y la vuelve a habilitar el PPS.

Los campos del mensaje en protocolo NMEA 0183 [39] se hará basándose en la siguiente lectura y su significado de los principales campos del formato se observan en la tabla 3.1.

\$GPGGA,222226.0,1930.295,N,09909.331,W,1,04,2.39,02254,M,-009,M,,*6E

Tabla 3.1. Significado de los principales campos del renglón 1 de la trama NMEA.

CAMPO	SIGNIFICADO
\$	Indica el inicio de mensaje.
GPGGA	Indica que es información fija del GPS.
222226.0	Indica el UTC, en formato hhmmss.d

Los datos posteriores al UTC proporcionan información de la velocidad de la Tierra y posicionamiento [37], estos no se ocupan.

3.4 Medición.

El subsistema de medición incluye diferentes bloques de programación, los cuales pueden ser descritos por separado; en la figura 3.12 se muestra la interacción en tiempo real de dichos bloques, este es un diagrama de secuencias.

Como se puede observar en la figura 3.12 la primer rutina que se ejecuta es el programa principal, inmediatamente inicializa los periféricos internos de la tarjeta M52233DEMO, después espera la interrupción del pulso por segundo, enseguida se captura el estampado de tiempo y se inicia el muestreo de las señales de voltaje y corriente; cada 8 muestras corre el algoritmo de medición de frecuencia.

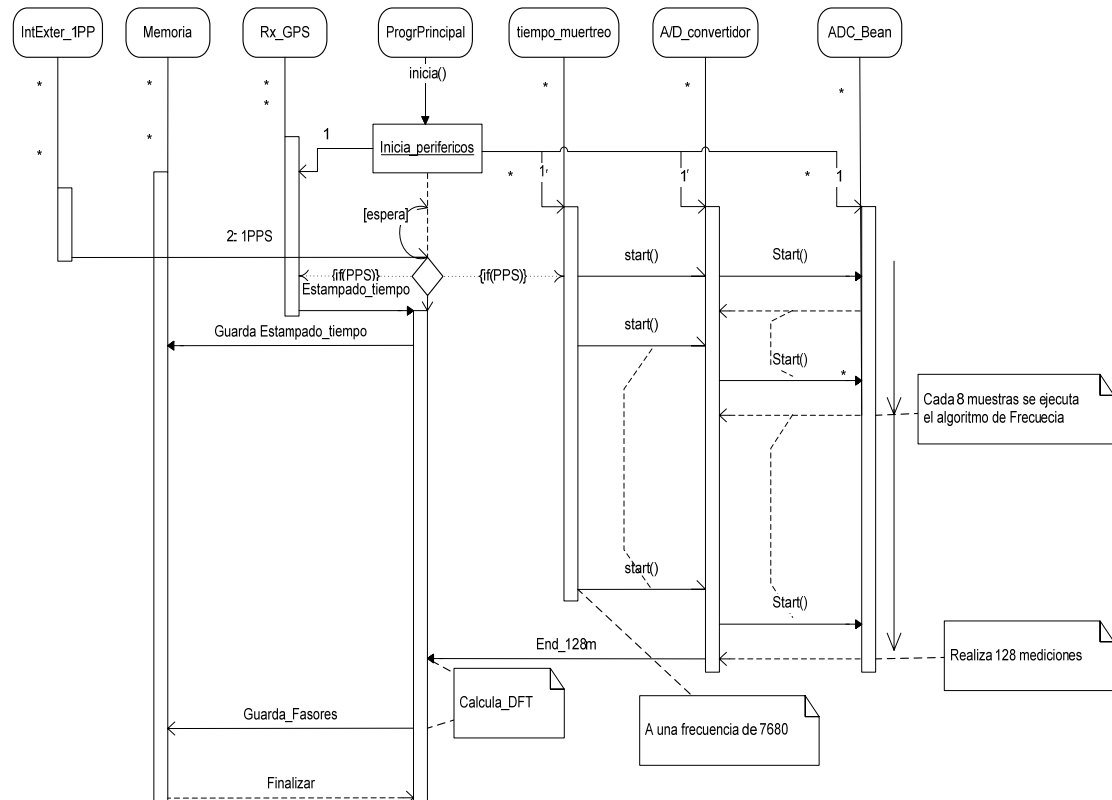


Figura 3.12.- Diagrama de secuencias de mediciones.

En la captura de las señales del sistema eléctrico, se tiene que definir la frecuencia de la señal fundamental así como el número de muestras por ciclo que se requieren para el cálculo de los fasores; con estos datos se calcula la frecuencia de muestreo con la ecuación 2.7.

- Frecuencia nominal 60Hz.
- Número de muestras por ciclo 128.

Frecuencia de muestreo 7680 Hz ($f_m = 60 \cdot 7680 \text{ Hz}$).

Se captura una señal analógica de tipo sinusoidal en una ventana de 128 muestras, a una frecuencia fundamental de 60Hz, estos datos se procesan y se calcula el fasor resultante con el algoritmo de la DFT.

Algoritmo: Calculo de la DFT a 60hz de una señal digital.

Precondición: Ventana de 128 Muestras tomadas a una señal eléctrica, TOTALMUESTRAS=128, Tablas del seno y coseno de tamaño $2\pi/\text{TOTALMUESTRAS}$.

Postcondición: mag60 = Magnitud a 60hz de la señal muestreada, ang60 = Ángulo a 60hz de la señal muestreada.

Algoritmo detallado:

1. Inicia SumaSenos 0, SumaCosenos 0, ContMuestras 0
2. Mientras ContMuestras < TOTALMUESTRAS; repite
 - 2.1. SumaSenos = SumaSenos +
Muestras[ContMuestras]*TablaSenos[ContMuestras]
 - 2.2. SumaCosenos = SumaCosenos +
Muestras[ContMuestras]*TablaCosenos[ContMuestras]
 - 2.3. ContMuestras = ContMuestras + 1
3. real60 = SumaCosenos/(TOTALMUESTRAS/2)
4. imag60 = -SumaSenos/(TOTALMUESTRAS/2)
5. mag60 = Raiz_Cuadrada(real60² + imag60²)
6. ang60 = ArcTan(imag60/real60)

3.4.1 Programa principal

El diagrama de flujo del programa principal se muestra en la figura 3.10, Aquí lo primero que se realiza es la inicialización de todos los periféricos, con la función PE_low_level_init, Inicializa “beans” y proporciona registro común de inicialización, el método se llama automáticamente como parte de la aplicación del código de inicialización; este método es interno y es utilizado únicamente por processor expert.

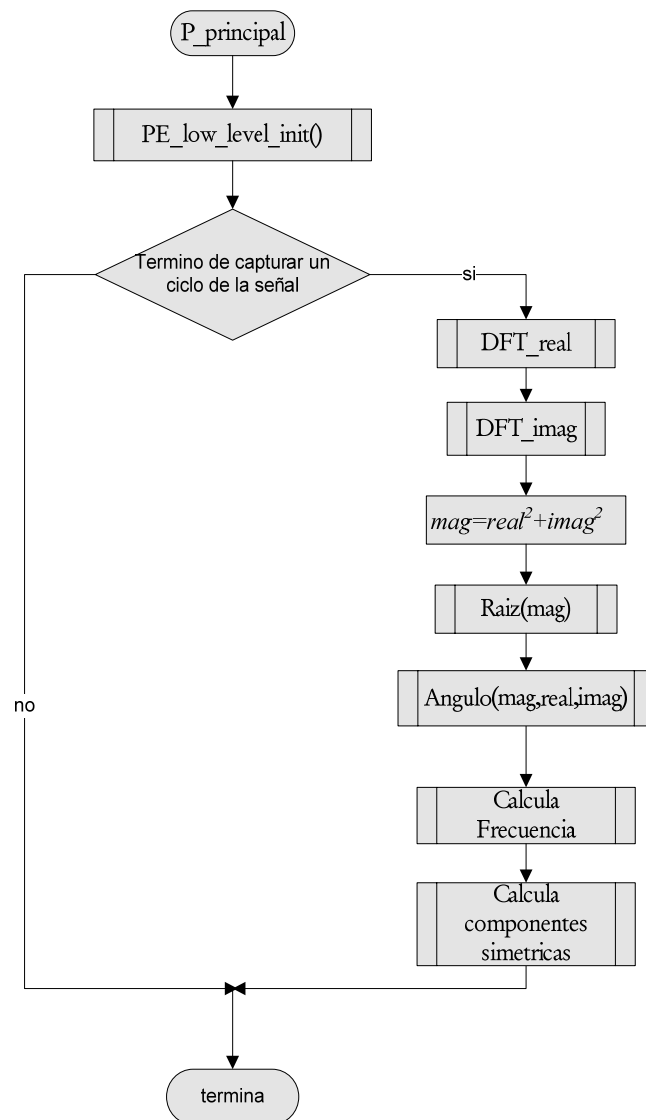


Figura 3.13.- Diagrama de flujo del programa principal.

Después de la inicialización, se pregunta si es un ciclo completo, de ser así se procede a calcular la parte real e imaginaria o magnitud y ángulo de la señal de entrada capturada por el ADC con el algoritmo de la DFT. Se calcula la parte real e imaginaria y la magnitud y ángulo del fasor ya que la norma IEEE C37.118 tiene la opción de enviar la forma polar o cartesiana del fasor. En la figura 3.13 se puede observar que el programa principal cuenta con diferentes subrutinas.

Las subrutinas para calcular la parte imaginaria y real se muestran en las figuras 3.14.

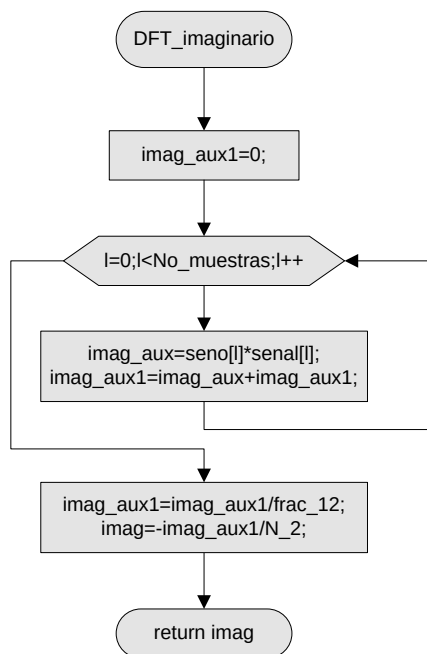


Figura 3.14 a).- Diagrama del cálculo del imaginario.

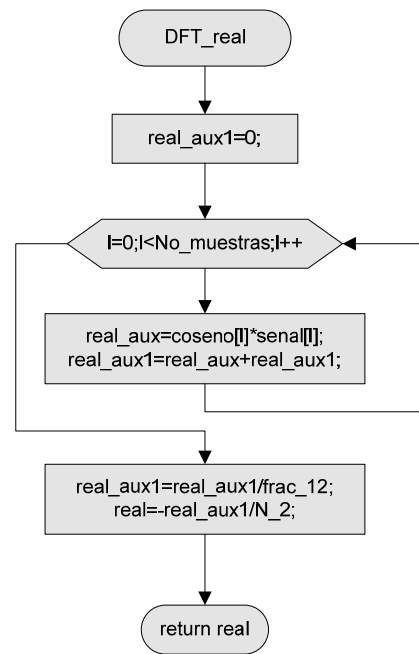


Figura 3.14 b).- Diagrama de flujo para calcular el real.

Después que se capturaron las 128 muestras en un arreglo, estas se procesan para calcular la parte real e imaginaria con el método de la DFT, explicado en el capítulo 2.

Para calcular la magnitud se utiliza la ecuación 2.22, se realiza una subrutina para calcular la raíz y se muestra en la figura 3.15, esto se debe a que se está trabajando con un microcontrolador que no cuenta con funciones trigonométricas.

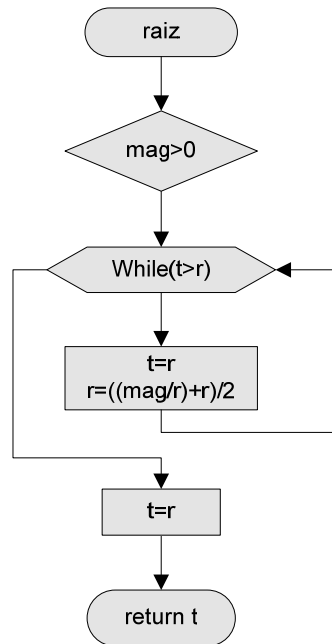


Figura 3.15.- Diagrama de flujo para calcular la raíz.

Otro punto es calcular el ángulo, usualmente se utiliza el arco tangente del valor imaginario entre el real, sin embargo cuando el ángulo es 90° y 270° el real es cero la división entre imaginario y real es un número infinito. Para evitarlo se decide trabajar con el arco coseno, aquí no se presenta discontinuidad ya que el divisor es la magnitud, dado que si existe medición de señal, la magnitud es diferente de cero. En la figura 3.16 se muestra un fasor y el cálculo del ángulo a través de la función arco coseno.

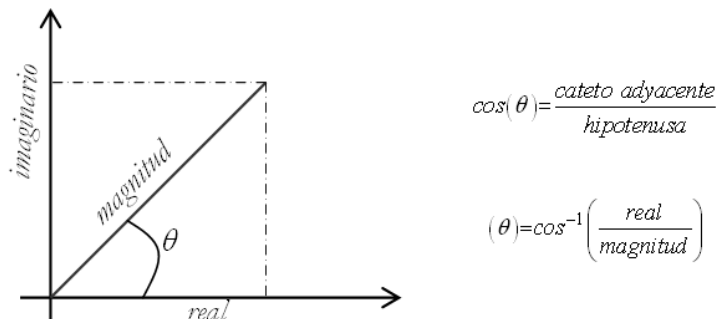


Figura 3.16.- Calculo del ángulo de un fasor.

Como se puede observar en la figura 3.16 se trabaja solo en el primer cuadrante, donde el imaginario y real son positivos; para calcular el ángulo se ocupan el valor absoluto de la parte real entre la magnitud; a partir de este resultado se busca el ángulo que le corresponde en una tabla. Si el valor del real/magnitud se encuentra entre dos ángulos se emplea la interpolación.

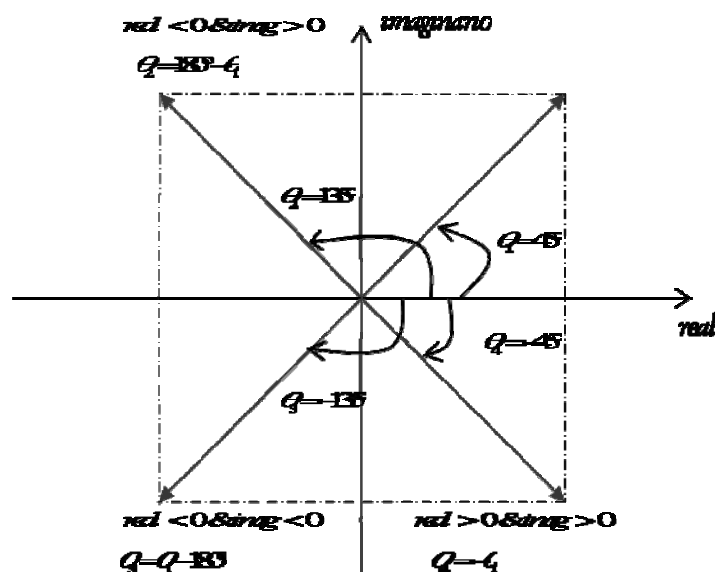


Figura 3.17.- Cálculo del ángulo en los 4 cuadrantes.

La tabla del ángulo va de 0° a 90° , abarca el primer cuadrante. A partir de estos resultados es posible calcular el ángulo cuando el real e imaginario no son positivos, es decir el fasor se encuentra en los cuadrantes 2, 3 o 4 como se muestra en la figura 3.17 y en la tabla 3.2 se muestra el cálculo del ángulo cuando el fasor se encuentra en diferente cuadrante.

Tabla 3.2 Signo del real e imaginario para el cálculo del ángulo

Real	Imaginario	Cuadrante	Angulo
Positivo	Positivo	1	$\text{ang} = \arccos(\text{real} / \text{magnitud})$
Positivo	Negativo	2	$\text{ang} = 180 - \arccos(\text{real} / \text{magnitud})$
Negativo	Positivo	3	$\text{ang} = \arccos(\text{real} / \text{magnitud}) - 180$
Negativo	Negativo	4	$\text{ang} = -\arccos(\text{real} / \text{magnitud})$

Como se puede observar en la tabla 3.2 el ángulo depende del cuadrante donde se encuentre el fasor y esta dado por el signo de las componentes real e imaginario. Para obtener el ángulo correcto es necesario sumar o restar 180° a ang1 , dependiendo del cuadrante donde se encuentre el fasor. El diagrama de flujo para calcular el ángulo se muestra en la figura 3.18.

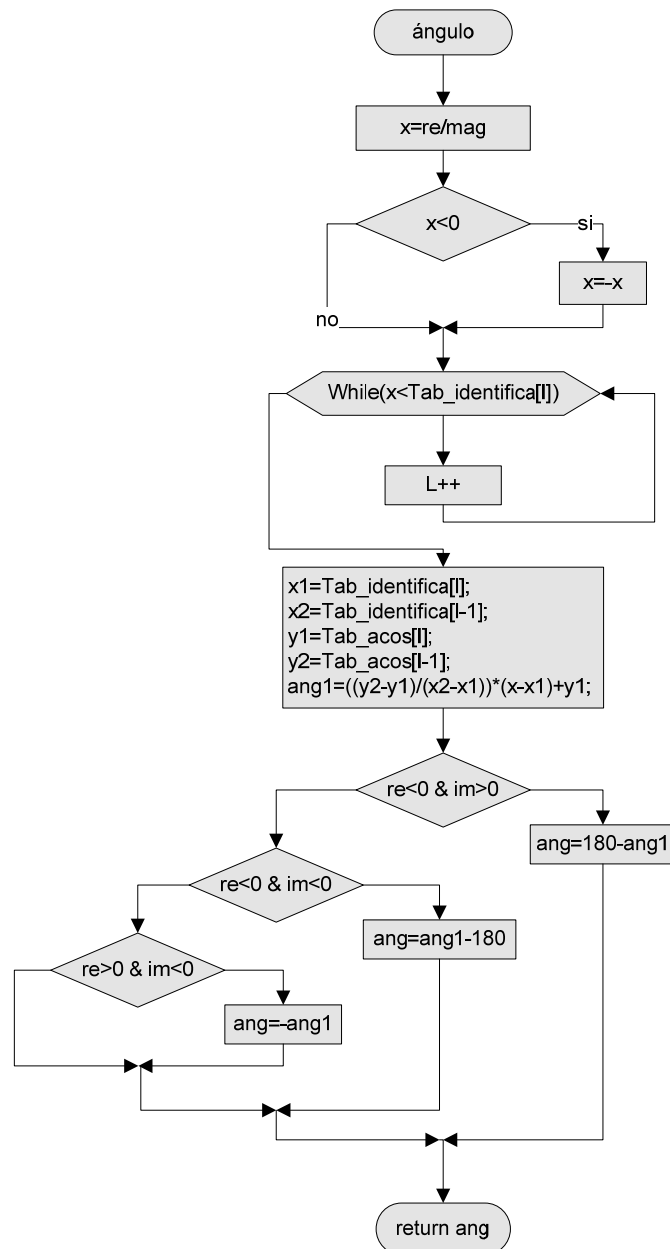


Figura 3.18.- Diagrama da flujo para el cálculo del ángulo.

3.4.2 Estimación de componentes simétricas.

Una de las herramientas más poderosas para trabajar con circuitos trifásicos desbalanceados es el método de las componentes simétricas desarrollado por Fortescue [40].

El trabajo de Fortescue prueba que un sistema desbalanceado de n fasores relacionados, se puede resolver con n sistemas de fasores balanceados llamadas componentes simétricas de los fasores originales [41] y su ecuación se muestra en (3.1).

$$\begin{bmatrix} V_a^{(0)} \\ V_a^{(1)} \\ V_a^{(2)} \end{bmatrix} = \frac{1}{3} \begin{bmatrix} 1 & 1 & 1 \\ 1 & a & a^2 \\ 1 & a^2 & a \end{bmatrix} \cdot \begin{bmatrix} V_a \\ V_b \\ V_c \end{bmatrix} \quad (3.1)$$

Donde: $a = 1\angle 120^\circ$
 $a^2 = 1\angle 240^\circ$

La ecuación 3.1 muestra como descomponer tres fasores asimétricos en tres fasores simétricos. La componente de secuencia positiva consiste en tres fasores de igual magnitud desplazados uno del otro por una fase de 120° y que tienen la misma secuencia de fase que los fasores originales [41], su ecuación se muestra en (3.2).

$$V_a^{(1)} = \frac{1}{3} (V_a + aV_b + a^2V_c) \quad (3.2)$$

Componentes de secuencia negativa consiste en tres fasores de igual magnitud desplazados uno del otro por una fase de 120° y que tienen una secuencia de fase opuesta que los fasores originales [41], su ecuación se muestra en (3.3).

$$V_a^{(2)} = \frac{1}{3}(V_a + a^2V_b + aV_c) \quad (3.3)$$

Componentes de secuencia cero consiste en tres fasores de igual magnitud y con un desplazamiento de fase cero uno del otro [41], su ecuación se muestra en (3.4).

$$V_a^{(0)} = \frac{1}{3}(V_a + V_b + V_c) \quad (3.4)$$

Para calcular las componentes simétricas a partir de los fasores previamente estimados es necesario hacer las operaciones con números reales e imaginarios. A continuación se muestran los cálculos para la componente positiva.

$$V_a^{(1)} = \frac{1}{3}(V_a + aV_b + a^2V_c)$$

Sustituyendo el valor de a:

$$V_a^{(1)} = \frac{1}{3}(V_a + (0.5 + .866i)V_b + (0.5 - .866i)V_c)$$

Realizando las operaciones indicadas y teniendo en cuenta que $i^2 = -1$, se separan las componentes real e imaginario:

$$V_{a(REAL)}^{(1)} = \frac{1}{3}(V_{a(REAL)} + 0.5V_{b(REAL)} - 0.866V_{b(IMAG)} + 0.5V_{c(REAL)} + 0.866V_{c(IMAG)})$$

$$V_{a(IMAG)}^{(1)} = \frac{1}{3}(V_{a(IMAG)} + 0.5V_{b(IMAG)} + 0.866V_{b(REAL)} + 0.5V_{c(IMAG)} - 0.866V_{c(REAL)})$$

Se realizan la misma metodología para obtener la corriente de secuencia positiva, así como las componentes de secuencia negativa y cero si se requiere.

El diagrama de flujo se muestra en la figura 3.19.

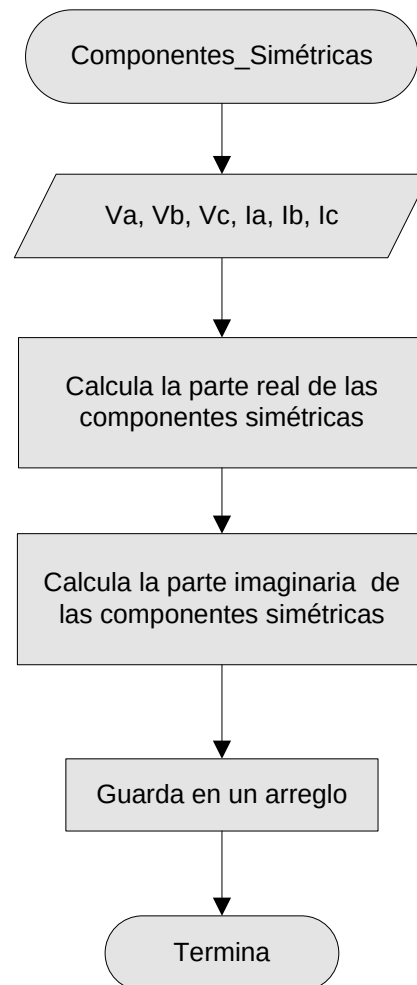


Figura 3.19.- Diagrama da flujo para el cálculo de componentes simétricas.

3.4.3 ADC.

En el IDE del “Bean Inspector” del periférico interno ADC. Se habilitan 6 canales de manera simultánea con una resolución de 12 bits cada uno [36]; las tres fases de Comisión Federal de electricidad (CFE) son monitoreadas por los pines con los identificadores AN0-PAN0, AN1-PAN1 y AN2-PAN2, para los voltajes y AN3-PAN3, AN4-PAN4 y AN5-PAN5 para las corrientes.

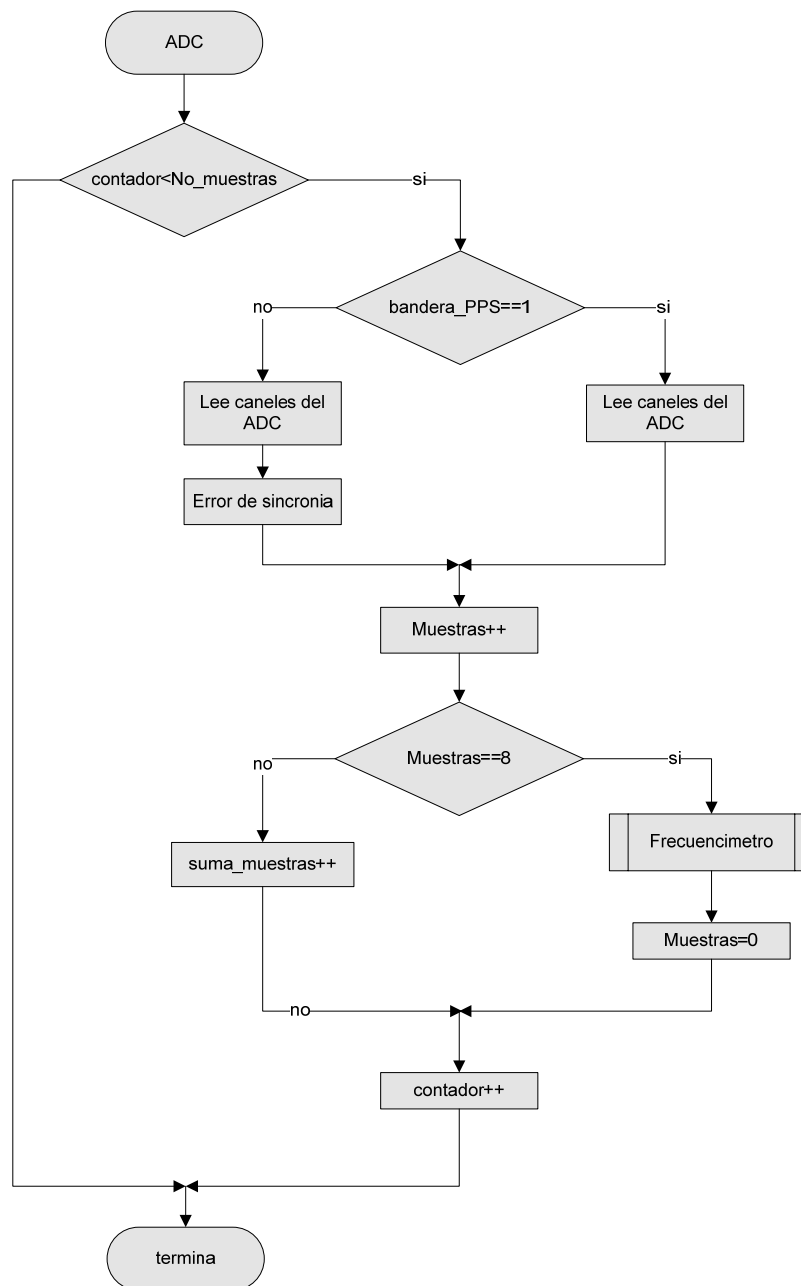


Figura 3.20.- Diagrama de flujo del ADC.

Del diagrama de flujo de la figura 3.20, se observa que se manda llamar una función, realizar parte de sus respectivos algoritmos, con la intención de aprovechar el tiempo que existe entre cada periodo de interrupción y no esperar hasta que se tengan las 128 muestras.

Lo primero que se hace es preguntar por la bandera_PPS, esta indica que el PMU esta sincronizado mediante el PPS; si la bandera es cero se procede a leer los canales del ADC e inmediatamente se indica que el PMU tiene un error de sincronía poniendo el Bit 13 de la palabra del STAT en 1.

3.4.4 Frecuencímetro.

La metodología implementada para determinar la frecuencia en tiempo real de la señal sinusoidal muestreada por el ADC, consiste en detectar los cruces por cero cuando se tiene una pendiente negativa.

En el capítulo 2 se muestra la ecuación 2.34 la cual se utiliza para calcular la fracción de muestra, donde y_1 es la muestra anterior y y_2 la muestra actual; en la ecuación 3.4 se muestra como se calcula la fracción de muestra con las variables actuales.

$$fraccion_muestra = \frac{muestra_anterior}{muestra_actual + muestra_anterior} \quad (3.4)$$

Debido a que las señales sinusoidales de voltaje son montadas sobre un nivel de CD para su codificación por los canales ADC del microcontrolador, las señales precisamente no realizan un cruce por cero, por lo que se monitorea su cruce a través del “Offset” agregado en su acondicionamiento.

A continuación se describe el algoritmo detalladamente.

Algoritmo: Calculo de la Frecuencia de una señal alterna digitalizada, en base a una frecuencia de muestreo constante. El algoritmo consiste en contar el número exacto de muestras y fracciones de muestras que caben en un ciclo de la señal a medir, multiplicarlos por el periodo de muestreo y obtener la inversa.

Precondición: Encontrar el primer cambio de signo de la señal a medir, de positivo a negativo, En ese momento se inicializa $\text{Suma_muestras} = 1 - \text{fracción_muestras}$.

Postcondición: $\text{Frec} = \text{Frecuencia de la señal muestreada, en cada cambio de signo de la señal a medir de positivo a negativo}$

Algoritmo detallado:

1. Si $\text{signo_muestra_actual} = \text{negativo}$ y $\text{signo_muestra_anterior} = \text{positivo}$
 - 1.1. $\text{Calcula } \text{fracción_muestra} = -\text{ad_anterior} / (\text{ad_actual} - \text{ad_anterior})$
 - 1.2. $\text{Suma_muestras} = \text{Suma_muestras} + \text{fracción_muestra}$
 - 1.3. $\text{Frecuencia} = 1 / (\text{Suma_muestras} * \text{Periodo})$
 - 1.4. $\text{Suma_muestras} = 1 - \text{fracción_muestra}$
2. Cualquier otro caso diferente a la condición (1)
 - 2.1. $\text{Suma_muestras} = \text{Suma_muestras} + 1$
3. $\text{signo_muestra_anterior} = \text{signo_muestra_actual}$
 $\text{ad_anterior} = \text{ad_actual}$

El microcontrolador realiza una operación en dos ciclos del reloj interno ($2 * (1/60\text{MHz}) = 33.333\text{ns}$) [35], el ADC captura 128 muestras por ciclo; recordando, la frecuencia de muestreo es de 7680, es decir; se tendría que ejecutar el algoritmo de cálculo de frecuencia cada $130.21 \mu\text{s}$, tiempo suficiente para realizar el algoritmo, sin embargo el microcontrolador trabajaría de más, entonces se propone calcular la frecuencia cada 8 muestras y además el algoritmo no se ve afectado por el ruido. El algoritmo se ejecuta cada 1.042 ms ($1/(60*16)$ o $(8*130.21 \mu\text{s})$), de esta manera se tiene ahora una ventana de 16 muestras por ciclo.

El algoritmo no se debe ver afectado por el número de muestras por ciclo, siempre que se cumpla el teorema de muestreo de Nyquist; se realizaron pruebas cada 4, 8 y 16 muestras, realizando el algoritmo cada 8 se obtienen mejores resultados.

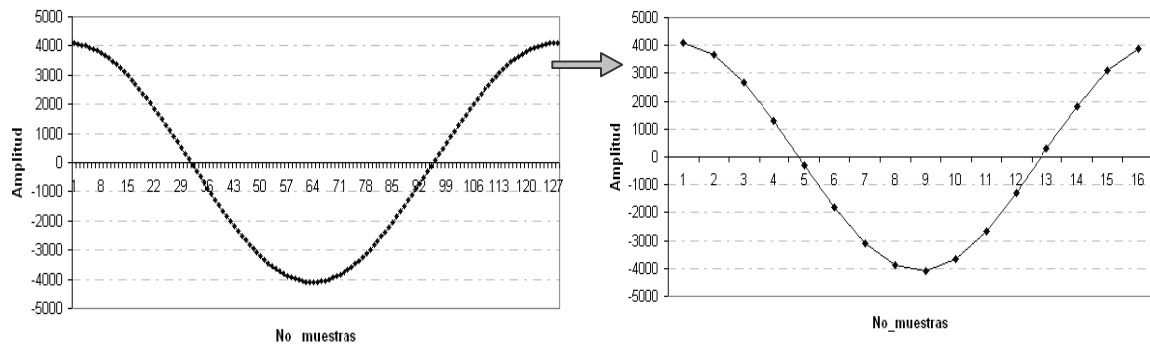


Figura 3.21.- Cálculo de la frecuencia cada 8 muestras

En la figura 3.21 se puede observar que la señal de entrada es puramente sinusoidal, al despreciar 7 muestras y tomar la octava muestra de una señal discreta de 128 muestras; se tiene una señal de 16 muestras por ciclo, esta no tiene alteración alguna.

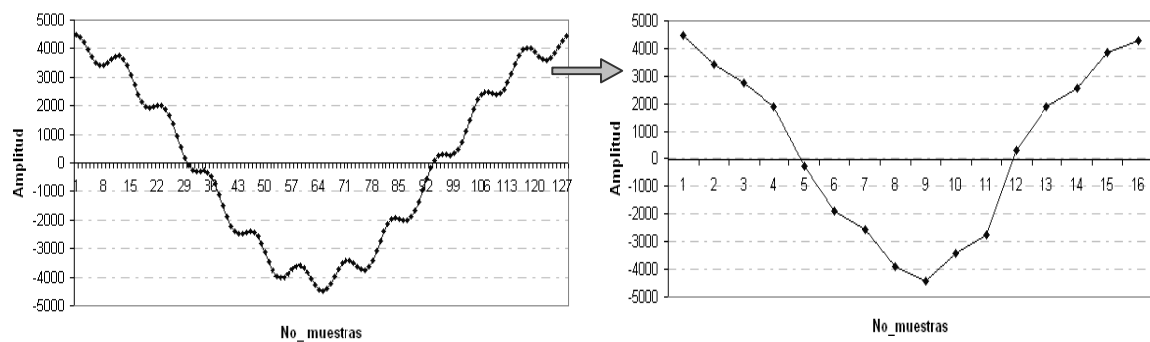


Figura 3.22.- Cálculo de la frecuencia cada 8 muestras con una señal contaminada de armónicos

Si la señal de entrada esta contaminada de armónicos se tiene una distorsión. Se realiza una prueba con el 10% del 11vo armónico; se elige con este armónico ya que la distorsión es más visible que con los otros armónicos.

En la figura 3.22 se muestra la señal de 16 muestras a partir de una señal contaminada con el 10% del onceavo armónico con 128 muestras, la señal es periódica de 60 Hz; el algoritmo del calculo de la frecuencia se puede aplicar sin ningún problema, ya que la señal que se tiene es periódica; en el capítulo 4 se muestran pruebas con diferentes señales de entrada.

En la figura 3.23 se muestra el diagrama de flujo del frecuencímetro.

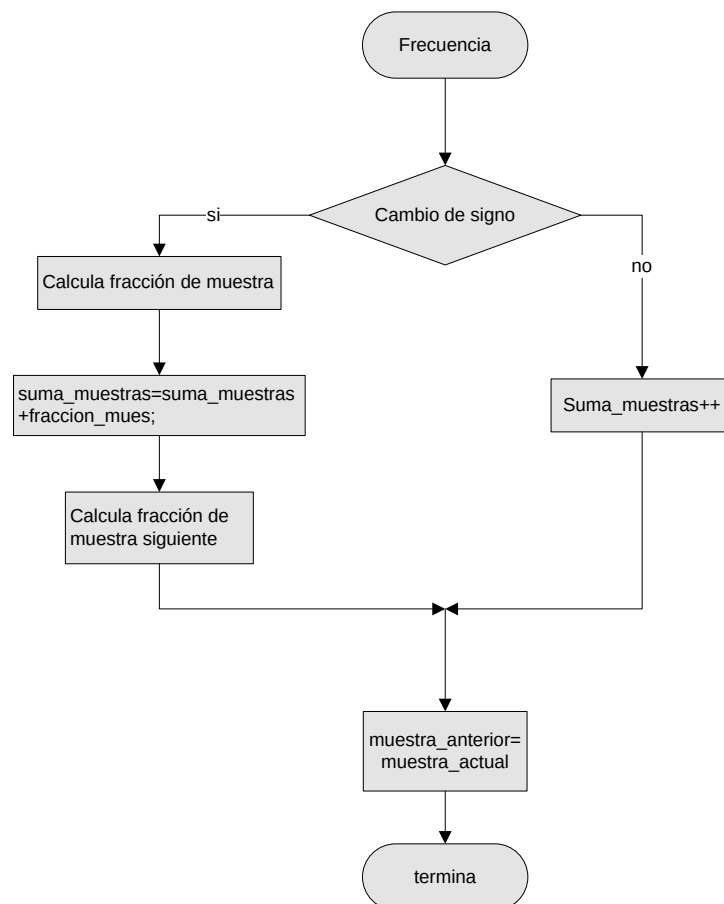


Figura 3.23.- Diagrama de flujo del frecuencímetro.

3.4.5 Temporizador para el periodo de muestreo.

Para dar inicio de las conversiones digital-analógicas (A/D) en los canales ADC se hace uso de un “TimerInt” que es una interrupción periódica que cada vez que se ejecute ^[34], indica el periodo de muestreo.

La inversa de la frecuencia de muestreo proporciona el periodo de interrupción, siendo este de $130.20833\mu\text{s}$ ($1/7680$) [38]. La figura 3.24 se muestra el diagrama de flujo requerido para configurar la señal de inicio de conversión de los canales ADC configurados.

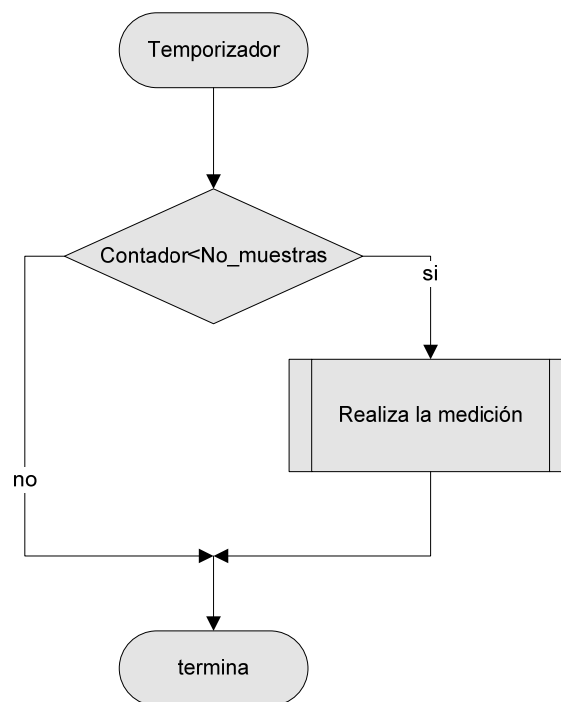


Figura 3.24.- Diagrama de flujo de la interrupción “TimerInt” para la frecuencia de muestreo de conversión del ADC.

3.5 Comunicaciones.

Como ya se ha planteado en esta tesis es necesario un protocolo de comunicaciones para el intercambio de información entre los diferentes PMU instalados en la red y el maestro o sistemas de control. El estándar IEEE C37.118 proporciona en su cláusula 6 un protocolo de comunicación [43], esta cláusula describe los formatos de mensaje de y para el PMU para la comunicación en tiempo real.

El protocolo permite la identificación de información necesaria, como IDCODE del PMU y el estado, para correcta interpretación del maestro de los datos medidos por el PMU [43].

Cuatro tipos de mensajes son definidos: datos, configuración, encabezado y comandos. Los primeros 3 son transmitidos por el PMU y el último es recibido.

Al igual que el subsistema de medición, el subsistema de comunicación incluye diferentes bloques de programación, los cuales pueden ser descritos por separado. En la figura 3.21 el diagrama de secuencias del subsistema de comunicación, aquí se muestra la interacción en tiempo real de los bloques que lo constituyen.

El diagrama de secuencias mostrado en la figura 3.25 se encuentra en un ciclo infinito esperando las órdenes del maestro.

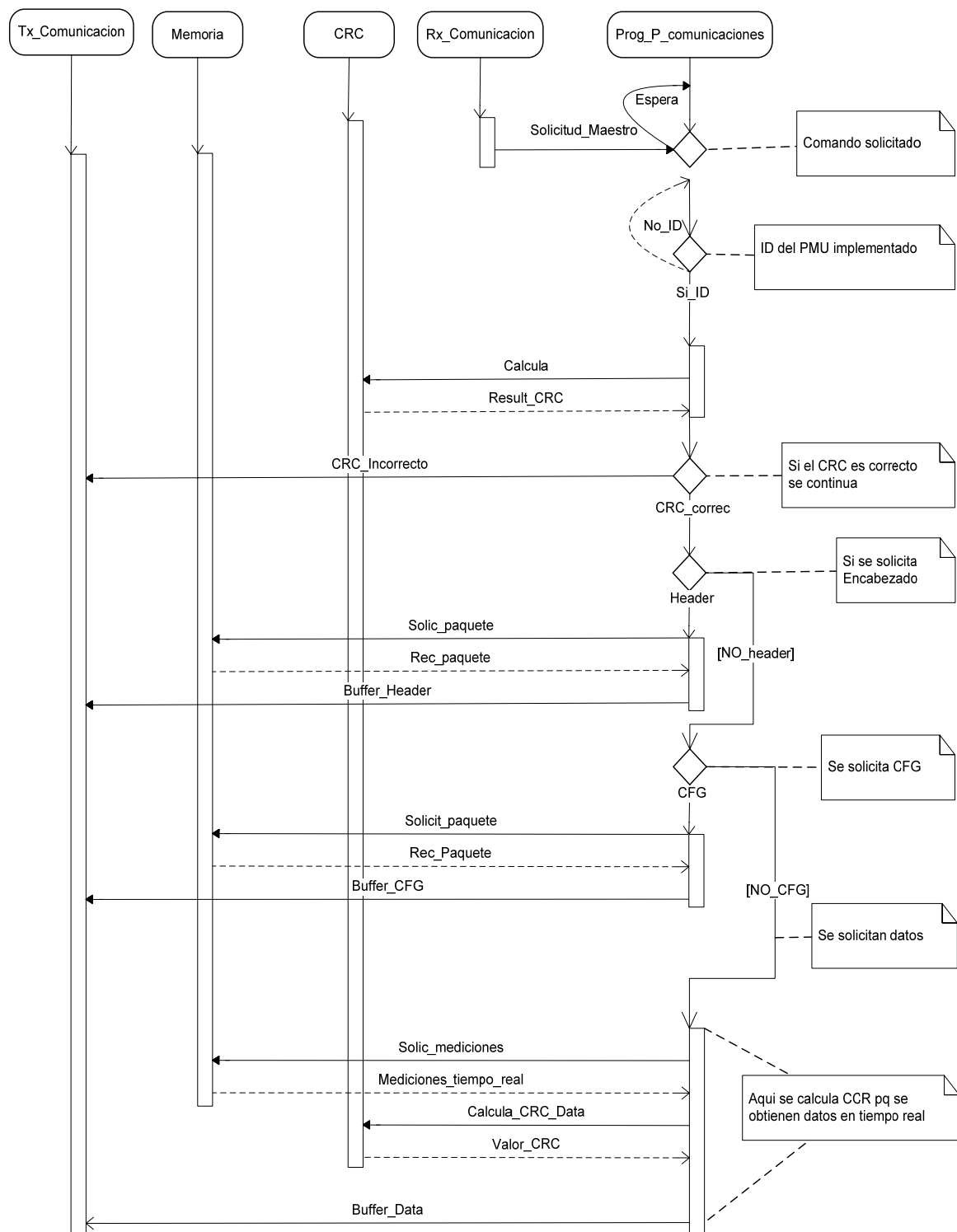


Figura 3.25.- Diagrama de secuencias de comunicaciones.

3.5.1 Programa principal de comunicaciones.

Lo primero que hace el programa principal de comunicaciones es esperar la trama de comandos enviada por el maestro o sistema de control, para tomar las acciones apropiadas [43] para esto es necesario saber que datos contiene.

En la tabla 3.3 se muestra como esta constituido el marco de comando y los diferentes comandos enviados del maestro al PMU se muestran en la tabla 3.4.

Tabla 3.3 Organización de la trama de comandos

No.	Campo	Tamaño	Descripción
1	SYNC	2	Tipo de trama y número de versión
2	FRAME SIZE	2	Número de bytes en la trama
3	IDCODE	2	Número del PMU, 16 BIT enteros
4	SOC	4	Estampado de tiempo
5	FRACSEC	4	Fracción de segundo y calidad de tiempo
6	CMD	2	Comando que esta siendo enviado al PMU
8	CRK	2	Para verificar CRC

En la tabla 3.3 se define una longitud de 18 en el mensaje que envía el maestro [43], en la posición 15 se encuentra un elemento muy importante, este es el comando definido en la tabla 3.4.

Con estos datos el PMU debe contestar correctamente a la requisición del maestro.

Tabla 3.4 Comandos enviados al PMU

No. Comando	Palabra del comando en bits	Definición
	Bits 15-4	Reservados para usos futuros
Bits 3-2-1-0		
1	0001	Terminar la transmisión de la trama de datos
2	0010	Comenzar la transmisión de la trama de datos
3	0011	Envía el archivo de encabezado
4	0100	Envía el archivo de CFG-1
5	0101	Envía el archivo de CFG-2

En cuanto a la longitud y contenido del mensaje de la configuración se muestran en la tabla 3.5, los campos del mensaje de configuración se describen en esta.

El mensaje de configuración mostrado en la tabla 3.5 (valor hexadecimal) es del PMU implementado en esta tesis, es decir; son valores particulares de cada PMU, la descripción más detallada se encuentra en la cláusula 6.4 en las tablas 9 y 10; el mensaje de datos es definido en la cláusula 6.3 en la tabla 7 y 8 del estándar C37.118.

El programa de comunicación lo primero que hace es preguntar si se ha recibido alguna solicitud por el maestro, de ser así se ejecuta el código siguiente, si no se ha recibido entonces se queda en un bucle infinito realizando la misma pregunta.

Tabla 3.5 Mensaje de configuración del PMU

CONFIGURACION			
Campo	Tamaño	Descripción	Valor Hexadecimal
SYNC	2	Versión	AA 31
FRAMESIZE	2	Tamaño del mensaje	00 B6
IDCODE	2	Nombre del PMU	00 01
SOC	4	Estampado de tiempo	44 85 27 F0
FRACSEC	4	Fracciones de segundo	56 07 10 98
TIME BASE	4	Resolución del estampado de tiempo	00 0F 42 40
NUM PMU	2	Nombre/número del PMU	00 01
STN	16	Nombre de la estación = STATION SEPI	53 54 41 54 49 4F 4E 20 53 45 50 49 20 20 20 20
IDCODE	2	Nombre/número del PMU	00 01
FORMAT	2	Formato de los datos, aquí se define si se envían las componentes real e imaginario o el fasor (magnitud y ángulo)	00 04
PHNMR	2	Número de fasores	00 06
ANNMR	2	Número de valores analógicos	00 00
DGNMR	2	Número de valores digitales	00 00
CHNAM	96	Nombre de los fasores	
		V FASE 1	56 20 46 41 53 45 20 31 20 20 20 20 20 20 20 20
		V FASE 2	56 20 46 41 53 45 20 32 20 20 20 20 20 20 20 20
		V FASE 3	56 20 46 41 53 45 20 33 20 20 20 20 20 20 20 20
		I FASE 1	49 20 46 41 53 45 20 31 20 20 20 20 20 20 20 20
		I FASE 2	49 20 46 41 53 45 20 32 20 20 20 20 20 20 20 20
		I FASE 3	49 20 46 41 53 45 20 33 20 20 20 20 20 20 20 20
		Factor de conversión para los canales del fasor	
		V FASE 1	00 0D F8 47
		V FASE 2	00 0D F8 47
PHUNIT	24	V FASE 3	00 0D F8 47
		I FASE 1	01 00 B2 D0
		I FASE 2	01 00 B2 D0
		I FASE 3	01 00 B2 D0
		Factor de conversión de los canales analógicos	00 00 00 00
		Máscara para la palabra digital	00 00 00 00
FNOM	2	Frecuencia nominal	00 00
CFGCNT	2	Contador del cambio de configuración	00 00
DATA RATE	2	Tasa de transmisión de los datos	00 1E
CHK	2	Código de comprobación de redundancia cíclica	D4 3F

La recepción de la trama de comandos se realiza a través de una interrupción en el programa eventos llamada `AS1_OnFullRxBuf(void)`, su diagrama de flujo se muestra en la figura 3.26.

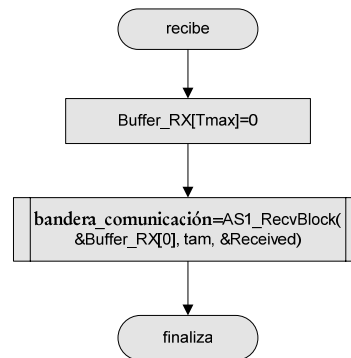


Figura 3.26.- Diagrama de flujo del evento que recibe.

La función `AS1_RecvBlock` es de tipo entero, regresa un 1 si existe algún error en la recepción de la trama y regresa un cero si no existe ningún error y la trama es recibida por completo.

Ya que se ha recibido el paquete de datos del maestro correctamente el PMU verifica si es el mismo IDCODE, este puede definirse como el nombre del PMU; ya que en la red pueden existir varios PMU's conectados, esto se hace a través del IDCODE. Si es el mismo IDCODE ahora verifica que no tenga ningún error en la recepción del paquete de datos, esto se verifica calculando el CRC (comprobación de redundancia cíclica), este garantiza la integridad de los datos. El diagrama del programa de comunicaciones flujo se muestra en la figura 3.26, el número de caso esta directamente ligado con el número de comando definido en la tabla 3.4.

Se ejecuta este programa cuando la bandera respuesta esta en cero, esto quiere decir que el PMU ha recibido la trama de comandos del sistema de control o maestro correctamente.

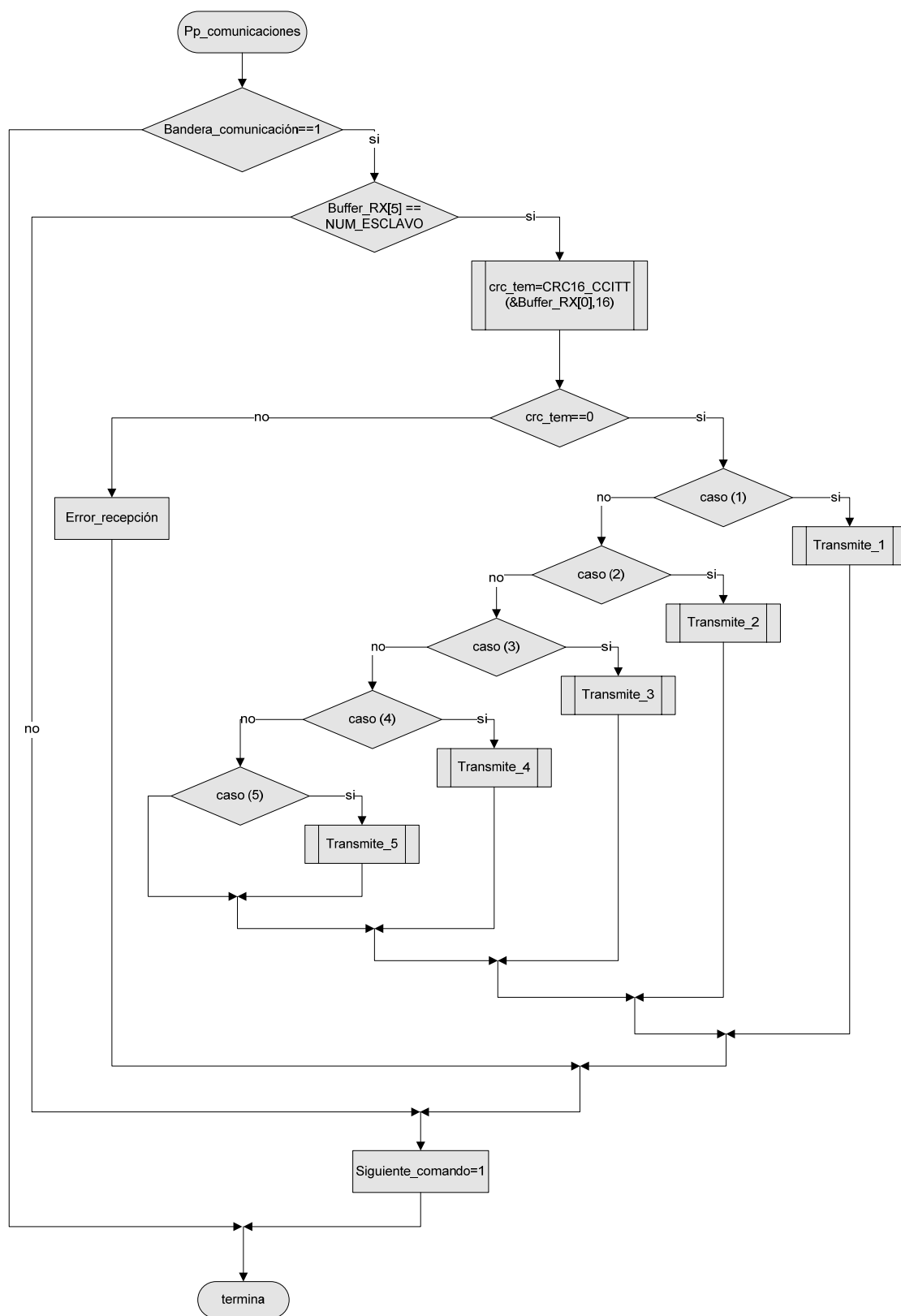


Figura 3.27.- Diagrama del programa principal de comunicaciones.

Como se observa en el diagrama de flujo de la figura 3.27, hay dos subrutinas: la del cálculo del CRC y la de transmisión, la última es llamada varias veces para enviar un buffer diferente en cada caso.

3.5.2 Comprobación de redundancia cíclica.

El código del CRC es utilizado para la detección de error en la transmisión de datos digitales [43, 44]. Un marco de control de secuencia se adjunta al mensaje original para la detección de errores. CRC se ha empleado eficazmente en muchos protocolos de comunicación [45]

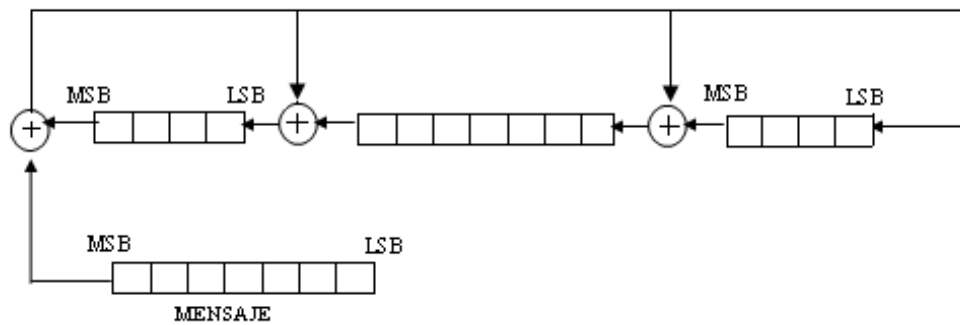
El código CRC es descrito por un polinomio generador $g(x)$ de grado p , mostrado en la ecuación 3.6; este polinomio ha sido adaptado como norma, comúnmente es el código de comprobación de redundancia cíclica con un polinomio de 16 bits (CRC-CCITT) [43, 45].

$$\text{CRC-CCITT} = 10001000000100001 = X^{16} + X^{12} + X^5 + 1 \quad (3.6)$$

En el cálculo del CRC se ven los datos como números binarios, a estos bits de datos se les añade p bits de redundancia de forma que el polinomio resultante sea divisible por el polinomio generador.

El receptor verifica que el polinomio recibido sea divisible por $g(x)$, si no lo es habrá un error en la transmisión [43].

El código del CRC comúnmente referido como CRC-CCITT se ilustra en la figura 3.28.



$$g(x) = x^{16} + x^{12} + x^5 + 1, \quad \text{Condición inicial FFFFh}$$

Figura 3.28.-Funcionamiento del CRC.

El diagrama de flujo del CRC se muestra en la figura 3.29.

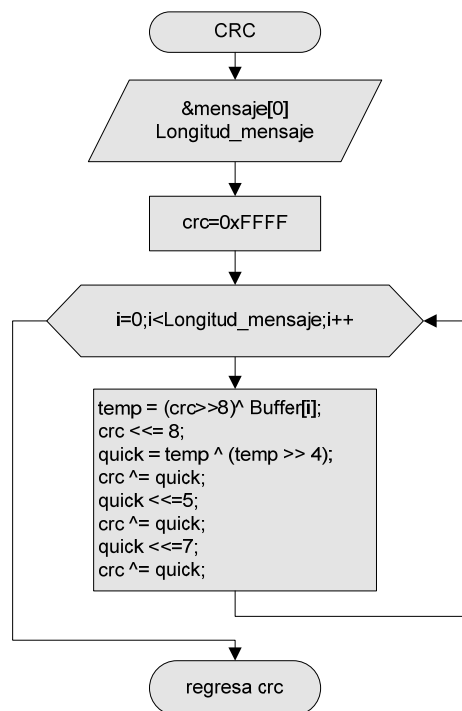


Figura 3.29.- Diagrama de flujo del cálculo del CRC.

La función transmite se utiliza para transmitir los paquetes de datos o configuración del PMU hacia el maestro, su diagrama de flujo se muestra en la figura 3.30.

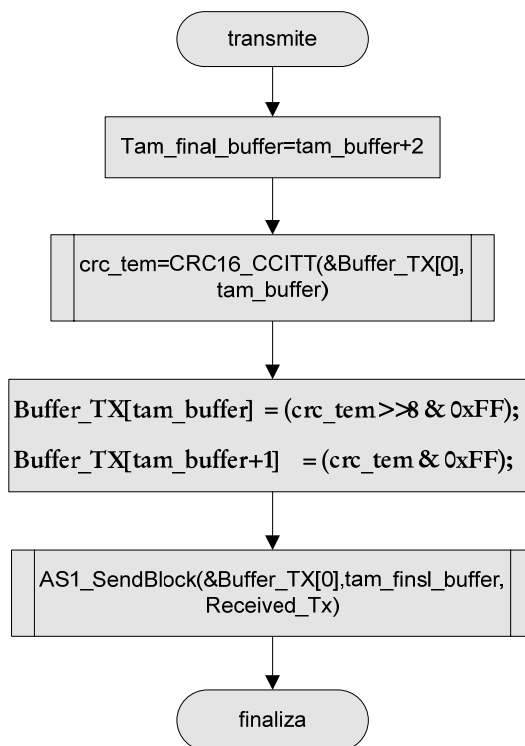


Figura 3.30.- Diagrama de flujo de la función transmite.

En el diagrama de flujo de la función transmite primero se calcula el CRC del paquete de datos que solicita el maestro e inmediatamente se incorpora este valor calculado a la trama de datos que envía el PMU.

Si el maestro solicita al PMU la configuración, este después de enviar los datos requeridos se mantiene en estado de alerta para recibir cualquier otro requerimiento. Ahora bien, si los datos (fasores del sistema) son solicitados, el PMU envía los datos constantemente a una tasa 30 veces por segundo, esa tasa de actualización, en un sistema de 60Hz puede ser 10, 12, 15 20 o 30 mensajes por segundo y es definido por el estándar C37.118 en la cláusula 6.

CAPÍTULO 4.- Pruebas del PMU Implementado para verificar que cumple con el estandar IEEE C37.118

En este capítulo se analiza el PMU normalizado mediante el estándar C37.118, se muestran las diferentes características con las que debe contar para que cumpla con el estándar.

Otro punto es mostrar los resultados de las mediciones del PMU, analizando la respuesta del mismo ante diferentes señales de prueba generadas en Excel y calcular el vector de error total (TVE) para cada una de las señales de prueba para verificar que no salgan del valor especificado por el estándar IEEE C37.118.

4.1 Introducción

Con señales en tiempo real, es necesario definir un tiempo de referencia para poder medir los ángulos de fase en forma sincronizada, el estándar IEEE 1344-1995 define el inicio de un segundo como el tiempo de referencia para establecer el valor del ángulo del fasor [46].

La utilidad de un fasor aumenta si pueden ser referidos a una base de tiempo común. Esto puede ser logrado si la señal de entrada que se mide en varios sitios se sincroniza una respecto a la otra. El estándar IEEE 1344-1995 define que una forma de onda en estado estacionario es aquella en la que la magnitud, frecuencia y ángulo de fase no varían con el tiempo [33, 34]. La forma de onda sinusoidal pura se representa mediante la ecuación 4.1 es comúnmente representada como un fasor, mostrado en la ecuación 4.2.

$$x(t) = x_m \cos(\omega t + \varphi) \quad \text{Ec. (4.1)}$$

$$x = x_r + jx_i = \left(\frac{x_m}{\sqrt{2}} \right) (e^{j\varphi}) \quad \text{Ec. (4.2)}$$

Donde φ es el defasamiento y x_m es el valor pico de la señal sinusoidal.

El concepto básico de fasor es adaptado como la representación de señales sinusoidales de sistemas de potencia como se muestra en la figura 4.2

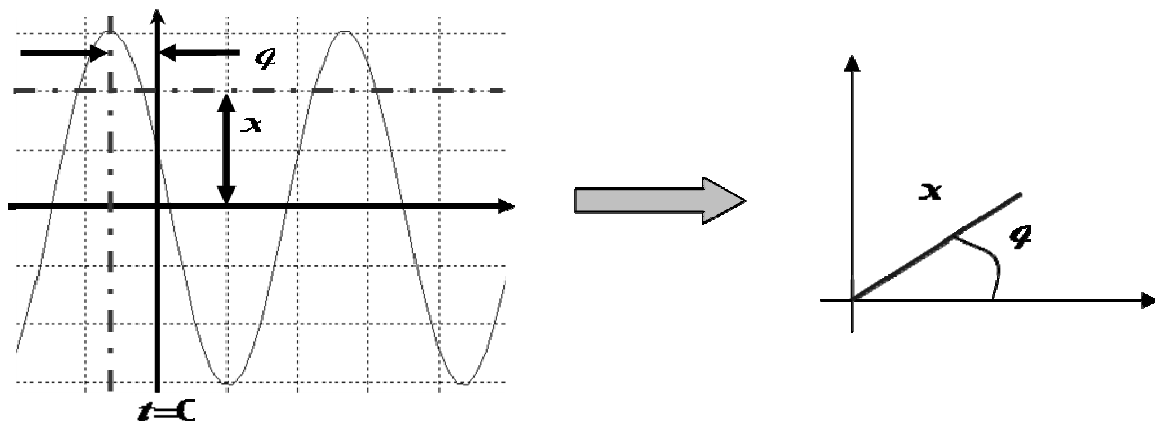


Figura 4.1.- Representación fasorial de una onda sinusoidal.

4.2 Medición de frecuencia.

En este apartado se realizan pruebas al algoritmo del cálculo de la frecuencia cuando es nominal y cuando esta fuera de la nominal, además de realizar pruebas ejecutando el algoritmo cada 4, 8 y 16 muestras por ciclo, obteniendo un mejor resultado cuando se ejecuta cada 8 muestras.

La primera prueba que se realiza con una señal sinusoidal sin perturbaciones a una frecuencia nominal de 60Hz, se ejecuta el algoritmo de cálculo de frecuencia cada 4, 8 y 16 muestras y los resultados se muestran en la tabla 4.1.

Tabla 4.1 Cálculo de la frecuencia con una señal sinusoidal pura

# muestras	Frecuencia real	Frecuencia calculada	Error
4	60	60.000	0
8	60	60.000	0
16	60	60.000	0

En la primer columna se tienen el número de muestras que utiliza el algoritmo, en la columna 2 se muestra la frecuencia real (se sabe que es la real ya que la señal es generada a través de Excel con esta frecuencia), en la tercer columna se muestra la frecuencia calculada por el algoritmo implementado y la tercer columna indica el error de la estimación de la frecuencia.

En esta prueba se tiene un error de cero para los tres casos, esto se debe a que la señal no tiene perturbación alguna y no se puede observar con claridad cual es la diferencia de ejecutar el algoritmo cada 4, 8 y 16 muestras.

Ahora bien en la red eléctrica no se tiene una señal sinusoidal pura, por el contrario está contaminada de armónicos y ruido; los armónicos más comunes en el sistema eléctrico son [47]: el 5°, 7°, 11° y 13° armónicos.

En la grafica 4.5 se muestra la señal de entrada sinusoidal pura con una frecuencia nominal de 60 Hz agregándole el 3% del 5° y 11° armónico y el 25% del 7° y 12°. Se puede observar en la gráfica de la figura 4.2 a) que existen 4 cambios de signo con pendiente positiva, esto tomando las 128 muestras de la señal de entrada. El cálculo de la frecuencia se hace tomado en cuenta los cambio de signo de positivo a negativo por consiguiente si se ejecuta el algoritmo cada muestra el calculo es erróneo.

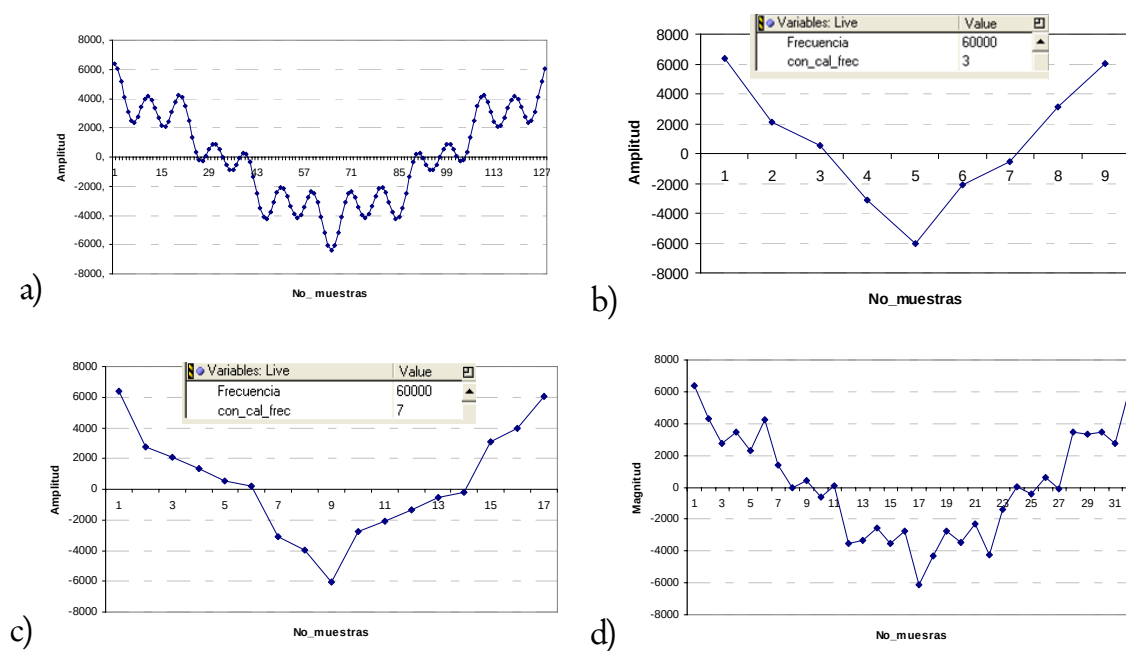


Figura 4.2.- Cálculo de la frecuencia para una señal sinusoidal pura.

En la figura 4.2 d) cuando el algoritmo de frecuencia se ejecuta cada 4 muestras se observa que en un ciclo de 60Hz se tienen tres cambios de signo positivo a negativo, para una señal de entrada contaminada de armónicos; por lo cual el cálculo de la frecuencia será erróneo, en este caso.

La frecuencia estimada al transcurrir un ciclo cuando el algoritmo corre cada 4, 8 y 16 muestras se observa en la tabla 4.2.

Tabla 4.2 Cálculo de la frecuencia con una señal sinusoidal pura más 20% del séptimo y onceavo armónico

# muestras	Figura	Frecuencia real	Frecuencia calculada	Error
4	4.2 d)	60	673.967	613.967
8	4.2 c)	60	60.000	0
16	4.2 b)	60	60.000	0

En los resultados mostrados en la tabla 4.2, cuando el algoritmo corre cada 4 muestras la frecuencia calculada después de un ciclo completo genera un error extremadamente elevado, esto se debe a que existen varios cambios de signo en un ciclo. Para el caso donde el algoritmo corre cada 8 y 16 muestras obtienen un error de cero.

Veamos ahora que resultados obtenidos en el algoritmo si se tiene una señal a una frecuencia nominal de 60Hz con ruido, es común encontrar este tipo de señales en la red. Los resultados se muestran en la figura 4.3.

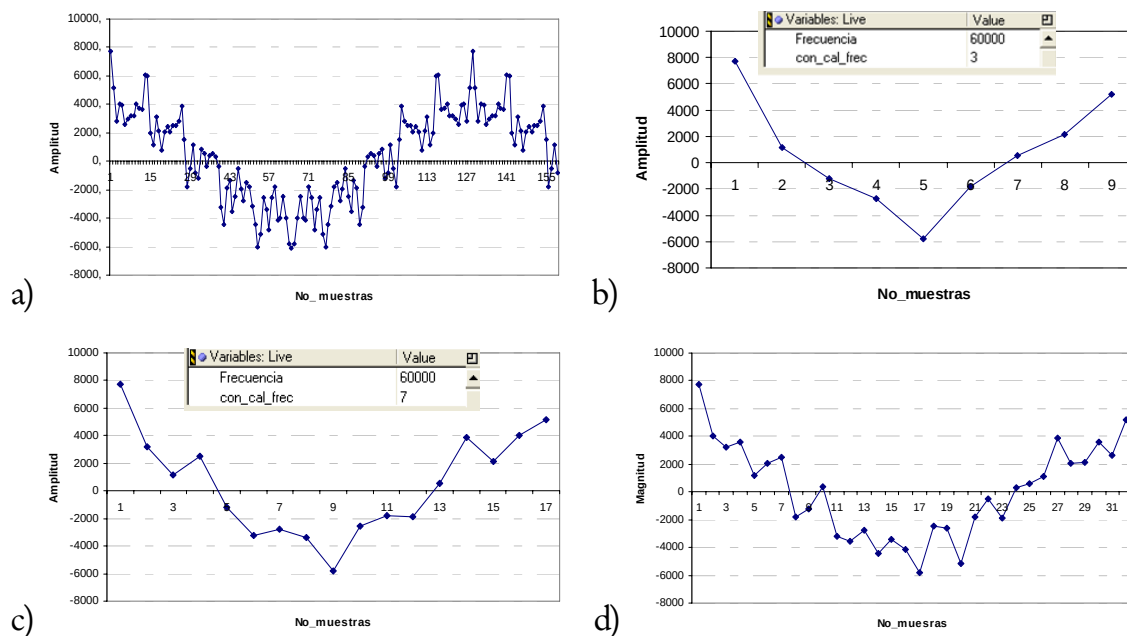


Figura 4.3.- Cálculo de la frecuencia para una señal sinusoidal pura.

Cuando se tiene una señal de entrada con ruido y se ejecuta el algoritmo cada 4 muestras se tienen más de dos cambios de signo al igual que el caso anterior, siempre que se tenga esta condición el calculo de la frecuencia será errónea; es por esta razón que el algoritmo no se ejecuta cada 4 muestras.

Tabla 4.3 Cálculo de la frecuencia con una señal sinusoidal con ruido

# muestras	Figura	Frecuencia real	Frecuencia calculada	Error
4	4.4 d)	60	65.141	5.141
8	4.4 c)	60	60.000	0
16	4.4 b)	60	60.000	0

En la tabla 4.3 se observan los resultados cuando se tiene una señal con ruido y se aprecia nuevamente el error al ejecutar el algoritmo cada 4 muestras, al igual que el caso anterior en los otros dos casos se tiene un error de cero.

Se realizan pruebas con la frecuencia por debajo de la nominal con una señal de entrada pura, En la tabla 4.4 se ejecuta el algoritmo cada 4 muestras, se observa que el error máximo se tiene cuando hay una frecuencia de 58 Hz y es de $2e-3$.

Tabla 4.4 Cálculo de la frecuencia cada 4 muestras

Frecuencia Real	Frecuencia Calculada	Error
59.99	59.990	0
59.95	59.950	0
59.90	59.899	1e-3
59.78	59.780	0
59.50	59.500	0
58.00	57.998	2e-3
57.50	57.499	1e-3
55.00	55.000	0

Tabla 4.5 Cálculo de la frecuencia cada 8 muestras

Frecuencia Real	Frecuencia Calculada	Error
59.99	59.990	0
59.95	59.950	0
59.90	59.899	1e-3
59.78	59.779	1e-3
59.50	59.499	1e-3
58.00	58.008	8e-3
57.5	57.506	6e-3
55.00	55.003	3e-3

Cuando el algoritmo de frecuencia se ejecuta cada 8 muestras se obtienen los resultados de la tabla 4.5. Se observa que el error máximo al igual que en caso anterior se tiene cuando la señal de entrada esta a una frecuencia de 58 Hz y es de 8e-3.

Tabla 4.6 Cálculo de la frecuencia cada 16 muestras

Frecuencia Real	Frecuencia Calculada	Error
59.99	59.990	0
59.95	59.948	2e-3
59.90	59.893	7e-3
59.78	59.768	12e-3
59.50	59.478	22e-3
58.00	58.056	56e-3
57.5	57.559	59e-3
55.00	55.066	66e-3

Cuando el algoritmo de frecuencia se ejecuta cada 16 muestras se obtienen los resultados de la tabla 4.6. Se observa que el error máximo se tiene cuando la señal de entrada esta a una frecuencia de 55 Hz y es de $66e-3$.

Para las pruebas realizadas con señales de frecuencia menor a la nominal se tiene que el mejor desempeño del algoritmo del cálculo de la frecuencia es cuando se ejecuta cada 4 muestras ya que se obtiene el mínimo error.

En el caso cuando se corre cada 4 muestras el ruido afecta significativamente el cálculo y cada 16 muestras el error se incrementa significativamente con respecto a cada 8, es por esa razón que el algoritmo se implementa cada 8 muestras.

4.2.1 Fasores de una señal con frecuencia nominal.

Si la señal se observa a intervalos regulares $\{0, T_0, 2T_0, 3T_0, \dots, nT_0, \dots\}$, estos conducen a la representación de fasores $\{X_0, X_1, X_2, \dots\}$. Si el intervalo de observación T_0 es igual a un múltiplo entero del periodo de la señal de entrada ($T = 1/f$), entonces un fasor constante se obtiene en cada intervalo de observación [20].

Esto sucede cuando se mide una señal con frecuencia nominal de 60Hz, debido a que el periodo de muestreo constante es de $1.302 \text{ e-}4$ ($1/(128*60)$).

Las mediciones realizadas por el PMU para una señal con una frecuencia fundamental de 60Hz se muestran en la figura 4.4.

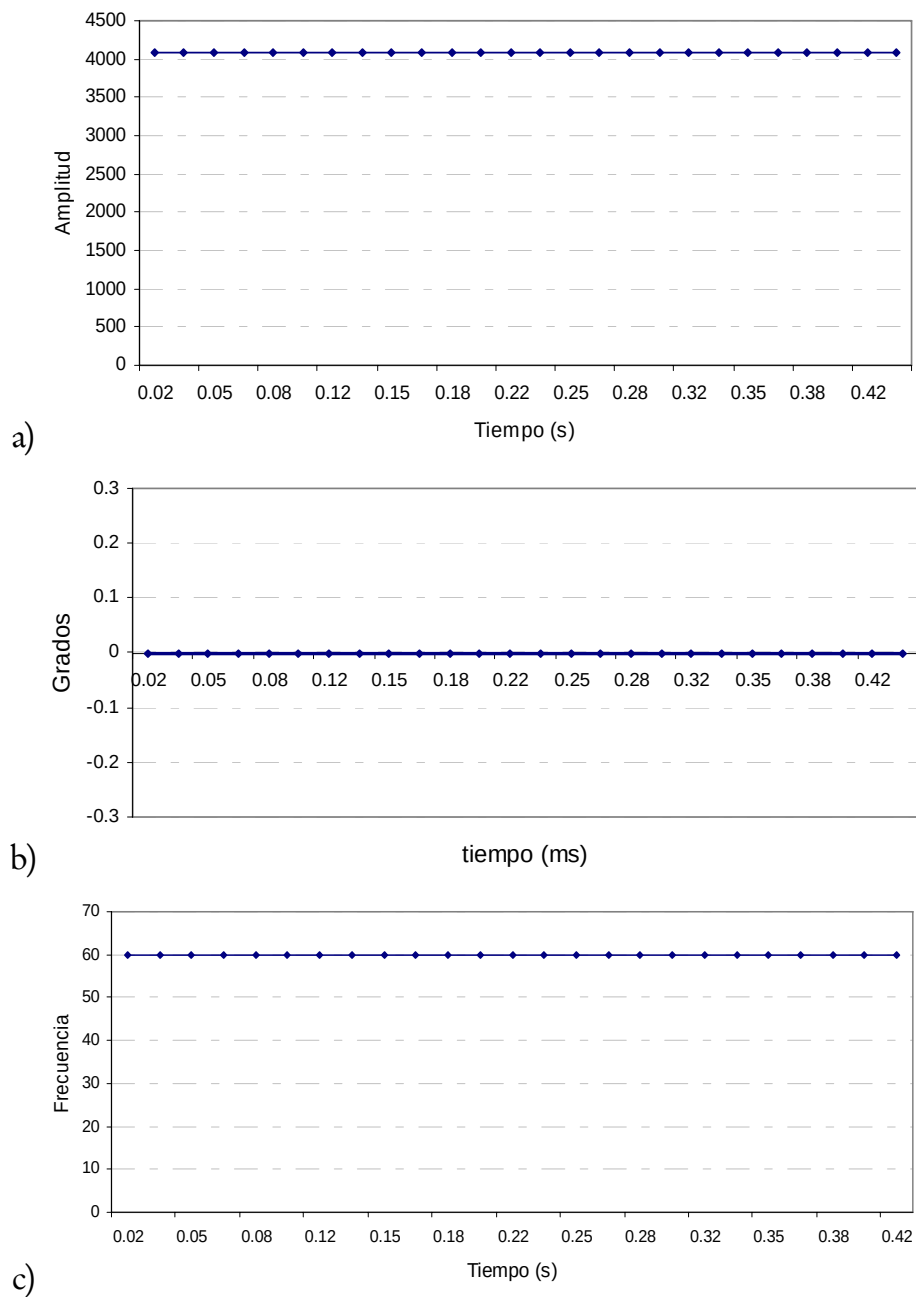


Figura 4.4.- Cálculo de los fasores a una frecuencia nominal.

La grafica de la magnitud se muestra en la figura 4.4 a), el ángulo en la b) y la frecuencia calculada en c), las mediciones son cada ciclo. La amplitud y frecuencia es constante debido a que no existe un cambio en la medición al igual que el ángulo del fasor es cero porque se mide una señal con frecuencia nominal

4.2.2 Fasores de una señal con frecuencia fuera de la nominal.

Si el intervalo de observación T_0 no es un entero múltiplo de T , el fasor observado tiene una magnitud constante, pero el ángulo de fase de la secuencia de fasores $\{X_0, X_1, X_2, \dots\}$ tendrá cambios de manera uniforme [43] como se muestra en la ecuación 4.3 y se muestra gráficamente en la figura 4.5.

$$a \text{ una razón de } 2\pi(f - f_0)T_0 \quad \text{Ec. (4.3)}$$

Donde $f_0 = 1/T_0$.

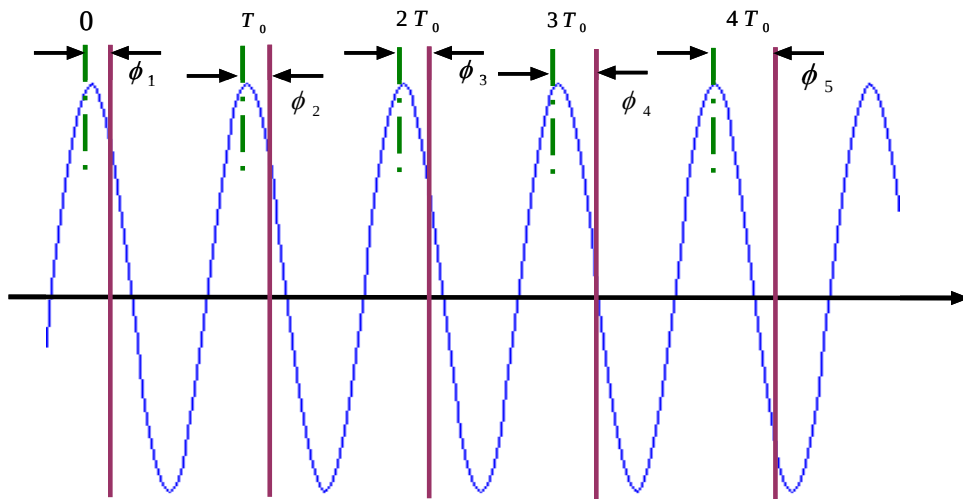


Figura 4.5.- Señal con un periodo de T observado en instante que son múltiplos de T_0 donde T_0 no es un múltiplo entero de T .

Esto es, se tienen variaciones en el ángulo de fase al tener una frecuencia diferente de la nominal; al aumentar o disminuir la frecuencia. La respuesta del ángulo de fase para una frecuencia nominal de 55 Hz se muestra en la figura 4.6.

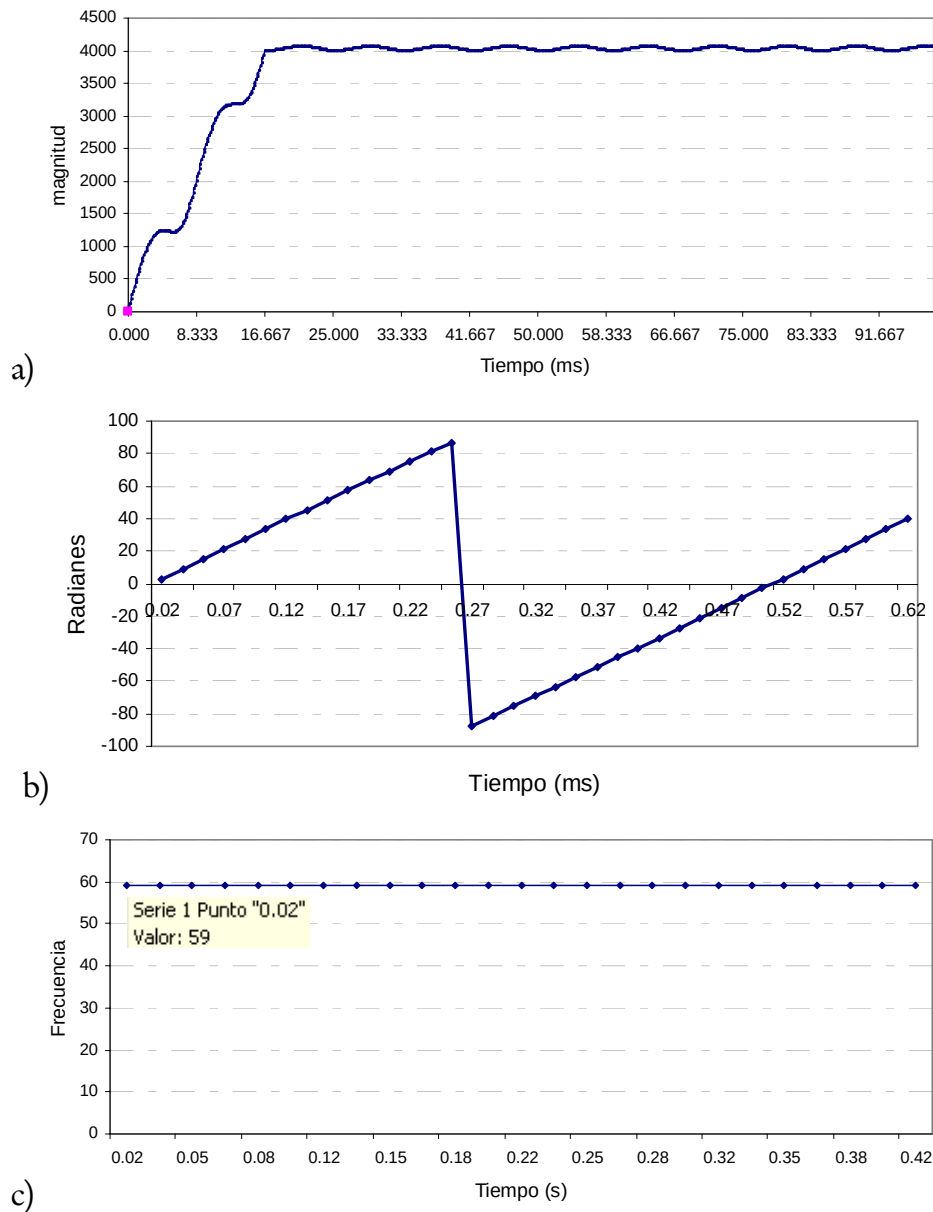


Figura 4.6.- Cálculo de los fasores a una frecuencia por debajo de la nominal.

La grafica de la magnitud se muestra en la figura 4.6 a) se observa una oscilación, el ángulo en la 4.6 b) aquí se observa que el fasor está en rotación debido a la frecuencia que está por debajo de la nominal y la frecuencia calculada en 4.6 c), esta es constante.

En la tabla 4.7 se muestran los resultados del sincrofasor a una frecuencia medida por debajo de la nominal comparando los resultados con del sincrofasor a una frecuencia nominal, ambos casos son con un defasamiento de 0° y se propone enviar 10 tramas por segundo y una frecuencia de 59Hz.

Tabla 4.7 Reportando mediciones 10 veces por segundo

Fracción de tiempo		Sincrofasor- 60Hz (0°)	Sincrofasor 59 Hz (0°)
Número de trama	Fracción de segundo		
0	0.0	$X_m/\sqrt{2}\angle 0^\circ$	$X_m/\sqrt{2}\angle 0^\circ$
1	0.1	$X_m/\sqrt{2}\angle 0^\circ$	$X_m/\sqrt{2}\angle -36^\circ$
2	0.2	$X_m/\sqrt{2}\angle 0^\circ$	$X_m/\sqrt{2}\angle -72^\circ$
3	0.3	$X_m/\sqrt{2}\angle 0^\circ$	$X_m/\sqrt{2}\angle -108^\circ$
4	0.4	$X_m/\sqrt{2}\angle 0^\circ$	$X_m/\sqrt{2}\angle -144^\circ$
5	0.5	$X_m/\sqrt{2}\angle 0^\circ$	$X_m/\sqrt{2}\angle -180^\circ$
6	0.6	$X_m/\sqrt{2}\angle 0^\circ$	$X_m/\sqrt{2}\angle 144^\circ$
7	0.7	$X_m/\sqrt{2}\angle 0^\circ$	$X_m/\sqrt{2}\angle 108^\circ$
8	0.8	$X_m/\sqrt{2}\angle 0^\circ$	$X_m/\sqrt{2}\angle 72^\circ$
9	0.9	$X_m/\sqrt{2}\angle 0^\circ$	$X_m/\sqrt{2}\angle 36^\circ$

Como se observa en la tabla 4.7 cuando la frecuencia medida por el PMU esta por debajo del nominal el fasor gira en sentido contrario de las manecillas del reloj. En la grafica 4.7 se observa el comportamiento del ángulo cuando la frecuencia es 55Hz.

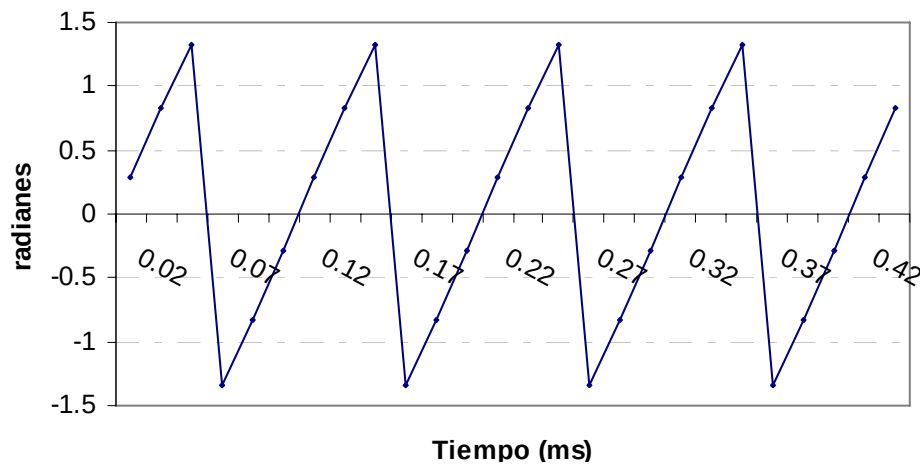


Figura 4.7.-Rotación del fasor a una frecuencia de 55Hz.

Cuando la frecuencia de la señal de entrada es exactamente igual a la nominal, el fasor será invariante en el tiempo. Cuando la señal de entrada tiene una frecuencia que no es igual a la nominal el fasor será sometido a una rotación en el plano complejo.

Cuando la frecuencia esta por arriba de la nominal el fasor estimado gira en sentido de las manecillas de reloj y su velocidad de giro depende de que tan alejado se encuentre el valor medido de la frecuencia nominal.

En la figura 4.8 a) se muestra la magnitud el fasor, en la 4.8 b) el ángulo del fasor a una frecuencia de 61 Hz y en la 4.9 se muestran los datos del fasor a una frecuencia de 65 Hz.

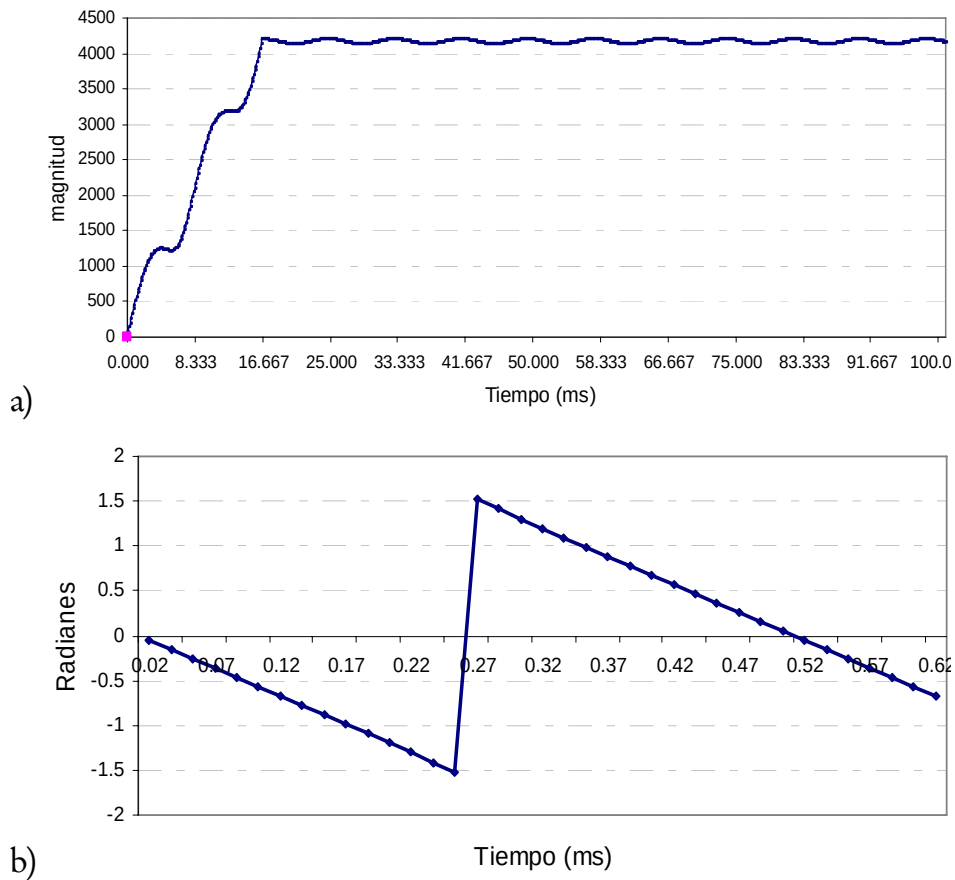


Figura 4.8.- fasor a una frecuencia de 61Hz.

En la figura 4.8 a) se muestra una oscilación la magnitud del fasor estimado, esto debido a que se estima la magnitud con el algoritmo de la transformada discreta de Fourier a una frecuencia nominal de 60Hz y conforme incrementa la diferencia entre la frecuencia nominal y la frecuencia medida la oscilación en la magnitud incrementara como se muestra en la figura 4.9 a).

El fasor gira más lento a una frecuencia de 61Hz en comparación cuando se tiene una frecuencia de 65Hz, esto se puede observar en la figura 4.8 b) y 4.9 b).

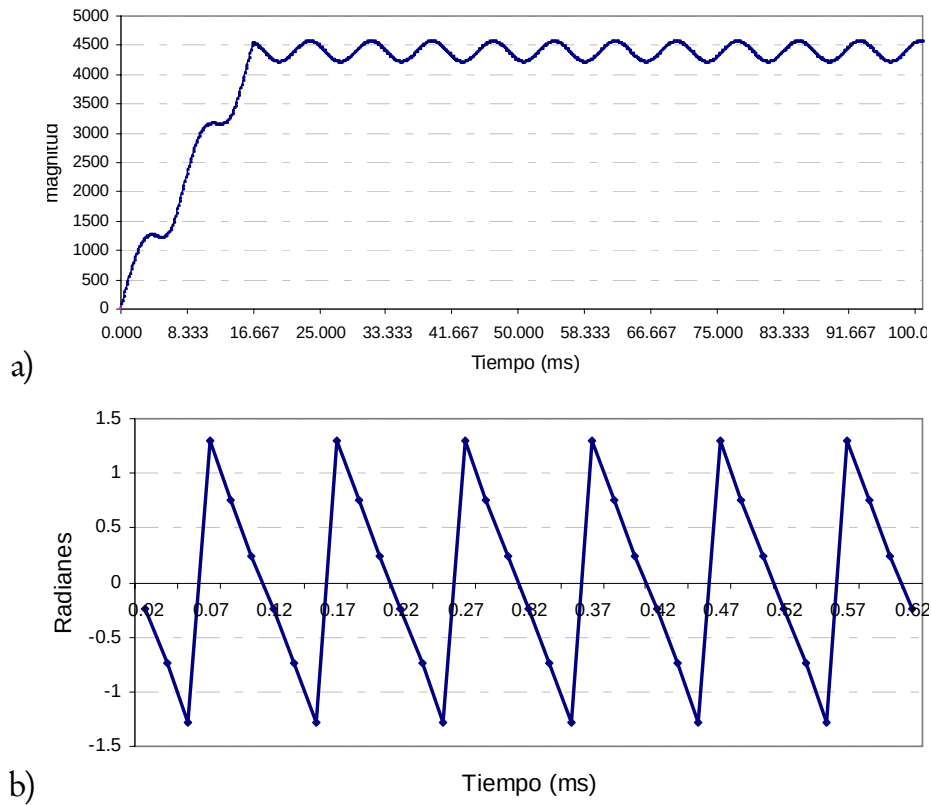


Figura 4.9.- fasor a una frecuencia de 65Hz.

4.3 Respuesta en el tiempo del PMU

La respuesta en el tiempo de un PMU se mide mediante un cambio en la magnitud de la señal del 100% al 110% [43], con frecuencia nominal. El tiempo de respuesta es el intervalo de tiempo que transcurre entre el instante en que se aplica el cambio y el instante que tarda en llegar a la magnitud especificada, la señal de entrada se muestra en la figura 4.10.

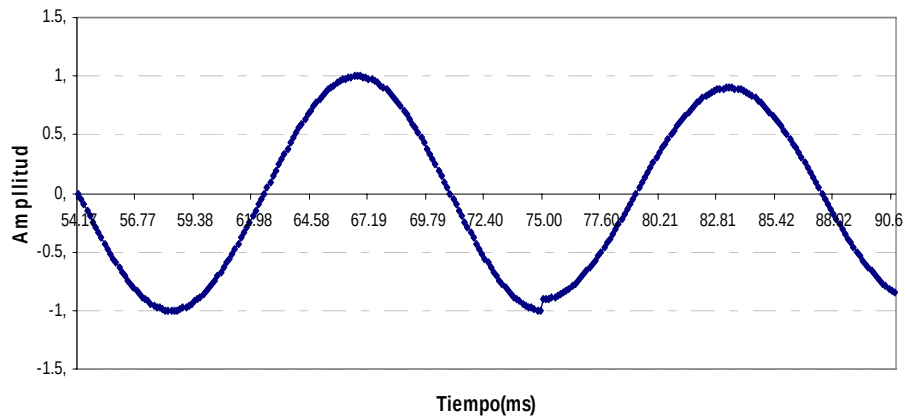


Figura 4.10.-Señal de entrada.

En este trabajo se estiman los fasores con el método de la DFT el cual requiere de una ventana de un ciclo para calcular los fasores, por lo tanto, requiere un ciclo para que se observen los cambios en magnitud. El tiempo de respuesta del PMU es de 16.666 ms y se muestra en la figura 4.11.

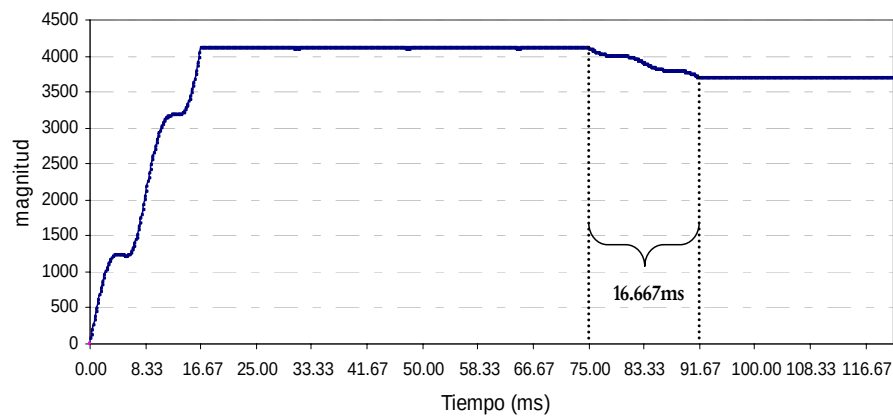


Figura 4.11.-Fasor de magnitud.

4.4 Límites de precisión.

Para cumplir con el estándar se utiliza el vector de error total (TVE), este permite cuantificar el error total del PMU [43].

La norma indica que bajo las condiciones donde X_m , ω , φ son fijos, el TVE no podrá superar el límite de 1% para el cumplimiento de determinado nivel.

El TVE es definido como se muestra en la ecuación 4.4:

$$TVE = \sqrt{\frac{(x_r(n) - x_r)^2 + (x_i(n) - x_i)^2}{x_r^2 - x_i^2}} \quad (4.4)$$

Donde:

$x_r(n)$ y $x_i(n)$ Son los valores medidos dados por el dispositivo de medición
 x_r y x_i Son los valores teóricos de la señal de entrada en el instante de tiempo de la medición y las condiciones conocidas de x_m , ω y φ .

Todas las pruebas se realizan con una señal patrón bajo las condiciones de referencia tal como se definen en la Tabla 4.8, los valores teóricos x_r & x_i son calculados de la señal patrón.

Tabla 4.8.- Límites de error admisibles para el cumplimiento de los niveles 0-1

	Condiciones de referencia	nivel 0		nivel 1	
		rango	TVE	rango	TVE
Frecuencia	F nominal	$\pm 0.5\text{Hz}$	1	$\pm 5\text{Hz}$	1
Magnitud	100%	80% a 120%	1	10% a 120%	1
Ángulo de fase	0 radianes	$\pm \pi$ radianes	1	$\pm \pi$ radianes	1
Distorsión armónica	<2%(THD)	1% cualquier armónico hasta el 50	1	10% cualquier armónico hasta el 50	1
Señal de interferencia	<0.2%	1% de la magnitud de señal de entrada	1	10% de la magnitud de señal de entrada	1

Es importante entender que la norma no requiere que se mida el fasor cuando se tiene la presencia de cualquier armónico, sino que al medir el fasor a la frecuencia fundamental no se tenga ningún error en presencia del 2^{do} al 50^{vo} armónicos.

La señal de interferencia es una señal fuera de banda con una frecuencia f_i , (Ec 4.5)

$$|f_i - f_0| > \frac{F_s}{2} \quad (4.5)$$

Donde:

F_s .- es la frecuencia de muestreo.

f_0 .- es la frecuencia nominal.

Para realizar pruebas de los niveles 0 y 1, se debe de variar solo un parámetro a la vez, es decir; cuando se realiza la prueba de magnitud, la magnitud de la señal debe ser variada, pero la frecuencia será nominal, el ángulo de fase 0 radianes, la distorsión armónica de menos del 0.2% y la señal de interferencia menos del 2%. El TVE debe permanecer dentro del margen de error en todas las pruebas.

Para el nivel 0 se realiza una variación de frecuencia de ± 0.5 Hz, en el nivel 1 se realiza una variación de ± 5 Hz, 10 veces más que en el nivel 0, en la magnitud existe una variación hasta del 10% y la distorsión armónica se incrementa hasta un 10%, como se puede observar en la tabla 1 el nivel 1 requiere de mayor exactitud.

Se definen las características de la señal de entrada de referencia con una frecuencia nominal de 60 Hz, una magnitud de 1 y un ángulo de fase de 0 radianes.

4.4.1 Cambio de magnitud en la entrada del PMU

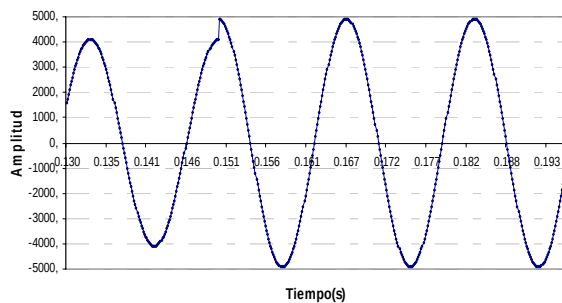
El microcontrolador ColdFire V2 M52233DEMO es de punto fijo [36], lo que quiere decir que no trabaja con flotantes; ya que le toma más tiempo realizar las operaciones. Es por ello que se trabaja en Q12, este es un factor de multiplicación,

por ejemplo; si la magnitud es de 1 unidad esta se multiplica por 2^{12} ; es decir la magnitud real es de 1 y la que utiliza el ColdFire es de 4096. Esto es muy importante ya que el cambio más pequeño del valor medido al cual responde el PMU es $1/4096$, es decir su resolución.

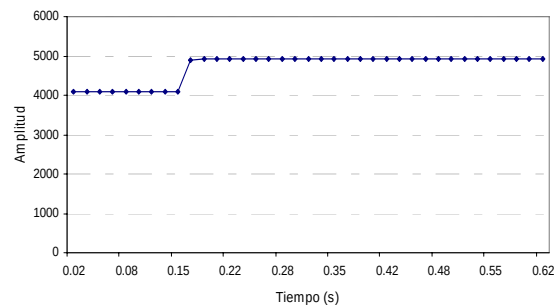
Para el nivel 0 se pide que la magnitud varíe del 80% al 120% de la señal de referencia indicad en el apartado 4.4. El 100% de la señal de referencia corresponde a 4096 por lo cual la magnitud variara de 3276.8 a 4915.2.

En la grafica 4.12 se muestra el fasor estimado cuando se tiene un cambio de magnitud del 20%. La frecuencia y el ángulo de fase son constantes al variar la magnitud de la señal de entrada, la magnitud del fasor si se ve afectada, se muestra su respuesta en la figura 2.12 b).

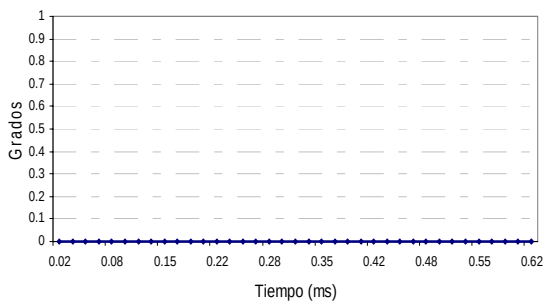
a) Señal de entrada.



b) Magnitud del fasor estimado.



c) Angulo del fasor estimado.



d) Frecuencia del fasor estimado.

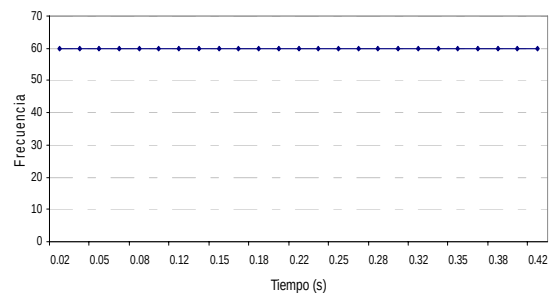


Figura 4.12.-Fasor cuando hay un cambio de magnitud.

Se realiza el cálculo del vector total de error con la ecuación 4.4. La magnitud varía cada 5% hasta llegar a los límites establecidos por el nivel 0. Los resultados del TVE se muestran en la tabla 4.9, donde el TVE máximo es de 0.024%.

Tabla 4.9.- TVE del cambio en magnitud para el nivel 0

	Valores PMU		Valores patrones		TVE (%)
	REAL	IMAGINARIO	Xr	Xi	
Magnitud 80%	3276,	0,	3276.8,	0,	0.024
Magnitud 85%	3481,	0,	3481.6,	0,	0.017
Magnitud 90%	3686,	0,	3686.4,	0,	0.011
Magnitud 95%	3891,	0,	3891.2,	0,	0.005
Magnitud 100%	4096,	0,	4096,	0,	0.000
Magnitud 105%	4300,	0,	4300.8,	0,	0.019
Magnitud 110%	4505,	0,	4505.6,	0,	0.013
Magnitud 115%	4710,	0,	4710.4,	0,	0.008
Magnitud 120%	4915,	0,	4915.2,	0,	0.004

El criterio de precisión de TVE detecta errores de la estimación de magnitud del fasor en la figura 4.13 se muestra la grafica del TVE vs el error en la magnitud, observando que 1% del de error en la Magnitud se obtiene 1% de error TVE, es decir se encuentra dentro del estándar.

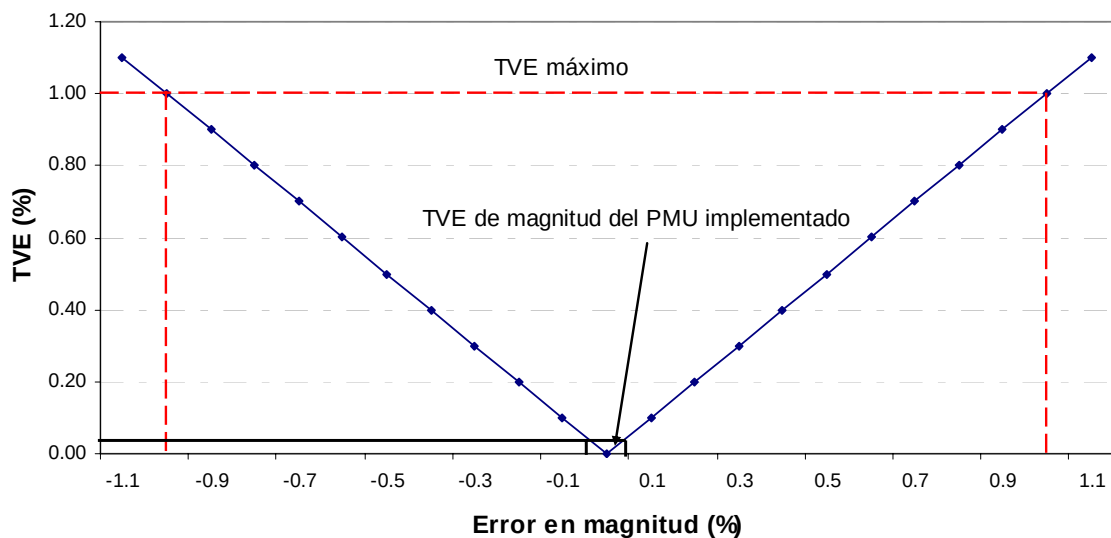


Figura 4.13.-TVE debido a la magnitud nivel 0.

En la figura 4.13 se muestra en líneas punteadas el área donde el TVE cumple con la norma, en línea continua se muestra el TVE máximo obtenido de las pruebas de cambio de magnitud realizadas al PMU implementado, observando que el TVE del PMU se encuentra dentro de la norma. Para el nivel 1 se pide que la magnitud varíe del 10% al 120% de la señal de referencia. La magnitud variara de 409.6 a 4915.2, los resultados se muestran en la tabla 4.10.

Al tener el 10% de la magnitud base se obtiene un TVE de 0.149%, aun así este cumple con la norma. La grafica de la figura 4.14 es del TVE vs el error en la magnitud, al igual que en el nivel 0 el TVE máximo calculado se encuentra dentro de los límites.

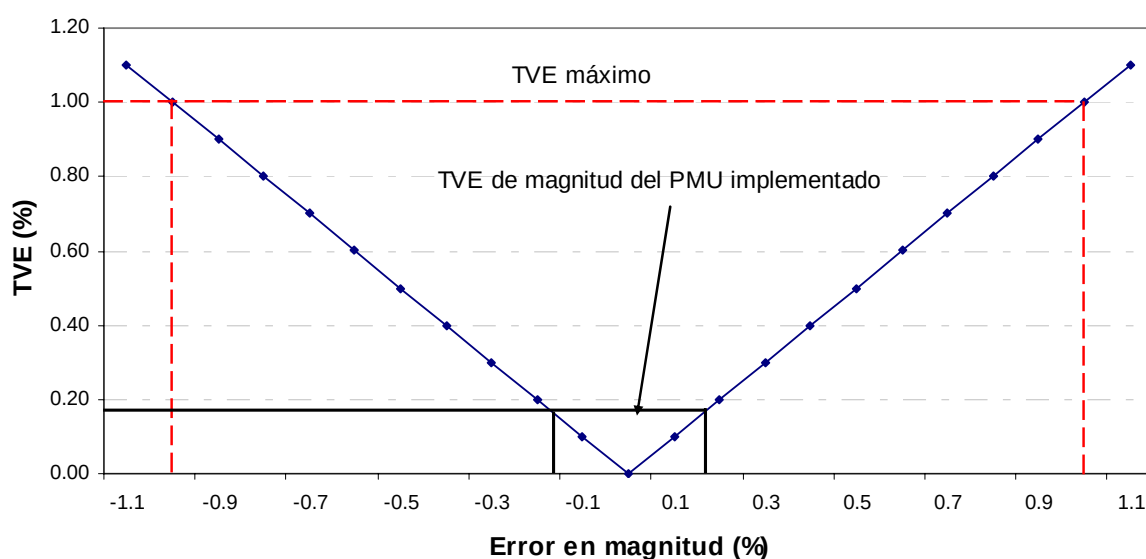


Figura 4.14.-TVE debido a la magnitud nivel 1.

Tabla 4.10.- TVE del cambio en magnitud para el nivel 1

	Valores PMU		Valores patrones		TVE (%)
	REAL	IMAGINARIO	Xr	Xi	
Magnitud 10%	409,	0,	409.6,	0,	0.146
Magnitud 15%	614,	0,	614.4,	0,	0.065
Magnitud 20%	819,	0,	819.2,	0,	0.024
Magnitud 25%	1023,	0,	1024,	0,	0.098
Magnitud 30%	1228,	0,	1228.8,	0,	0.065
Magnitud 35%	1433,	0,	1433.6,	0,	0.042
Magnitud 40%	1638,	0,	1638.4,	0,	0.024
Magnitud 45%	1843,	0,	1843.2,	0,	0.011
Magnitud 50%	2047,	0,	2048,	0,	0.049
Magnitud 55%	2252,	0,	2252.8,	0,	0.036
Magnitud 60%	2457,	0,	2457.6,	0,	0.024
Magnitud 65%	2662,	0,	2662.4,	0,	0.015
Magnitud 70%	2867,	0,	2867.2,	0,	0.007
Magnitud 75%	3072,	0,	3072,	0,	0.000
Magnitud 80%	3276,	0,	3276.8,	0,	0.024
Magnitud 85%	3481,	0,	3481.6,	0,	0.017
Magnitud 90%	3686,	0,	3686.4,	0,	0.011
Magnitud 95%	3891,	0,	3891.2,	0,	0.005
Magnitud 100%	4096,	0,	4096,	0,	0.000
Magnitud 105%	4300,	0,	4300.8,	0,	0.019
Magnitud 110%	4505,	0,	4505.6,	0,	0.013
Magnitud 115%	4710,	0,	4710.4,	0,	0.008
Magnitud 120%	4915,	0,	4915.2,	0,	0.004

4.4.2 Cambio del ángulo de fase.

En el nivel 0 se pide que el ángulo de fase varíe del π a $-\pi$ de la señal de referencia indicada en el apartado 4.4 Los resultados obtenidos se muestran en la tabla 4.11, el ángulo de fase varía cada $(1/4)$ de π hasta llegar a los límites establecidos, en este caso se tiene en mismo cambio para los niveles 0 y 1.

Tabla 4.11.- TVE del cambio ángulo en el nivel 0 y 1

	Valores PMU		Valores patrones		TVE (%)
	REAL	IMAGINARIO	Xr	Xi	
mas $\pi()$	-4096,	0,	-4096,	0,	0.000
mas 3/4 $\pi()$	-2896,	2896,	-2897,	2896,	0.024
mas 2/4 $\pi()$	0,	4096,	0,	4096,	0.000
mas 1/4 $\pi()$	2896,	2896,	2896,	2896,	0.000
menos 1/4 $\pi()$	2896,	-2896,	2896,	-2897,	0.024
menos 2/4 $\pi()$	0,	-4096,	0,	-4096,	0.000
menos 3/4 $\pi()$	-2896,	-2896,	-2897,	-2897,	0.035
menos $\pi()$	-4096,	0,	-4096,	-1,	0.024

Como se observa en la tabla 4.11 el TVE máximo es de 0.035%, el TVE no rebasa sus limites para ninguno de los dos niveles.

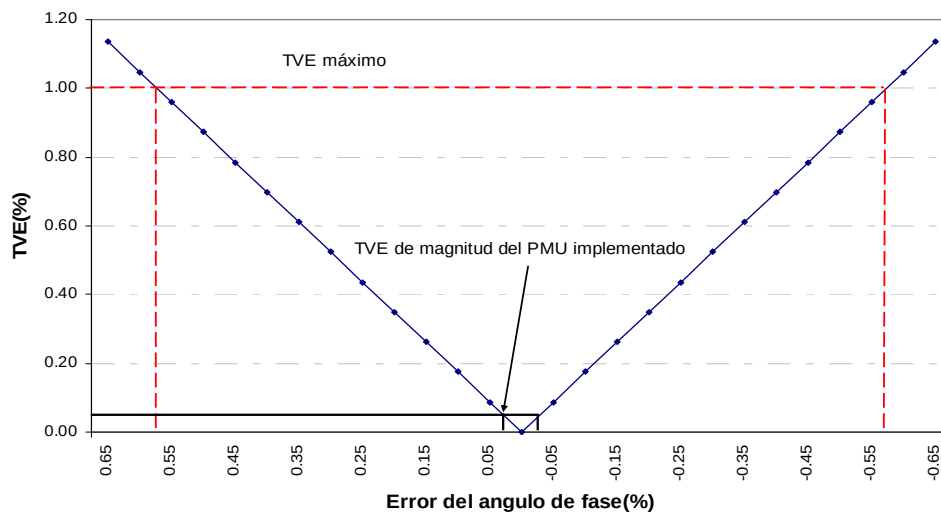


Figura 4.15.-Error debido al ángulo de fase.

El análisis del error en la estimación del ángulo de fase se muestra en la Figura 4.15. En este caso, el error se muestra en grados de ángulo de fase a 60 Hz. Aquí se obtiene un 1% de TVE cuando el error del ángulo de fase llega a $\pm 0,57^\circ$.

En la figura 4.15 se muestra en líneas punteadas el área donde el TVE cumple con la norma dando un intervalo de error en el ángulo de fase de 0.57° , en línea continua se muestra el TVE máximo obtenido de las pruebas realizadas al PMU implementado.

4.4.3 Distorsión armónica.

Este apartado es muy importante ya que se comprueban los filtros para armónicos pares e impares de la DFT. En el nivel 0 se requiere comprobar que el TVE es menor a 1 cuando se tiene una contaminación del 1% cualquier armónico hasta el cincuentavo.

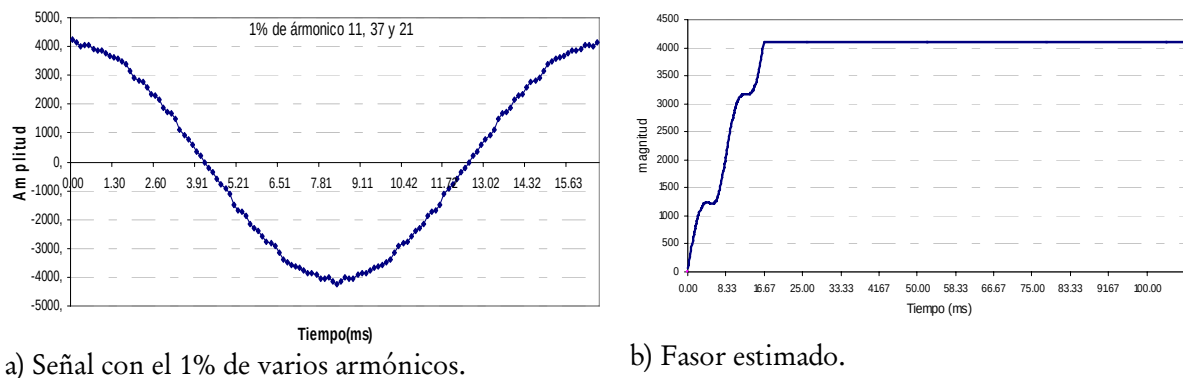


Figura 4.16.-Señal con 1% de cualquier armónico.

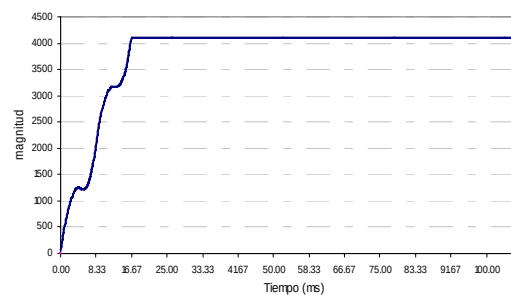
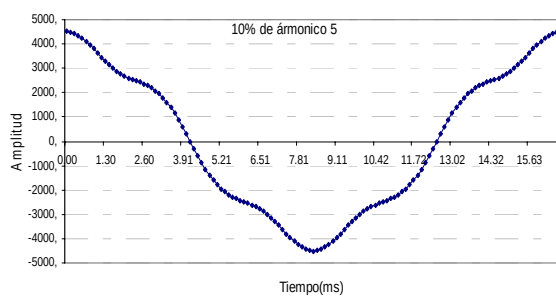
En la figura 4.16 b) se puede observar que el 1% del armónico 19 no afecta notoriamente a la señal, si se suma el 1% de varios armónicos se observa una mínima distorsión, la grafica se muestra en la figura 4.16 a) y en la grafica 4.16 b) se muestra su fasor.

En la tabla 4.12 se muestran los resultados cuando se mide una señal contaminada con el 1% desde el segundo armónico hasta el cincuentavo. Se muestran los resultados del TVE, su valor máximo es de 0.035% y se presenta cuando la señal esta contaminada por el tercer armónico, por lo tanto no rebasa el límite establecido en el estándar.

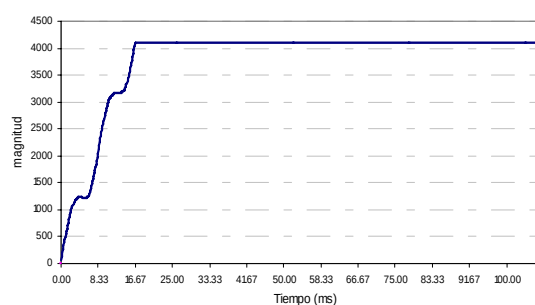
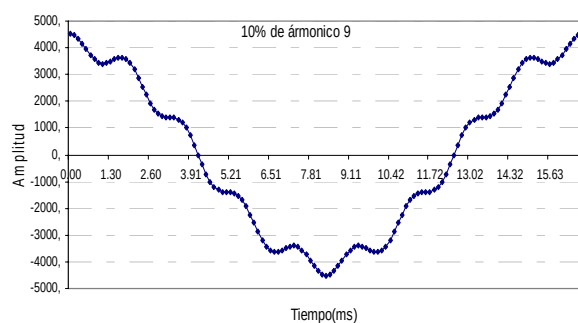
Tabla 4.12.- TVE de una señal con distorsión armónica del 1% para el nivel 0

Armonico	Valores PMU		Valores patrones		TVE (%)
	REAL	IMAGINARIO	Xr	Xi	
2,	4096,	0,	4096,	0,	0.000
3,	4095,	1,	4096,	0,	0.035
4,	4096,	0,	4096,	0,	0.000
5,	4096,	1,	4096,	0,	0.024
6,	4096,	0,	4096,	0,	0.000
7,	4095,	0,	4096,	0,	0.024
8,	4096,	0,	4096,	0,	0.000
9,	4096,	0,	4096,	0,	0.000
10,	4095,	0,	4096,	0,	0.024
11,	4095,	0,	4096,	0,	0.024
12,	4095,	0,	4096,	0,	0.024
13,	4095,	0,	4096,	0,	0.024
14,	4096,	0,	4096,	0,	0.000
15,	4096,	0,	4096,	0,	0.000
16,	4096,	0,	4096,	0,	0.000
17,	4096,	0,	4096,	0,	0.000
18,	4096,	0,	4096,	0,	0.000
19,	4096,	0,	4096,	0,	0.000
20,	4095,	0,	4096,	0,	0.024
21,	4096,	0,	4096,	0,	0.000
22,	4095,	0,	4096,	0,	0.024
23,	4095,	0,	4096,	0,	0.024
24,	4096,	0,	4096,	0,	0.000
25,	4095,	0,	4096,	0,	0.024
26,	4096,	0,	4096,	0,	0.000
27,	4096,	0,	4096,	0,	0.000
28,	4096,	0,	4096,	0,	0.000
29,	4096,	0,	4096,	0,	0.000
30,	4096,	0,	4096,	0,	0.000
31,	4096,	0,	4096,	0,	0.000
32,	4096,	0,	4096,	0,	0.000
33,	4096,	0,	4096,	0,	0.000
34,	4096,	0,	4096,	0,	0.000
35,	4096,	0,	4096,	0,	0.000
36,	4096,	0,	4096,	0,	0.000
37,	4096,	0,	4096,	0,	0.000
38,	4096,	0,	4096,	0,	0.000
39,	4095,	0,	4096,	0,	0.024
40,	4096,	0,	4096,	0,	0.000
41,	4096,	0,	4096,	0,	0.000
42,	4095,	0,	4096,	0,	0.024
43,	4096,	0,	4096,	0,	0.000
44,	4095,	0,	4096,	0,	0.024
45,	4095,	0,	4096,	0,	0.024
46,	4096,	0,	4096,	0,	0.000
47,	4095,	0,	4096,	0,	0.024
48,	4096,	0,	4096,	0,	0.000
49,	4096,	0,	4096,	0,	0.000
50,	4096,	0,	4096,	0,	0.000

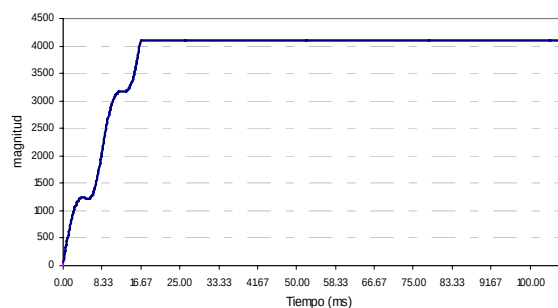
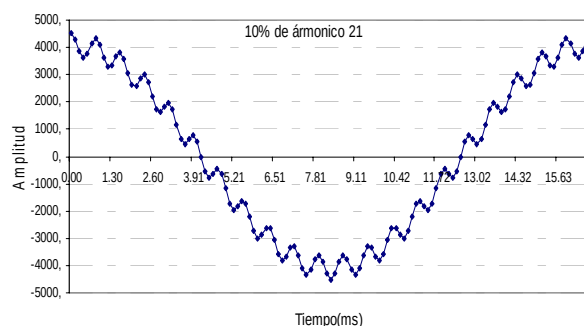
En el nivel uno se incrementa la distorsión es del 10%, el TVE debe ser menor a 1%. Este porcentaje de distorsión armónica es más visible que el caso anterior, en las figuras 4.17 se muestran señales contaminadas con el 10% de diferentes armónicos y la magnitud del fasor.



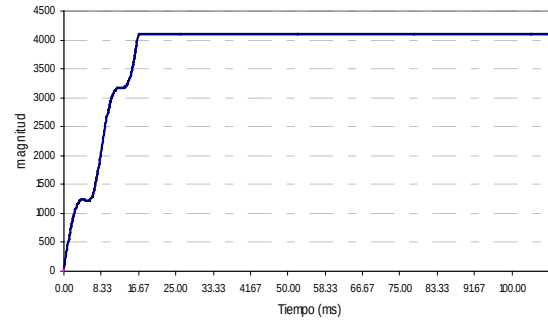
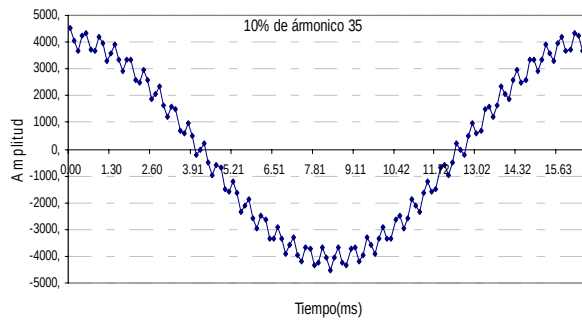
a) Señal a 60Hz con el 10% del 5^{to} armónico



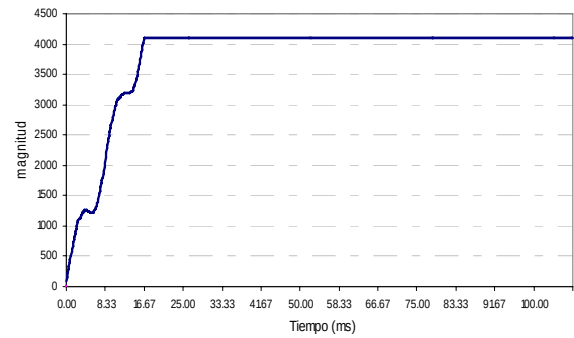
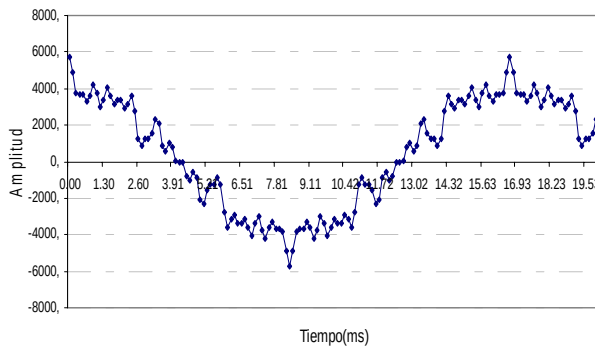
b) Señal a 60Hz con el 10% del 9^{no} armónico



c) Señal a 60Hz con el 10% del 21^{vo} armónico



d) Señal a 60Hz con el 10% del 21^{vo} armónico



e) Señal a 60Hz con el 10% de varios armónicos

Figura 4.17.-Señales con 10% de cualquier armónico y sus fasores.

Cuando existe el 10% de armónicos 5, 9, 21 y 35 (ver figura 4.17), la estimación de la magnitud del fador es constante.

En la tabla 4.13 se muestra el TVE para la distorsión armónica en el nivel 1, se realiza la prueba para cada uno de los armónicos del 2 al 50, la magnitud de estos armónicos es del 10% con respecto a la magnitud de la frecuencia fundamental.

Tabla 4.13.- TVE de una señal con distorsión armónica del 10% para el nivel 1

Armonico	Valores PMU		Valores patrones		TVE (%)
	REAL	IMAGINARIO	Xr	Xi	
2,	4096,	0,	4096,	0,	0.000
3,	4095,	0,	4096,	0,	0.024
4,	4095,	0,	4096,	0,	0.024
5,	4095,	0,	4096,	0,	0.024
6,	4096,	0,	4096,	0,	0.000
7,	4096,	0,	4096,	0,	0.000
8,	4095,	0,	4096,	0,	0.024
9,	4096,	0,	4096,	0,	0.000
10,	4095,	0,	4096,	0,	0.024
11,	4095,	0,	4096,	0,	0.024
12,	4096,	0,	4096,	0,	0.000
13,	4096,	0,	4096,	0,	0.000
14,	4095,	0,	4096,	0,	0.024
15,	4095,	0,	4096,	0,	0.024
16,	4096,	0,	4096,	0,	0.000
17,	4095,	0,	4096,	0,	0.024
18,	4095,	0,	4096,	0,	0.024
19,	4096,	0,	4096,	0,	0.000
20,	4096,	0,	4096,	0,	0.000
21,	4096,	0,	4096,	0,	0.000
22,	4096,	0,	4096,	0,	0.000
23,	4096,	0,	4096,	0,	0.000
24,	4095,	0,	4096,	0,	0.024
25,	4096,	0,	4096,	0,	0.000
26,	4095,	0,	4096,	0,	0.024
27,	4095,	0,	4096,	0,	0.024
28,	4095,	0,	4096,	0,	0.024
29,	4095,	0,	4096,	0,	0.024
30,	4096,	0,	4096,	0,	0.000
31,	4095,	0,	4096,	0,	0.024
32,	4095,	0,	4096,	0,	0.024
33,	4095,	0,	4096,	0,	0.024
34,	4095,	0,	4096,	0,	0.024
35,	4096,	0,	4096,	0,	0.000
36,	4095,	0,	4096,	0,	0.024
37,	4095,	0,	4096,	0,	0.024
38,	4095,	0,	4096,	0,	0.024
39,	4096,	0,	4096,	0,	0.000
40,	4095,	0,	4096,	0,	0.024
41,	4095,	0,	4096,	0,	0.024
42,	4096,	0,	4096,	0,	0.000
43,	4096,	0,	4096,	0,	0.000
44,	4096,	0,	4096,	0,	0.000
45,	4095,	0,	4096,	0,	0.024
46,	4095,	0,	4096,	0,	0.024
47,	4096,	0,	4096,	0,	0.000
48,	4096,	0,	4096,	0,	0.000
49,	4096,	0,	4096,	0,	0.000
50,	4095,	0,	4096,	0,	0.024

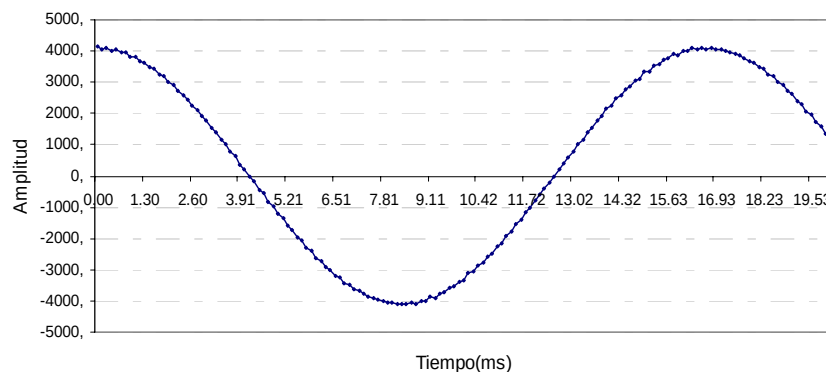
4.4.4 Señal de Interferencia.

La señal de interferencia es una señal fuera del banda con una frecuencia f_i , (Ec 4.5) [39].

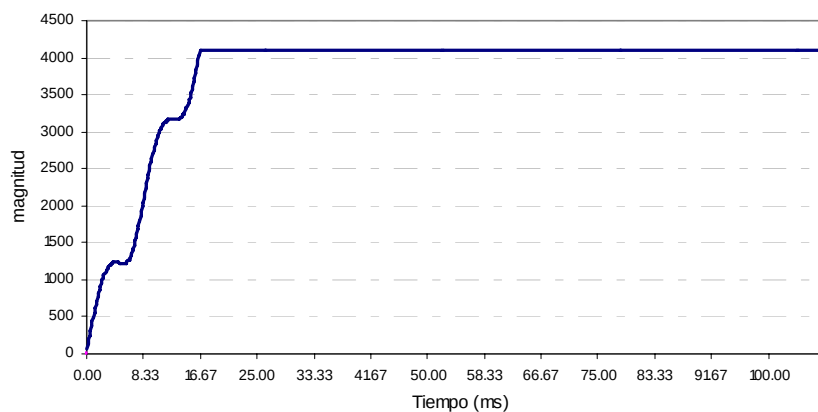
Para el nivel 0 se le agrega a la señal de entrada tipo sinusoidal pura el 1% de una señal de interferencia con una frecuencia $f_i=3940$, que cumple con la ecuación 4.5 se obtiene la magnitud del fasor mostrada en la figura 4.18 y en la tabla 4.14 se muestra el TVE obtenido.

$$|3940 - 60| > \frac{7680}{2}$$

$$3880 > 3840$$



Señal de entrada



Magnitud del fasor estimado

Figura 4.18.-Señales con 1% de una señal de interferencia.

Tabla 4.14.- TVE con 1% de una señal de interferencia para el nivel 0

Valores PMU		Valores patrones		TVE(%)
REAL	IMAGINARIO	Xr	Xi	
4096,	-1,	4096,	0,	0.024

La magnitud del fasor no se ve afectada por el 1% de la señal e interferencia y el TVE no rebasa sus límites como se observa en la tabla 4.14.

Para el nivel 1 se realiza la prueba con la misma frecuencia de la señal de interferencia pero ahora con el 10%, la señal se muestra en la figura 4.19 a) y la magnitud del fasor estimado en la figura 4.19 b).

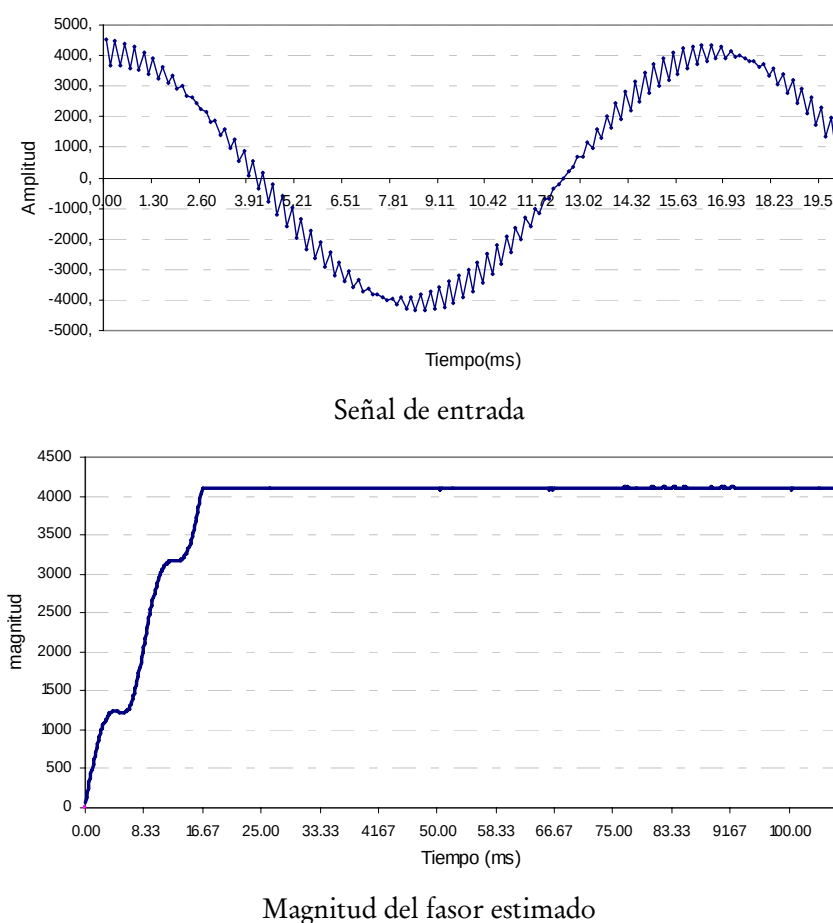


Figura 4.19.-Señales con 10% de una señal de interferencia.

Tabla 4.15.- TVE con 10%de una señal de interferencia para el nivel 1

Valores PMU		Valores patrones		TVE
REAL	IMAGINARIO	Xr	Xi	
4100,	0,	4096,	0,	0.00097656,

Aunque la señal de entrada se distorsiona notablemente con el 10% de la señal de interferencia a una frecuencia de 3940 Hz, la magnitud del fasor no se ve afectada por ella y el TVE no rebasa sus límites (tabla 4.15).

4.4.5 Cambios en la frecuencia.

En el apartado 4.2 se analiza la respuesta del PMU ante una señal diferente de la frecuencia nominal, en este apartado se mostrara el vector total de error para el nivel 0 y 1.

En el nivel 0 se calcula el TVE para un rango de frecuencias de $\pm 0.5\text{Hz}$ de la nominal, los resultados se muestran en la tabla 4.14.

Tabla 4.16.- TVE con 10%de una señal de interferencia para el nivel 1

	Valores PMU		Valores patrones		TVE (%)
	REAL	IMAGINARIO	Xr	Xi	
Frec_nom+ 0.5 Hz	4111,	-106,	4111,	-107,	0.024
Frec_nom + 0.4 Hz	4108,	-85,	4108,	-86,	0.024
Frec_nom + 0.3 Hz	4105,	-64,	4105,	-65,	0.024
Frec_nom + 0.2 Hz	4102,	-42,	4102,	-43,	0.024
Frec_nom + 0.1 Hz	4099,	-21,	4099,	-22,	0.024
Frec_nom- 0.1 Hz	4092,	21,	4092,	21,	0.000
Frec_nom - 0.2 Hz	4088,	42,	4088,	42,	0.000
Frec_nom - 0.3 Hz	4085,	65,	4085,	64,	0.024
Frec_nom - 0.4 Hz	4081,	86,	4081,	86,	0.000
Frec_nom - 0.5 Hz	4077,	107,	4077,	107,	0.000

En el nivel 1 se calcula el TVE para un rango de frecuencias de $\pm 5\text{Hz}$ de la nominal, los resultados se muestran en la tabla 4.17. Para el nivel 0 y 1 el TVE se obtiene dentro de los límites.

Tabla 4.17.- TVE con 10% de una señal de interferencia para el nivel 1

	Valores PMU		Valores patrones		TVE (%)
	REAL	IMAGINARIO	Xr	Xi	
Frecuencia + 5 Hz	4071,	-1006,	4071,	-1007,	0.024
Frecuencia + 4 Hz	4108,	-818,	4108,	-819,	0.024
Frecuencia + 3 Hz	4128,	-622,	4128,	-623,	0.024
Frecuencia + 2 Hz	4133,	-420,	4133,	-421,	0.024
Frecuencia +1 Hz	4122,	-212,	4122,	-213,	0.024
Frecuencia - 1 Hz	4054,	216,	4054,	216,	0.000
Frecuencia - 2 Hz	3998,	434,	3998,	434,	0.000
Frecuencia - 3 Hz	3927,	654,	3927,	654,	0.000
Frecuencia - 4 Hz	3843,	874,	3843,	874,	0.000
Frecuencia - 5 Hz	3746,	1093,	3745,	1093,	0.026

4.5 Comunicación.

El medidor fasorial sincronizado vía satélite, mediante el modulo receptor GPS, tiene la capacidad de medir los fasores de voltaje y corriente de un sistema trifásico de 60 Hz, los fasores calculados y el estampado de tiempo son enviados mediante el RS232.

Los PMUs están situados en las subestaciones en donde se supervisan voltaje, corriente, y frecuencia. Estas medidas se envían a un concentrador de datos, que correlaciona los datos mediante una etiqueta de tiempo para crear sistemas con una medida [48].

Para poder probar el protocolo de comunicación del PMU normalizado es necesario hacer uso del programa PMU Connection Tester que fue desarrollado por la autoridad del valle de Tennessee (TVA) para apoyar la iniciativa del sincrofasor.

Para poder instalar el programa es necesario contar con la versión 2.0 de .NET.

Este programa apoya actualmente los protocolos siguientes: La versión 1/Draft 7 de IEEE C37.118, el bosquejo 6 de IEEE C37.118, IEEE 1344 y BPA PDCstream [48].

Antes de realizar las pruebas del PMU normalizado es necesario configurar el programa PMU Connection Tester:

En los parámetros de la conexión se exhibe los detalles referentes a la interacción entre el probador de la conexión de PMU y el dispositivo probado, ver figura 4.20.

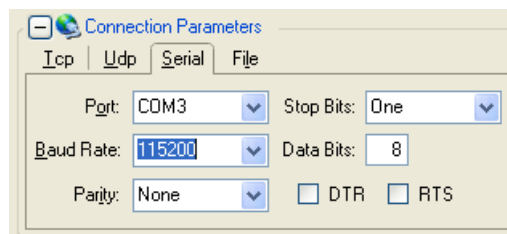


Figura 4.20.-Conexión de Parámetros.

- El puerto serial local a través del cual el probador de la conexión de PMU está recibiendo datos del dispositivo probado, se utiliza el COM1.
- Velocidad a utilizar durante la transmisión al dispositivo probado, de 115200 baudios para enviar 30 tramas por segundo.

- Paridad a utilizar durante la transmisión al dispositivo probado, en este caso no se utiliza paridad.
- El nombre del archivo que crea el probador que guarda el registro de secuencias de datos.

El probador de la conexión de PMU permite que el usuario guarde los últimos datos que fluyen del PMU si así lo requiere el usuario, no es indispensable llenar este campo.

Se selecciona el protocolo IEEE C37.118 (Versión 1/ Draft7 figura 4.21) el cual se utiliza para probar el PMU Normalizado, aunque es utilizado para probar otros protocolos.

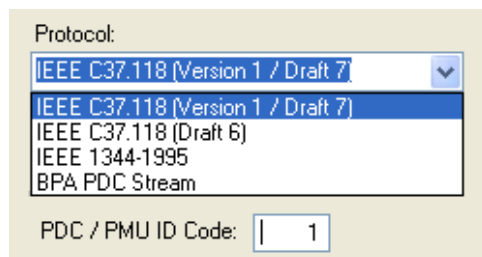


Figura 4.21.-Seleccionando Protocolo.

Código de identificación (ID) de PMU: en donde se especifica el código de la identificación necesitado para establecer una conexión, se muestra en la figura 4.22.

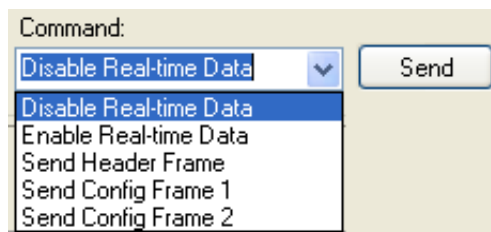


Figura 4.22.-Protocolo e ID del PMU.

Comando: Enumera los comandos disponibles para que el dispositivo probado responda lo que se esta pidiendo, como puede ser la configuración o el envío de datos.

La lista de comandos que puede ser transmitida se muestra en la figura 4.22.

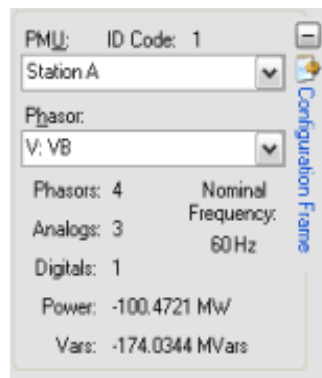


Figura 4.23.-Datos del PMU.

En la figura 4.23 se muestra el bloque de configuración donde se exhibe la información relevante de los elementos previamente configurados en el dispositivo conectado.

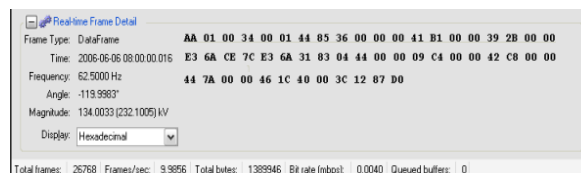


Figura 4.24.-Mediciones en tiempo real.

Sección Datos en Tiempo Real se exhibe los detalles sobre las medidas recibidas del dispositivo probado, como son tiempo recibido del GPS, frecuencia, ángulo, magnitud y forma de ver los datos (4.24).

CAPÍTULO 5.- Conclusiones y Recomendaciones para trabajos futuros.

Con los resultados al elaborar este trabajo, se derivan las siguientes conclusiones y recomendaciones para la elaboración de trabajos futuros.

5.1 Conclusiones.

El calculo de los fasores y frecuencia con el algoritmo de la DFT y FFT es correcta y precisa, pero existen diferencias muy significativas en el tiempo de ejecución, la DFT tarda 100 veces más tiempo en ejecutarse que la FFT para una resolución de una décima en la frecuencia, debido al número de operaciones que cada uno de estos algoritmos realiza.

Para la implementación de un medidor fasorial estandarizado no es conveniente utilizar la FFT ya que requiere de 100 ciclos para poder recalculer los fasores y la frecuencia del sistema. La DFT es el algoritmo más adecuado para el cálculo de fasores, ya que elimina ruido, armónicas y tarda un ciclo en actualizar las mediciones y cumple con la cláusula 5 del protocolo IEEE C37.118.

El PMU implementado en esta tesis es capaz de trabajar con circuitos trifásicos desbalanceados debido a que este calcula las componentes simétricas a

través de los fasores A, B y C medidos; teniendo así una amplia ventaja para utilizarlos en sistemas de distribución.

Las pruebas del vector de error total no rebasan sus límites de precisión establecidos y el PMU se comunica de acuerdo al protocolo. Finalmente después de realizar las pruebas se concluye que el PMU implementado cumple con el estándar IEEE C37.118. .

5.2 Recomendaciones para trabajos futuros.

Realizar pruebas de este prototipo en instalaciones de Comisión Federal de Electricidad (CFE).

Realizar la comunicación de la unidad de medición fasorial con el sistema de control a través de Ethernet.

Realizar diferentes aplicaciones con el sistema desarrollado.

Trasladar el proyecto a un microcontrolador o DSP de punto flotante.

Bibliografía.

- [1].S.S. Thakur, A.K. Sinha, "A Robust Dynamic State Estimator for Electric Power Systems", Technical Journals of Electric Engineering, vol. 84, No.8, Indian Institution of Engineering, Kolkata, Junio 2003.
- [2].A. P. Sakis Meliopoulos, F. Zhang, and S. Zelingher, "Hardware and Software Requirements for a Transmission System Harmonic Measurement System," Proceedings of the Fifth International Conference on Harmonics in Power Systems (ICHPS V), pp. 330-338, Atlanta, Septiembre 1992.
- [3].A. P. Sakis Meliopoulos, B. Fardanesh, S. Zelingher "Power System State Estimation: Modeling Error Effects and Impact on System Operation", IEEE Transactions Power App. Syst., vol. 11, No.2, pp.939-946, 1996.
- [4].Pablo Álvarez Danker, Desarrollo de un modelo computacional de. estimador de estado para sistemas eléctricos de potencia. Universidad de Chile, Facultad de Ciencias Físicas y Matemáticas Departamento de Ingeniería eléctrica.
- [5].Sunil Kabra "A Technique for Estimating System-Wide Phasors in Real Time" Thesis submitted to the Faculty of the Virginia Polytechnic Institute and State University in partial fulfillment of the requirements for the degree of Master of Science in Electrical Engineering.
- [6].Polo Francisco Padilla Monroy, "Desarrollo de un sistema de medición de variables Eléctricas para un Sistema de baja tensión tipo industrial" Tesis que para obtener el grado de Maestro en Ciencias 2006.
- [7].Edmund O. Schweitzer, III August 25, 2006. "Ten Synchrophasor application Tips" Schweitzer Engineering Laboratories, Inc.
- [8].Martin, K. E.; Benmouyal G.; Adamiak, M. G.; Begovic, M.; Burnett, R.O., Jr.; Carr, K. R.; Cobb, A. Kusters, J. A.; Horowitz, S. H.; Jensen, G. R.; Michel, G.

- L.; Murphy, R.J.; Phadke, A. G.; Sachdev, M.S.; Thorp, J.S. IEEE standard for synchrophasors for power systems. IEEE Transactions on Power Delivery, 1998.
- [9]. Phadke, A.G. Synchronized phasor measurements-a historical overview. Transmission and Distribution Conference and Exhibition 2002: Asia Pacific. IEEE/PES, 2002.
- [10]. Phadke, A.G. Synchronized phasor measurements in power systems. Computer Applications in Power, IEEE, April 1993.
- [11]. Burnett, R.O., Jr.; Butts, M.M.; Cease, T.W.; Centeno, V.; Michel, G.; Murphy, R.J.; Phadke, A.G. Synchronized phasor measurements of a power system event. Power Systems, IEEE Transactions on, Aug. 1994.
- [12]. Reglamento de la Ley Federal del Servicio Público de Energía Eléctrica. CFE Reglamentos.
- [13]. Zivanovic, R.; Cairns, C. Implementation of PMU Technology in State Estimation: An overview. IEEE African 4th. 1996.
- [14]. Liu C.W.; Thorp J. Application of synchronized phasor measurements to real-time transient stability prediction. Generation, Transmission and Distribution, IEEE Proceedings. 1995.
- [15]. Ota, Yutaka; Ukai Hiroyuki; Nakamura, Koichi; Fujita, Hideki. Evaluation of Stability and Electrical Power Quality in Power System by using phasor measurements. IEEE Power System Technology, 2000.
- [16]. Cortés, R.; Libreros, D.; San Vicente, R.; Robles, J.. Construcción de un sincrofasor de bajo costo basado en un DSP-Controlador. Third LACCEI International Latin American and Caribbean Conference for Engineering and Technology (LACCET'2005). Cartagena de Indias, Colombia June 2005.
- [17]. San Vicente R.; Cortes, R.; Robles J.; Chong-Quero, J.E. Synchrophasor design based on a DSP-microcontroller. 2005 IEEE International Workshop on Intelligent Signal Processing. Sept 2005.

- [18]. San Vicente R.; Cortes, R.; Robles J.; Chong-Quero, J.E. DSP-microcontroller implementations of a simplified algorithm for synchrophasor calculation. 2005 2nd International Conference on Electrical and Electronics Engineering. Sept 2005.
- [19]. Wilson, Robert E. PMU's Satellite synchronized measurements confirms power equation. IEEE Potentials. 1994.
- [20]. Carlos Cuvas Castillo, "Implementación de un medidor fasorial" Tesis que para obtener el grado de Maestro en Ciencias 2006.
- [21]. Joseph Schmuller, "Aprendiendo UML en 24 horas" Pretice Hall
- [22]. ColdFire® Embedded Controllers, MCF5223x Family, Fact Sheet
- [23]. David G. Hart, David Uy, Vasudev Gharpure, Damir Novosel, Daniel Karlsson, Unidades PMU Supervisión de las redes eléctricas: un nuevo enfoque.
- [24]. A. G. Phadke, J. S. Thorp. Synchronized Phasor Measurements and Their Applications. Springer 2008.
- [25]. Rafael quintanilla adánez, "sistemas de transporte inteligentes medición sincronizada de fasores", La Red Eléctrica del Futuro y la Generación Distribuida. Bilbao, 27-28 de octubre de 2005.
- [26]. Xu Bei, Yeo Jun Yoon and Ali Abur, Texas A&M University College Station, Texas, U.S.A. abur@ee.tamu.edu "Optimal placement and utilization of phasor measurements for state estimation"
- [27]. V. Centeno, et al: An adaptive out-of-step relay. IEEE Trans. on Power Delivery, vol 12, no 1, January 1997, 61–67.
- [28]. O. Faucon, L. Dousset: Coordinated defense plan protects against transient instabilities. IEEE Computer Applications in Power, vol 10, no 3, July 97, 22–26.
- [29]. Steven W. Smith, "The Scientist and Engineer's Guide to Digital Signal Processing", Second Edition. www.DSPguide.com

- [30]. Dr. David Sebastian Baltazar “Algoritmos no recursivos, Algoritmos Trigonométricos”.
- [31]. Dr. David Sebastian Baltazar “Algoritmos no recursivos, Algoritmos de mínimos errores cuadrados”.
- [32]. Dr. David Sebastian Baltazar “Algoritmos no recursivos, Algoritmos de Transformada Discreta de Fourier”.
- [33]. Benmouyal, Gabriel; Schweitzer, E. O.; Guzman, A. Synchronized phasor measurement in protective relays for protection, control, and analysis of electric power system. 29th annual western protective relay conference Spokane, Washington October 22-24, 2002.
- [34]. Oppenheim, Alan V.;Willsky, Alan V. Sistemas y Señales. Prentice Hall. México.
- [35]. <http://www.selinc.com>
- [36]. Freescale Semiconductors Data Sheet: MCF52235 ColdFire microcontroller. Document number: MCF52235DS Rev 3, 02/2007.
- [37]. Daniel Torres, Hands on Workshop: Dynamic Webpage Server with the MCF56223x Family. Freescale Semiconductors.
- [38]. Phadke, A. G.; Thorp, J. S.; Adamiak, M. G. A new measurement technique for tracking voltage phasors, local system frequency, and rate of change frequency. IEEE Transactions on Power Apparatus and Systems, 1983.
- [39]. Trimble. GPS Boards & Chipsets Home, Embedded: Portable Devices <http://www.trimble.com/>
- [40]. C.L. Forescue, “Method of Symmetrical Coordinates Applied to the Solution of Polyphase” Networks, Trans. AIEE, vol.37, 1918, págs. 1 027 - 1 140.
- [41]. J. J. Grainger, W. D. Stevenson Jr. “Análisis de sistemas de potencia” Mc Graw Hill 1996. pags 391 – 396.

- [42]. Valdano, Jonathan W. Excerpts from Introduction to Embedded Microcomputer Systems: Motorola 6811 and 6812c Simulation 2001.
- [43]. IEEE Standard for Synchrophasors for Power Systems IEEE Power Engineering Society, Sponsored by the Power System Relaying Committee. Ken Martin, Dan Hamai, 2006.
- [44]. Jun Jin Kong and Keshab K. Parhi “Interleaved Cyclic Redundancy Check (CRC) Code” Department of Electrical and Computer Engineering, University of Minnesota Minneapolis, MN 55455, USA.
- [45]. H. Michael Ji, and Earl Killian “Fast Parallel CRC Algorithm and Implementation on a Configurable Processor” Tensilica, Inc. 3255-6 Scott Blvd Santa Clara, CA 95054.
- [46]. Phadke, A. G.; Martin, K. IEEE standard for synchrophasors for power systems. IEEE Std 1344-1995.
- [47]. 1999, Raúl Borjas M. / Nelson Quispe. “Instrumentación Virtual Para Sistema de Medición y Análisis de la Red Eléctrica” Revista de la Facultad de Ingeniería, vol. 6. Universidad de Tarapacá Arica, Chile pp.45-54.
- [48]. Eastern Interconnect Phasor Project (EIPP) Tools. TVA PMU Connection Tester (Version 2.2.0.24862, posted May 2, 2007).
- [49]. O. Faucon, L. Dousset: Coordinated defense plan protects against transient instabilities. IEEE Computer Applications in Power, vol 10, no 3, July 1997, 22–26.
- [50]. Raúl Alarcón, “Diseño orientado a objetos con UML” Grupo Eidos.
- [51]. Stephen J. Mellor, “Embedded System in UML”
- [52]. Figliola, Beasley, “Mediciones mecánicas teoría y diseño”, Alfaomega
- [53]. Cooper, William D.; Helfrick, Albert D. Instrumentación electrónica moderna y técnicas de medición. Prentice Hall.

APÉNDICE A.-UML (Lenguaje unificado de modelado)

Permite a los creadores de sistemas generar diseños que capturen sus ideas en una forma convencional y fácil de comprender para comunicarlas a otras personas [21], [50], [51]. Cuando los sistemas son de alta complejidad, la clave esta en organizar el proceso de diseño de tal forma que los analistas, desarrolladores y otras personas involucradas en el desarrollo del sistema lo comprendan y el UML proporciona tal organización [21].

Un arquitecto no podría crear una compleja estructura como lo es un edificio de oficinas sin crear primero un anteproyecto detallado, es decir; así como el arquitecto muestra un anteproyecto, el programador deberá mostrar un plan de diseño [21], [50].

A.1 Diagramas UML

El UML esta compuesto por diversos elementos gráficos que se combinan para conformar diagramas y cuenta con reglas para combinar tales elementos ya que UML es un lenguaje. Los diagramas UML permiten examinar un sistema desde distintos puntos de vista. En un modelo UML no es importante que aparezcan todos los diagramas. Pero ¿Por qué es necesario contar con diferentes perspectivas de un sistema? Por lo general un sistema cuenta con varias personas implicadas las cuales tienen enfoques particulares en diversos aspectos del sistema [21], [52].

El escrupuloso diseño del sistema involucra todas las posibles perspectivas, y el diagrama UML le da una forma de incorporar una perspectiva en particular. El objetivo es satisfacer a cada persona implicada.

El desarrollo de un sistema es una actividad humana. Sin un sistema de notación fácil de comprender, el proceso de desarrollo tiene una cantidad enorme de errores. El UML es un sistema de notación que se ha convertido en estándar en el mundo de desarrollo de sistemas, hecho por Grady Booch, James Rumbaugh e Iván Jacobson [21, 50,52].

El UML esta constituido por una serie de diagramas y proporciona un estándar que permite al analista del sistema generar un anteproyecto de varias facetas que sean comprensibles para los programadores y los que están involucrados en el proceso de desarrollo. Un modelo UML que es lo que supuestamente hará el sistema, más no como lo hará [21].

En el mundo del software el encapsulamiento permite reducir el potencial de errores que pudiera ocurrir. En un sistema que consta de objetos, estos dependen unos de otros en diversas formas. Si uno de ellos falla y uno de los especialistas del software tienen que modificarlo de cierta forma, el ocultar sus operaciones de otros objetos significara que tal vez no será necesario modificar los demás objetos [21].

A.2 Diagramas de Clase.

Muchas de las cosas que nos rodean tienen atributos (propiedades) y realizan determinadas acciones. Se podría imaginar cada una de estas acciones como un conjunto de tareas.

Un rectángulo es el símbolo que representa a la clase y se divide en 3 áreas. El área superior contiene el nombre, el área central contiene los atributos y el área inferior las acciones. Un diagrama de clase esta formado por varios rectángulos de este tipo conectados por líneas que muestran la manera en que las clases se relacionan entre si.

Los diagramas de clase facilitan las representaciones a partir de las cuales los desarrolladores podrían trabajar, como también colaboran en lo referente al análisis [52].

A.3 Diagrama de objetos.

Un objeto es una instancia de clase. La figura siguiente muestra la forma en que el UML representa la forma de un objeto. El nombre de la instancia específica se encuentra a la izquierda de los dos puntos y el nombre de la clase a la derecha [50, 51, 52].

A.4 Diagramas de caso de uso

Un caso de uso es una descripción de las acciones del un sistema desde el punto de vista del usuario. Para los desarrolladores del sistema es una técnica de aciertos y errores para obtener los requerimientos del sistema desde el punto de vista del usuario. En la figura siguiente se presenta un caso de uso en UML [21].

A.5 Diagrama de secuencias.

Los diagramas de fase y los de objetos representan información estática. No obstante en un sistema funcional los objetos interactúan entre si, y tales interacciones suceden con el tiempo. El diagrama de secuencias UML muestra la mecánica de la interacción con base en tiempos.

Teniendo como ejemplo una lavadora; entre los componentes de la lavadora se encuentran: una manguera de agua (para obtener agua), un tambor (donde se coloca la ropa) y un sistema de drenaje. De igual manera estos también son objetos (como vera un objeto puede estar conformado por varios objetos).

¿Qué sucederá cuando invoque al caso uso lavar ropa? Si damos por hecho que se completaron las operaciones “Agregar ropa”, “agregar detergente” y “activar” [21, 50, 52], la secuencia seria de la siguiente manera:

- El agua empieza a llenar el tambor mediante una manguera.
- El tambor permanece inactivo durante unos 5 minutos.
- La manguera dejara de abastecer el agua.
- El tambor girara de un lado a otro durante 15 minutos.
- El agua jabonosa saldrá por el drenaje.
- Comenzara nuevamente el abastecimiento del agua.
- El tambor continuara girando.
- El abastecimiento de agua se detendrá.
- El agua del enjuague saldrá para el drenaje.
- El tambor girara en un sentido y se incrementara su velocidad.
- El tambor dejara de girar y el proceso habrá terminado.

La figura siguiente presenta un diagrama de secuencias que captura las interacciones que se realizan a través del tiempo entre el abastecimiento de agua, el tambor y el drenaje [21, 50, 52] (representados como rectángulos en la parte superior del diagrama). En este diagrama el tiempo se da de arriba hacia abajo.

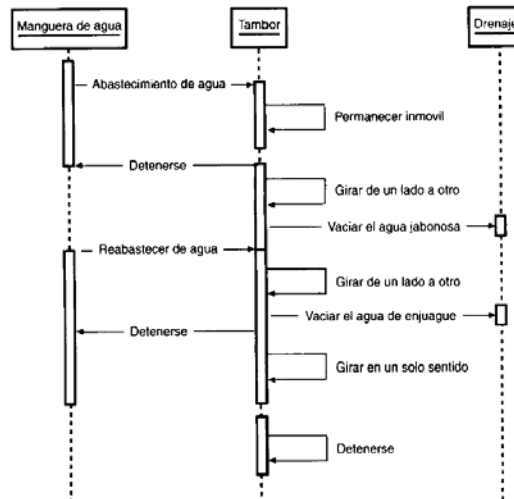


Figura A.1.-Diagrama de secuencias UML

A.6 Diagrama de actividades.

Las actividades que ocurren dentro de un caso de uso o dentro del comportamiento de un objeto se dan, normalmente, en secuencia, como en los once pasos de la sección anterior. La figura siguiente muestra la forma en que el diagrama de actividades UML representa los pasos del 4 al 6 de tal secuencia [21, 50, 52].

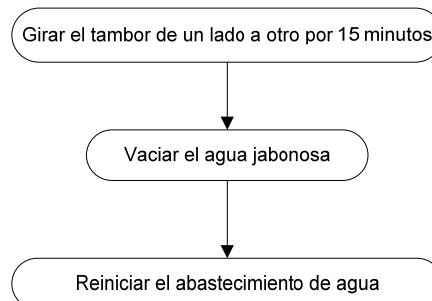


Figura A.2.-Diagrama de actividades UML.

A.7 Paquetes.

A veces se necesita organizar los elementos de un diagrama en un grupo. Tal vez para mostrar que ciertas clases o componentes son parte de un subsistema en particular, para ello los agrupamos en un paquete, que se representara por una carpeta tabular, como se muestra en la siguiente figura [21, 51].

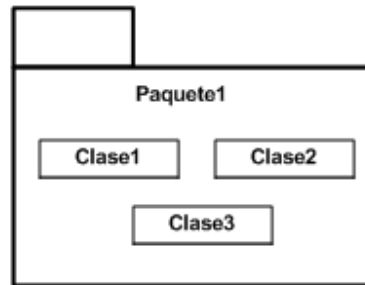


Figura A.3.-El paquete UML le permite agrupar los elementos de un diagrama.

Es frecuente que alguna parte del diagrama no presente una clara explicación del porque esta allí o la manera en que trabaja. Cuando este sea el caso la nota UML será útil. La nota es un rectángulo con una esquina doblada, dentro del rectángulo se coloca una explicación [21]. Se adjunta la nota al elemento del diagrama conectándolos mediante una línea discontinua.

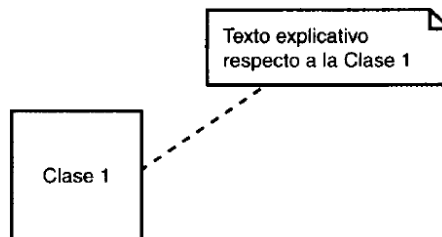


Figura A.4.- Notas en UML

En cualquier diagrama, podrá agregar comentarios aclaratorios mediante una nota.

A.8 Envío de Mensajes.

En un sistema los objetos trabajan en conjunto. Esto se logra mediante el envío de mensajes entre ellos. Un objeto envía un mensaje a otro para realizar una operación, y el objeto receptor realizará la operación.

Asociación.- Los objetos se relacionan entre si de alguna forma. Hay asociaciones de una sola dirección y otras de doble dirección [21, 50, 52].

APÉNDICE B.- Hardware

B.1 Introducción

En este apartado se presenta el hardware utilizado para la implementación del medidor fasorial síncrono estandarizado. Los principales elementos utilizados se enlistan a continuación:

- 68K ColdFire microprocesador M52233DEMO.
- Receptor GPS.
- Adecuadores de señales.

B.2 68K ColdFire microprocesador M52233DEMO.

Esta sección proporciona una visión general de las principales características y componentes funcionales del microcontrolador CF52233 de la familia de Coldfire.



Figura B.1.-Tarjeta de desarrollo M52233DEMO

El MCF52233 representa una familia de integrados de 32 bits basados en microcontroladores Coldfire V2 de micro arquitectura. Estos dispositivos son ideales para aplicaciones que requieren un control significativo de la conectividad, memoria de datos, y la interfaz de usuario, así como el procesamiento de señales. La familia

MCF5223x de dispositivos Coldfire son de un solo chip que proporcionan soluciones de control de 32 bits con una interfaz Ethernet. En la tabla B.1 se muestran las características básicas con las que cuenta el microprocesador ColFire V2 [53] utilizado en esta aplicación.

Tabla B.1.- Características básicas del microcontrolador.

Flash/SRAM	Características claves	Velocidad
256 KB/32 KB	➤ Coldfire V2 básicos con una mejora de la unidad multiplica-acumula (EMAC). ➤ Aceleración de la Unidad de cifrado (CAU). ➤ Fast Ethernet Controller (FEC) con el chip transceptor (ePHY). ➤ Tres receptores universales asíncronos / transmisores (UARTs). ➤ Módulo controlador de red de área 2.0B (FlexCAN). ➤ Interamericano de circuito integrado (I2C) controlador de bus. ➤ Convertidor analógica-digital (ADC) de 10 o 12 bits. ➤ Módulo de interfaz periférica serial (QSPI). ➤ Cuatro canales, 32-bit acceso directo a memoria (DMA). ➤ Cuatro canales, 32-bit de captura de entrada / salida de temporizadores para comparar con el DMA de apoyo. ➤ Dos temporizadores de interrupción periódica de 16-bit (PITs). ➤ Software programable del temporizador de vigilancia. ➤ Dos controladores de interrupción, cada uno de ellos capaz de manejar hasta 63 fuentes de interrupción.	60 MHZ

El chip del microprocesador cuenta con 80 pines configurados como se muestra en la figura B.2.

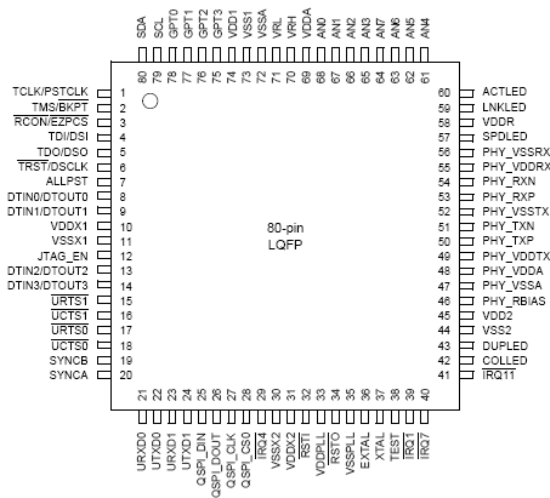


Figura B.2.- Microprocesador de 80 pines

B.2.1 Diagrama a Bloques.

El controlador embebido MCF5223x proporciona un conjunto de periféricos, la memoria de un tamaño compacto y plataforma Ethernet. En la figura B.3 y B.4 se muestra el diagrama a bloques del MCF52233.

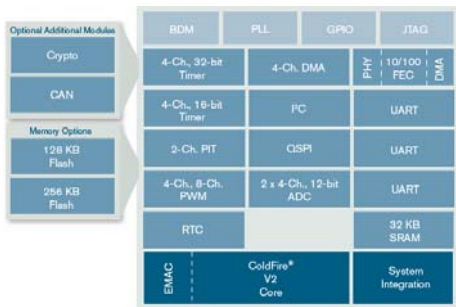


Figura B.3.-Diagrama a bloques del MCF5223x.

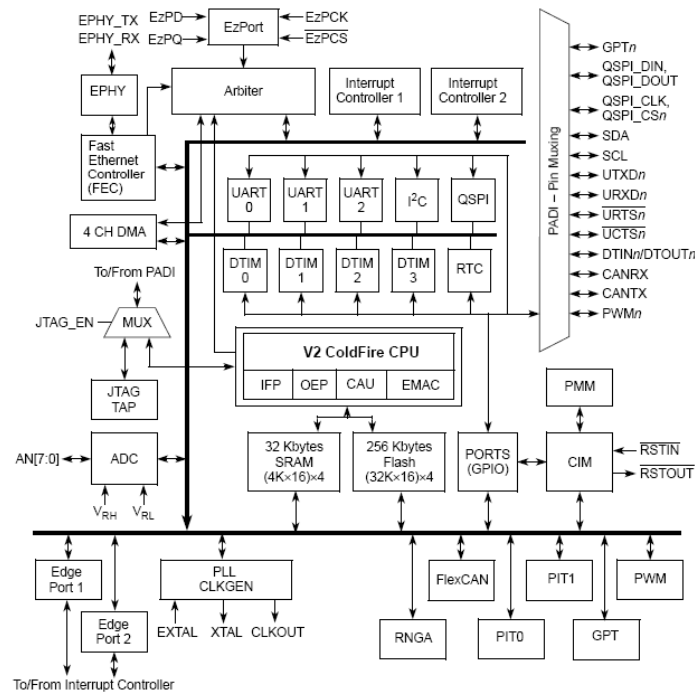


Figura B.4.-Diagrama a bloques del MCF52233.

Es importante mencionar que el MC52233 cuenta con un módulo flash Coldfire (CFM) el cual es una memoria no volátil (NVM) del módulo que se conecta al transformador de alta velocidad de los buses locales. El CFM está construido con cuatro bancos matrices flash de $32\text{ K} \times 16\text{-bits}$ para generar 256 Kbytes de 32-bits de memoria flash. Estos conjuntos son eléctricamente borrable y programable, no volátil de datos de programa y memoria. La memoria flash es ideal para el programa y el almacenamiento de datos, que permite la reprogramación. La memoria flash también puede ser programado a través de la EzPort, que es una interfaz serie flash de programación que permite que el flash pueda ser leído, borrado y programado por un controlador externo en un formato compatible con la mayoría de bus SPI de chips de memoria flash. Este dispositivo permite la fácil programación automatizada a través de equipo de prueba o de distribución masiva de herramientas de programación [23].

B.2.1.1 ADC.

El ADC consta de ocho canales de entrada con una resolución de 12 bits. Este almacena los resultados en un accesible búfer para su posterior procesamiento. Puede ser configurado para realizar una única exploración y detenerse, realizar un análisis cada vez activado, o realizar una exploración en secuencia programada varias veces hasta que se detiene manualmente.

El ADC puede ser configurado en forma secuencial o simultánea de conversión. Cuando se configura para forma de conversión secuencial, hasta ocho canales pueden ser incluidos en la muestra y se almacena en el orden especificado por el registro de lista de canales. Interrupciones opcionales se pueden generar al final de la secuencia de exploración si un canal está fuera del rango (por debajo de la medidas de bajo umbral o límite por encima del umbral de alto límite establecido en el límite de registros) o en diferentes condiciones de cruce por cero. Entre sus principales características del ADC se encuentran [23]:

- Resolución de 12 bits.
- La Frecuencia de reloj ADC máxima es 5MHz con periodo de 200ns.
- Rango de muestreo arriba de 1.66 millones de muestras por segundo.
- Tiempo de conversión sencilla de 8.5 ciclos de reloj del ADC ($8.5 \times 200\text{ns} = 1.7\mu\text{s}$).
- Tiempo de conversión adicional 6 ciclos de reloj del ADC ($6 \times 200\text{ns} = 1.2\mu\text{s}$).
- Ocho conversiones en 26.5 ciclos de reloj del ADC ($26.5 \times 200\text{ns} = 5.3\mu\text{s}$) usando modo simultaneo.
- Las conversiones del ADC pueden ser sincronizadas por el PWM y reloj.
- Muestreo simultáneo o secuencial.
- Capacidad para muestreo simultáneo y retención de dos entradas.
- Capacidad para escanear secuencial y almacenamiento de ocho mediciones.
- Multiplexado interno para seleccionar dos de ocho entradas.

- Interrupción al finalizar de escanear, si un límite de fuera de rango es excedido.
- Inclusión de offset, resultados signados y no signados.

Cada una de las conversiones puede ser seleccionada de uno de los ocho canales, teniendo cuidado de no muestrear el mismo canal. Los resultados de las conversiones son guardadas, dos al mismo tiempo, en pares consecutivos de registros de resultados. La rutina ADC es una interrupción del convertidor analógico digital, en esta interrupción se realizan las conversiones de analógico a digital las señales de corriente y voltaje. Al digitalizar dichas señales se determinan los fasores con el cálculo de la DFT.

Para este proyecto el ADC es configurado para una conversión simultánea, ya que se requiere obtener la lectura de 6 canales correspondientes a las corrientes y voltajes de las tres fases del sistema eléctrico de potencia.

B.2.1.2 Temporizador.

Hay cuatro temporizadores independientes de 32 bits (DTIM0, DTIM1, DTIM2, y DTIM3) en cada tarjeta de desarrollo. Cada módulo temporizador incorpora un registro de 32-bit establecido para la configuración y control.

El tiempo puede ser configurado para operar desde el reloj del sistema o de un reloj externo si el reloj del sistema está seleccionado, puede ser dividida por 16 o 1, este es el preescalador. Para este proyecto el temporizador es utilizado para generar el periodo de muestreo para el ADC, en este caso de 0.130208 ms, para capturar 128 muestras por ciclo.

B.2.1.3 Pines de entrada y salida de propósito general (GPIO)

Casi todos los pines en el MCF52233 son propósito general de E / S, además de su principal funciones y se agrupan en puertos de 8 bits. Algunos puertos no utilizar todos los 8 bits. Cada puerto tiene registros para configurar, supervisar y controlar.

Estos pines permiten comunicarse con hardware externo, mediante la lectura y escritura a los pines, y pueden ser utilizados para identificar una interrupción externa. Para este proyecto se ocupa uno de estos pines para recibir el pulso por segundo proveniente del receptor GPS.

B.2.1.4 Interfaz de comunicación serial.

La interfaz de comunicación serial permite enviar y recibir información entre el microcontrolador y un dispositivo remoto, incluyendo otros microcontroladores, un dispositivo GPS o una computadora. Consta de un transmisor y un receptor que opera de manera independiente, pero usa una misma velocidad de comunicación, el microcontrolador utilizado cuenta con tres puertos seriales.

Para la implementación del PMU se utiliza dos puertos de intercomunicación serial, el UART0 para la intercomunicación con el receptor GPS, con una velocidad de 4800 baudios, 8 bits, sin bit de paridad y un bit de paro, esta configuración es requerida por el GPS. El segundo puerto es el UART1, este se utiliza para la intercomunicación con el Maestro a una velocidad de 115200 baudios para poder realizar las actualizaciones de fasores 30 veces por segundo, 8 bits, sin bit de parada y un bit de paro.

B.3 Receptor GPS

La sincronización de fasores se realiza mediante la utilización de un GPS, para que el equipo PMU sincronice la medición de los fasores [47].

El sistema de satélites de posicionamiento global (GPS), consta de 24 satélites situados en seis órbitas, a unos 16.000 km de altura sobre la superficie de nuestro planeta. Es decir, están situados aproximadamente a la mitad de la altura que correspondería a una órbita geoestacionaria. El posicionamiento del plano orbital y la posición de los satélites en las órbitas es tal que en un instante dado se pueden ver al menos cuatro satélites desde cualquier punto de la superficie de la Tierra. Frecuentemente son visibles más de seis satélites [24]. El canal de uso civil del sistema GPS transmite las coordenadas posicionales de los satélites desde los que es posible determinar la ubicación de una estación receptora en tierra. Además, los satélites transmiten una señal de un impulso por segundo, junto con un identificador de la señal que puede ser interpretado por los receptores de la estación terrestre.

La transmisión de uso civil, de la señal de tiempo tiene una precisión de $1\mu s$ pero en la práctica a menudo se considera que es mucho más precisa. El impulso de tiempo tiene una importancia fundamental para la aplicación que aquí se considera. El formato exacto del etiquetado de tiempo se define en la norma IEEE 1344 [30].

Es importante mencionar que el sector de la energía utiliza actualmente un estándar de tiempo, conocido como estándar IRIG-B, para los registradores digitales de fallos con etiquetado de tiempo y para otros sistemas de supervisión de eventos en las subestaciones. No obstante, con los receptores según estándar IRIG-B la precisión de la sincronización es del orden de 1 milisegundo, lo cual no es suficiente para

obtener mediciones precisas del sistema de distribución de energía [53]. El receptor de la marca Trimble está integrado por las siguientes partes [20]:

- Antena GPS.
- Módulo Receptor.

B.3.1 Antena del GPS

La antena recibe los datos satelitales y los envía al módulo receptor. Debido a que las señales GPS se propagan a una frecuencia de 1575 MHz y no penetran a través de materiales opacos, la antena debe estar localizada al aire libre y con clara visión hacia el cielo. La antena Trimble incluye un pre-amplificador con filtros, para enviar una señal depurada al módulo receptor [39]. La figura B.5 muestra la antena utilizada en esta implementación. La antena es del tipo magnética para poder colocarla en una superficie metálica que de fijación a la misma. Este tipo de antena es del tipo magnética para poder colocarla en una superficie metálica que de fijación a la misma [39].

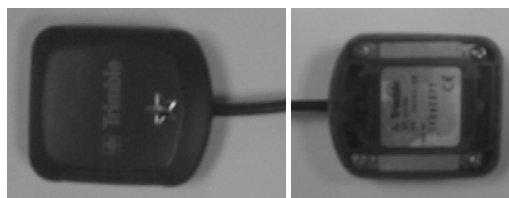


Figura B.5.- Antena del GPS

B.3.2 Módulo receptor.

En la figura A.6 se muestra el módulo receptor GPS. La parte principal del modulo receptor es el conector J4, en el cual se encuentran integradas las funciones de entrada/salida, además de la alimentación de el modulo [20].



Figura B.6 Módulo Receptor GPS

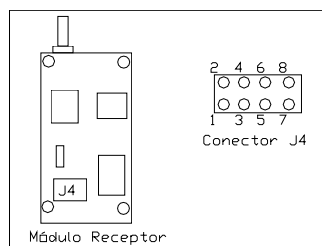


Figura B.7 Numeración de los pines del conector J4.

En la figura B.7 se muestra la forma de numeración de los pines de acuerdo a la forma de sostener el Módulo Receptor. Los pines de conexión se muestran en la tabla B.2:

Tabla B.2 descripción del conector j4.

Pin #	Función	Descripción
1	TXD 2	Transmisión del puerto 2, CMOS/TTL
2	Alimentación Primaria	5VDC $\pm$ 5%, 95 mA
3	TXD 1	Transmisión del puerto 1, CMOS/TTL
4	Alimentación de respaldo	+3.2VDC to +5.25VDC, 4 μ A
5	RXD 1	Recepción del puerto 1, CMOS/TTL
6	1 PPS	Pulso por segundo, CMOS/TTL
7	RXD 2	Recepción del puerto 2, CMOS/TTL
8	Tierra	Tierra

En la tabla B.2 se puede notar que el módulo receptor cuenta con dos pines de alimentación de energía, de las cuales se habla a continuación. El módulo requiere en su alimentación primaria de +5 volts DC $\pm 5\%$ at 95 mA, excluyendo la antena. Una importante consideración de diseño para la alimentación es el reloj interno del módulo receptor, el cual trabaja a una frecuencia de 12.504 MHz ± 3 KHz [35].

B.3.3 Pulso Por Segundo.

La señal de 1PPS es una señal TTL generada por el sistema GPS receptor. Por omisión cada vez que se energiza el receptor, se genera el PPS, aunque no se haya detectado satélite alguno, una vez que se detecta por lo menos un satélite, el receptor sincroniza el flanco de subida del PPS con la recepción del UTC [42]. En el pin 6 del conector J4 del GPS de la marca trimble se transmite la señal de un pulso por segundo (pps) con una anchura de 10 microsegundos, compatible con CMOS/TTL. Este pulso es usado una vez por segundo con el flanco de subida del pulso sincronizado con el UTC del sistema satelital [39].

Para cualquier aplicación en donde se utilice el GPS para sincronizar, se deberá de usar el flanco de subida del 1PPS ya que este se sincroniza a partir de la recepción del UTC.

B.4 Adecuador de señal

Los adecuadores de señal están diseñados para que el máximo voltaje de entrada al ADC del microcontrolador DSP sea de 3.3 volts. El ADC solo recibe valores en el rango de 0 a 3.3 volts por lo que es necesario superponer un voltaje de CD para que la onda senoidal se encuentre entre los rangos tanto en su pico positivo como su pico negativo.

Para poder monitorear el voltaje, corriente, frecuencia y ángulos de fase del sistema eléctrico de potencia, con los puertos ADC del microcontrolador, se utiliza la tarjeta CTPE-V4.

B.4.1 Especificaciones Eléctricas.

- Voltaje de alimentación: 12 V.
- Rango de Medición de Voltaje: 30 a 150 V.
- Contacto de Relevadores: 5A, 250 V.
- Transformadores de corriente a voltaje, (5 Amp $\approx$ 5 VCA).

B.4.2 Acondicionamiento de señales de voltaje de CA.

La medición de voltaje de CA se realiza por medio de un circuito basado en la topología de divisor de voltaje (figura B.8), el cual mide en un rango de 0 - 300 V CA y da una medición proporcional dentro del rango de 0 - 3.3V CD, el cual estará montado sobre una señal de 1.65 VCD. Para analizar el circuito se utiliza el teorema de superposición de fuentes de voltaje.

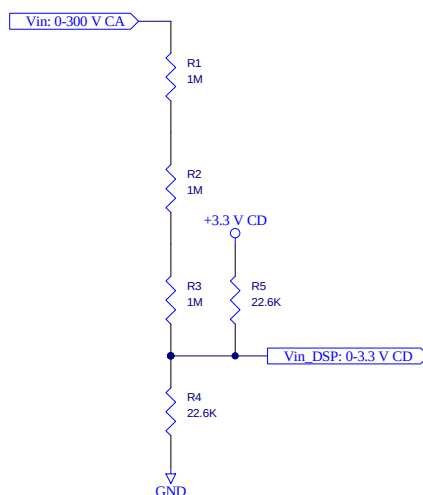


Figura B.8.- Circuito acondicionador de señal de VAC

B.4.2.1 Análisis en CA

Por el teorema de superposición de fuentes de voltaje, primero se analiza la red en CA, por lo que la fuente de voltaje de 3.3 V CD se comporta como un corto circuito, al forzar el voltaje en sus terminales a 0V, quedando el resistor R5 conectado a tierra y a Vin_DSP; la red se simplifica sumando los resistores en serie R1, R2, R3 (RS) y el paralelo que se forma con las resistores R4 y R5, obteniendo el circuito equivalente de dos resistores en serie. Considerando un voltaje de entrada Vin=300 VCA se calcula el voltaje pico para el semiciclo positivo:

$$Vin_{Pico} = Vin \times \sqrt{2} \quad (B.1)$$

$$Vin_{Pico_Positivo} = 300V \times \sqrt{2} = 424.26 \quad (B.2)$$

Y de igual manera para el semiciclo negativo.

$$Vin_{Pico_Negativo} = -300V \times \sqrt{2} = -424.26 \quad (B.3)$$

Por el método de divisor de voltaje, se obtiene Vin_DSPPico_Positivo.

$$Vin_DSP_{Pico_Positivo} = \frac{424.26V \times 11.3K\Omega}{(3M + 11.3K)\Omega} = \frac{4794183.976V}{3011300} = 1.5921V \quad (B.4)$$

Ahora, Vin_DSPPico_Negativo.

$$Vin_DSP_{Pico_Negativo} = \frac{-424.26 \times 11.3K\Omega}{(3M + 11.3K)\Omega} = \frac{-4794183.976V}{3011300} = -1.5921V \quad (B.5)$$

B.4.2.2 Análisis en CD:

Procediendo con el análisis por el teorema de superposición, se analiza la red de la figura B.9 en CD; en la cual, la fuente de voltaje Vin se comporta como un corto

circuito, al forzar el voltaje en sus terminales a 0V; la red se simplifica al sumar los resistores en serie R1, R2, R3 (RS) y el arreglo en paralelo de RS || R4.

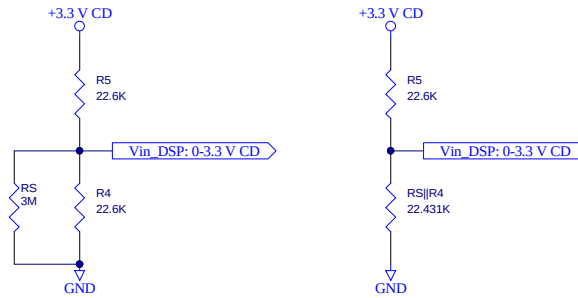


Figura B.9.- Circuito resultante con fuente de CA en corto

Se calcula por divisor de voltaje, el voltaje en V_{in_DSP} :

$$V_{in_DSP_{CD}} = \frac{3.3V \times 22.431K\Omega}{(22.6K + 22.431K)\Omega} = \frac{74022.3V}{45031.020} = 1.6438V \quad (B.6)$$

Por lo tanto, para el semiciclo positivo:

$$V_{in_DSP} = V_{in_DSP_{Pico_Positivo}} + V_{in_DSP_{CD}} = 1.592 + 1.6438 = 3.2359V \quad (B.7)$$

Y para el negativo:

$$V_{in_DSP} = V_{in_DSP_{Pico_Negativo}} + V_{in_DSP_{CD}} = -1.592 + 1.6438 = 0.0517V \quad (B.8)$$

Lo cual está dentro de los niveles permitidos de voltajes de entrada del puerto ADC, los cuales son de 0 a 3.3 VCD.

B.5.3 Acondicionamiento de señales de corriente alterna.

La medición de intensidad de corriente se lleva a cabo por medio de tres transformadores de corriente a voltaje, los cuales entregan 5 VCA cuando circula una corriente de 5 Amp, dicho voltaje pasa por una etapa de acondicionamiento por medio de un divisor de voltaje.

Para analizar el circuito se utiliza el teorema de superposición de fuentes de voltaje. En la figura B.10 se muestra el circuito del cual se parte para el análisis.

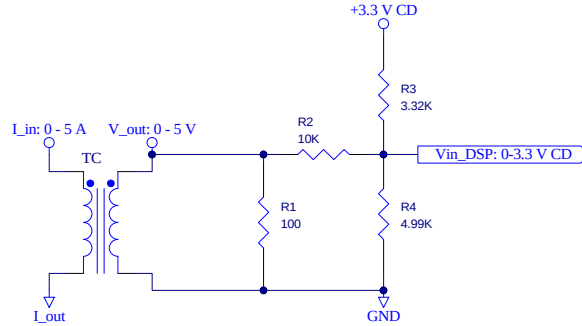


Figura B.10.- Circuito acondicionador de señal de I AC

B.5.3.1 Análisis en CA:

Primero se analiza la red en CA, por lo que la fuente de voltaje de 3.3 V CD se comporta como un corto circuito, al forzar el voltaje en sus terminales a 0V; el resistor R3 queda en paralelo a R4, simplificando de esta manera el circuito al resolver el arreglo en paralelo; ver figura B.11.

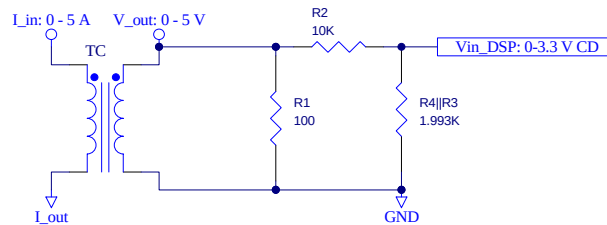


Figura B.11.- Circuito resultante con fuente de CD en corto.

Considerando una corriente de 5A tendremos un $V_{out} = 5 \text{ VCA}$ por lo que el voltaje pico $V_{outPico}$ es:

$$V_{outPico} = V_{out} \times \sqrt{2} \quad (B.9)$$

Para el semiciclo positivo se tiene

$$V_{_out_{Pico_Positivo}} = 5V \times \sqrt{2} = 7.071 \quad (B.10)$$

De igual manera el voltaje pico para el ciclo negativo es:

$$V_{_out_{Pico_Negativo}} = -5V \times \sqrt{2} = -7.071 \quad (B.11)$$

y por divisor de voltaje se calcula $Vin_DSP_{Pico_Positivo}$.

$$Vin_DSP_{Pico_Positivo} = \frac{7.071V \times 1.993K\Omega}{(10 + 1.993K)\Omega} = \frac{14096.8V}{11993.5} = 1.175V \quad (B.12)$$

Ahora se calcula $Vin_DSP_{Pico_Negativo}$.

$$Vin_DSP_{Pico_Negativo} = \frac{-7.071V \times 1.993K\Omega}{(10 + 1.993K)\Omega} = \frac{-14096.8V}{11993.5} = -1.175V \quad (B.13)$$

B.5.3.2 Análisis en CD.

Procediendo con el análisis por el teorema de superposición, se analiza la red de la figura 3.15 en CD; la fuente de voltaje $V_{_out}$ se comporta como un corto circuito, al forzar el voltaje en sus terminales a 0V, eliminando así el efecto del resistor R1. Se simplifica el circuito calculando el arreglo en paralelo $R2 || R4$; el circuito resultante se muestra en la figura B.12.

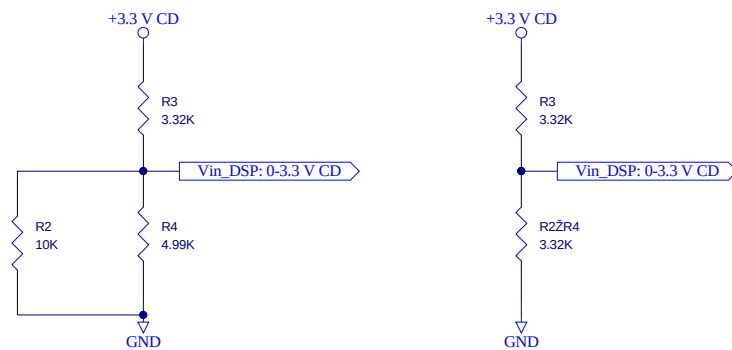


Figura B.12.- Circuito resultante con fuente de CA en corto.

Se calcula el voltaje Vin_DSP por divisor de voltaje obteniendo lo siguiente.

$$V_{in_DSP_{CD}} = \frac{3.3V \times 3.32K\Omega}{(3.32K + 3.32K)\Omega} = \frac{10.956V}{6.64} = 1.65V \quad (B.14)$$

Por lo tanto se tiene para el semiciclo positivo:

$$V_{in_DSP} = V_{in_DSP_{Pico_Positivo}} + V_{in_DSP_{CD}} = 1.175 + 1.65 = 2.825V \quad (B.15)$$

Y para el semiciclo negativo:

$$V_{in_DSP} = V_{in_DSP_{Pico_Negativo}} + V_{in_DSP_{CD}} = -1.175 + 1.65 = 0.475V \quad (B.16)$$

Lo cual está dentro de los niveles permitidos de voltajes de entrada del puerto ADC, los cuales son de 0 a 3.3 VCD.

APÉNDICE C.- Estándar IEEE C37.118

C.1 Introducción

El estándar del síncrofasor ha sido completamente revisado. El estándar original fue IEEE Std 1344-1995, que fue reafirmado en el 2001. Un trabajo grupal fue establecido en enero de 2001 para actualizar el estándar. Este nuevo estándar del síncrofasor IEEE Std C37.118, remplazo el original [43].

La medición, protección y sistemas de control basada en la computadora digital han llegado a ser características comunes de las subestaciones eléctricas de potencia. Este sistema usa muestreo de datos para calcular varias cantidades, tal como el fasor de voltaje y corriente. Los fasores en muchas funciones de adquisición de datos y protección, y su utilidad se incrementara en un futuro por referenciar a una base de tiempo común. Esto puede ser logrado para sincronizar la señal de entrada procesada en varios sitios de medición. Un grupo de mediciones simultaneas se obtienen de fasores sincronizados que proporcionan un método enormemente mejorado para los fenómenos dinámicos del sistema de potencia para mejorar el monitoreo, protección y control del sistema de potencia [43].

En este estándar se proporcionan definiciones claras de los fasores y síncrofasores. El concepto del error total del vector y pruebas completas se han estado introduciendo. Los formatos del mensaje se han estado actualizando del estándar original para mejorar el cambio de información con otros sistemas, tal como una estación maestro. Específicamente los campos de la sincronización, tamaño de la trama y la identificación de la estación han sido agregados a la trama de

datos, trama de configuración, trama de encabezado y trama de comandos. El CRC-CCITT es el que valida y checa la palabra [43].

C.1.2 Descripción

Esta norma cubre mediciones fasoriales sincronizadas que se usan en los sistemas de energía eléctrica. Define la medición, proporciona un método de cuantificación de las mediciones y calidad de pruebas especificadas. También define formatos para la transmisión de datos en tiempo real. Cláusula 1 establece el alcance y necesidades de la norma. Cláusula 2 referencias de otras normas que están relacionadas o pueden ser útiles en el estudio y la aplicación de esta norma. Cláusula 3 define los términos y siglas que se encuentran en esta norma. Cláusula 4 define la medición. En la clausula 5 define requisitos de medición, el método de cuantificación de la medición, un método de prueba, y límites de precisión. Cláusula 6 define comunicación de protocolo en tiempo real y formatos de mensaje [43].

C.1.3 Alcance

Este es un estándar para sistemas de medición de fasores sincronizados en los sistemas de potencia. Se ocupa de la definición de un fasor sincronizado, sincronización de tiempo, la aplicación de estampado de tiempo, el método verificar de medición para el cumplimiento de la norma, y los formatos de mensaje para la comunicación con una unidad de medición fasorial (PMU). En este contexto, un PMU puede ser una unidad física independiente, o una unidad funcional dentro de otra unidad física. Esta norma no especifica límites al tiempo de respuesta de la

medición, precisión bajo condiciones transitorias, hardware, software, o un método para el cálculo de fasor [43].

C.1.4 Propósito

Esta norma define mediciones de fasores sincronizados en subestaciones a fin de que los equipos de medida puedan ser fácilmente conectados con los sistemas asociados. Especifica los formatos de datos y requisitos de sincronización que permitan correlacionar fasores de diversas fuentes y los compara con datos similares de los distintos sistemas de medición [43].

C.2 Necesidades para este estándar

Los desarrollos recientes en el campo de adquisición de datos en sistema de potencia de facilitar a los usuarios la capacidad de poder calcular las cantidades fasoriales de sistema en tiempo real. Diferentes enfoques de hardware y software se están llevando a cabo para grabar, transmitir, y analizar estas mediciones, tanto en tiempo real y después del evento. Se requiere una norma que para la integración de sistemas de medición de energía en entornos de sistema, para especificar los formatos de la salida de datos, y para asegurar que la medición de los procesos está produciendo resultados comparables. El estándar del síncrofasor ayudará a garantizar los máximos beneficios de las mediciones fasoriales y permitirá el intercambio de datos entre una amplia variedad de sistemas para los usuarios en tiempo real y mediciones de fasores fuera de línea [43].

C.3 Definiciones:

A los efectos del presente documento, los siguientes términos y definiciones se aplican. El Diccionario de Autoridades de la norma IEEE Términos [B4] debe tener una referencia de los términos no definidos en el presente [43].

Anti-alias: El proceso de filtrado de una señal para eliminar los componentes de esa señal cuya frecuencia es igual o superior a la mitad la tasa de Nyquist (velocidad de muestreo). Si no se elimina, estos componentes de la señal parecen como un componente de menor frecuencia (un alias) [43].

IEEE punto flotante: Una representación de 32 bits de un número real, de conformidad con la norma IEEE Std 754 [43].

Tasa de Nyquist: Una tasa que es el doble de la frecuencia más alta en la señal de entrada analógica. La señal analógica debe ser muestreada a una tasa mayor que la tasa de Nyquist esta representada con precisión en forma digital [43].

Fasor: Un complejo equivalente de una simple onda coseno tales cantidades complejas son la magnitud que es la amplitud de la onda coseno y la ángulo (en forma polar) es el ángulo de fase de la onda coseno [43].

Sincronismo: El estado donde conectan los sistemas de corriente alterna, máquinas, o una combinación, operan a la misma frecuencia y en donde el ángulo de fase se desplaza entre tensiones constantes o varían de una constante y valor medio estable.

Fasor sincronizado: un fasor calculado a partir de datos de muestras utilizando un tiempo estándar, como señal de referencia para la medición. Fasor sincronizado desde sitios remotos tienen una relación de fase definida común [43].

C.4.- Medición de sincrofasor

La forma de onda sinusoidal pura $x(t) = x_m \cos(\omega t + \theta)$ es comúnmente representada como un fasor $X = X_r + jX_i = (X_m/\sqrt{2})(e^{j\varphi})$, donde φ depende de la definición de la escala de tiempo. Para este estándar, este concepto básico es adaptado como la representación de una señal sinusoidal de sistemas de potencia.

La representación fasorial de una senoide es independiente de su frecuencia. Una senoide $x(t) = X_m \cos(2\pi f t + \varphi)$, tiene un fasor representado como $\mathbf{X} = (X_m/\sqrt{2})(e^{j\varphi})$. El ángulo de fase φ del fasor es determinado por el comienzo de una señal sinusoidal en el tiempo ($t=0$) [43].

C.4.1.- Medición para el estampado de tiempo para el sincrofasor.

La medición de sincrofasor debe ser etiquetado con el tiempo correspondiente del UTC a el tiempo de medición. Este debe consistir de tres números: conteo del segundo del siglo (SOC), un conteo de fracción de segundo, y un estado del valor de tiempo. El contador SOC deberá ser de 4 bytes binarios contado en segundos desde la media noche del primero de enero de 1970, para el actual segundo. Deberán agregarse o quitar segundos como sea necesario del conteo para mantener la sincronización con el UTC. Este estampado de tiempo SOC es el mismo como es el sistema de computo de UNIX y es similar el uso para otros sistemas de computo incluyendo DOS, MAC OS y redes [networking time protocol (NTP)] [43].

C 4.2.- Sistema de tiempo de sincronización

La medición del síncrofasor debe ser sincronizado por el tiempo del UTC con una precisión suficiente. Note que un error de tiempo de $1\mu s$ corresponde a un error de fase de 0.022° para un sistema de 60 Hz y 0.018° para un sistema de 50 Hz. Un error de fase de 0.01 radianes o 0.57° causa un TVE de 1% como es definido en la ecuación 2. Este corresponde a un máximo error de tiempo de $\pm 26\mu s$ para un sistema de 60 Hz y $\pm 31\mu s$ para un sistema de 50 Hz [43].

El sistema debe ser capaz de recibir el tiempo de una fuente de alta confiabilidad, Tal como el sistema de posicionamiento global (GPS), que puede proporcionar una precisión suficiente en el tiempo para mantener el TVE dentro de los límites requeridos y provee indicaciones de la pérdida de sincronización. La bandera de información de datos (palabra STAT Bit 13) es proporcionada para indicar que una pérdida del tiempo de sincronización deberá ser afirmada cuando la pérdida de sincronización podría causar que el TVE exceda el límite o dentro de 1 mínimo de la actual pérdida de sincronización. La bandera deberá quedarse fijada hasta que los datos de adquisición sea re sincronizada para el requerimiento del nivel de precisión [43].

C.5.- Requerimientos de las mediciones del síncrofasor y el cumplimiento de la verificación

C.5.1.- Estimación del síncrofasor

La estimación del síncrofasor debe ser hecha y transmitida a una razón F_s que es un número entero de segundos por trama como lo especifica la variable DATA_RATE en la configuración de trama (ver tabla C.1). Esta estimación deberá

ser igualmente espaciados dentro de un intervalo de 1s. La trama de datos puede incluir múltiples canales de la estimación de fasores, palabras analógicas y digitales, palabras combinadas con un solo estampado de tiempo [43].

C.5.1.1 Reportando las tramas

El PMU deberá soportar reportar los datos un submúltiplo de la frecuencia nominal. Se requiere una tasa para sistemas de 50Hz y otra para 60Hz, son listadas en la tabla C.1 [43].

Tabla C.1.- Requerimientos del PMU para reportar tramas

Frecuencia del sistema	50 Hz		60Hz				
Tramas por segundo	10	25	10	12	15	20	30

C.5.1.2 Reportando tiempos.

Para un índice de reporte de N tramas por segundo, El reporte de tiempo debe ser uniformemente espaciado a través de cada segundo con trama número 0 coincidente con el PPS proveniente del GPS. Este reporte de tiempo o estampado de tiempo son usados para determinar los valores instantáneos del síncrofasor [43].

C.5.1.3.- Respuesta en el tiempo del PMU

La respuesta del tiempo de PMU será medido aplicando $\pm 10\%$ de la magnitud de la señal de entrada de la magnitud nominal y la razón de frecuencia. La respuesta en el tiempo es el intervalo de tiempo entre el instante del paso de cambio es aplicado y el estampado de tiempo de la primer medición del fasor para que el TVE permanezca dentro de sus límites de precisión [43].

C.5.2.- Límites de precisión

Bajo las condiciones donde X_m , ω , y φ son fijados, y para las condiciones mostradas en la tabla 4.8 (Capítulo 4), el TVE da limitaciones en la tabla 4.8 para dar el nivel de cumplimiento. El TVE es definido como se muestra en la ecuación (4.4 Capítulo 4) [43].

Cuando la frecuencia de la señal de entrada es exactamente igual a la nominal, El fasor será invariante en el tiempo y sus coordenadas serán fijadas y determinadas por la señal de fase. Cuando la señal de entrada no es exactamente igual a la tasa de frecuencia, el fasor será sometido a una rotación en el plano complejo. El TVE es la diferencia de la magnitud del vector entre el fasor teórico como se define en la ecuación (1.1) y el fasor estimado será dado por la medición del dispositivo, Se expresara como una fracción de la magnitud del fasor teórico como se muestra en la ecuación (4.4) [43].

C.5.3.- Cumplimiento de verificación

El cumplimiento de las pruebas deberá ser regular para comparar la estimación del fasor obtenido bajo las condiciones de estado estable a el correspondiente valor teórico de X_r y X_i y calcular el TVE, como se define en la ecuación (4.4). Las condiciones de estado estable son donde X_m , ω y φ de la señal de prueba son constantes. (Note que para la frecuencia fuera de la nominal, la medición del ángulo de fase será cambiada aunque la fase φ de la señal de prueba sea constante)

Todas las pruebas del cumplimiento son bajo las condiciones de estado estacionario, con condiciones de referencia y cantidades de influencia mostradas en la tabla 4.8. Efectos de las cantidades de influencia deberán ser consideradas acumulativas, y el TVE no deberá exceder el error listado bajo los niveles de cumplimiento en cualquier combinación de influencia de cantidades mostradas en la

tabla 4.8 (TVE). Para evaluar el cumplimiento con estos requerimientos, el efecto de la influencia de cantidades puede ser evaluada por separado [43].

NOTA.- El requerimiento del cumplimiento del nivel deberá ser determinado por el usuario. Las condiciones de referencia especifican las condiciones para que cantidad cuando no hay variedad. Por ejemplo, cuando la magnitud de la señal de prueba, la magnitud debe ser variada pero la frecuencia deberá ser la nominal, el ángulo de fase deberá ser de 0 radianes, la distorsión armónica de menos de 0.2 %, y la señal de interferencia menos del 2%. El TVE deberá permanecer dentro del rango especificado para cada nivel durante las pruebas realizadas [43].

Para realizar pruebas de los niveles 0 y 1, se debe de variar solo un parámetro a la vez, es decir; cuando se realiza la prueba de magnitud, la magnitud de la señal debe ser variada, pero la frecuencia será nominal, el ángulo de fase 0 radianes, la distorsión armónica de menos del 0.2% y la señal de interferencia menos del 2%. El TVE debe permanecer dentro del margen de error en todas las pruebas [43].

Este estándar no impone cualquier limitación para el uso del PMU bajo cualquier condición. Los PMU son muy precisos para hacer mediciones bajo muchas condiciones de transición y se han realizado varias publicaciones sobre esto. El estándar no aborda la precisión y respuesta en el tiempo bajo condiciones de transición, y también todas las pruebas son estrictas para condiciones de estado estacionario [43].

C.6.- Formatos de mensaje del síncrofasor

Esta subcláusula describe el formato de mensaje hacia y del PMU para uso de comunicación en tiempo real del fasor de datos. La transmisión de datos en tiempo real es definida aquí como el proceso de tomar mediciones en un lugar simultáneamente. Si el dispositivo PMU es usado con otro sistema donde la información de los datos del fasor son transmitidos en tiempo real, la

implementación de este protocolo es requerido con conformidad con este estándar. Si el dispositivo PMU es usado solo como un concentrador de datos entonces este protocolo no es requerido. La implementación adicional del protocolo no es requerida o limitada [43].

Cualquier sistema de medición o comunicación puede ser usada para la transmisión de datos. La trama de mensajes deberá ser transmitida en su totalidad como esta especificado [43].

Este protocolo de mensaje podrá ser usado para comunicar un simple PMU o un sistema secundario que recibe datos del PMU. El sistema secundario será referido como un concentrador de datos (DC, también llamado un phasor data concentrador o PDC en algunas referencias), deberá tener su propio usuario asignando un código de ID. El protocolo permite identificar la información, tal como el IDCODE PMU y estatus [43].

C.6.1.- Estructura del mensaje

Cuatro tipos de mensaje son definidos aquí: datos, configuración, encabezado y comando. Los primeros tres mensajes son transmitidos del PMU/DC y el último (comando) es recibido por el PMU. El mensaje de datos son las mediciones hechas por el PMU. La configuración describe a los datos, el PMU envía y proporciona factores de calibración. El encabezado es información legible para el humano describe la información enviada por el PMU pero proporcionada por el usuario. Los comandos son códigos enviados para el PMU por el control o configuración. La información puede ser almacenada por el PMU, pero cuando es transmitida esta debe ser formateada como describe la trama en la siguiente subcláusula [43].

Solamente la trama de datos, configuración, encabezado y comandos son definidas en este estándar. Otros tipos pueden ser descritos en un futuro. En operación normal, el PMU solo envía trama de datos [43].

C.6.1.1.- Mensaje total

Todas las tramas de mensaje comienzan con la palabra SYNC de 2 bytes seguida por la palabra FRAME SIZE de 2 bytes, un IDCODE de 2 bytes, y un estampado de tiempo que consiste de 4 bytes del SOC y 4 bytes para la fracción de segundo (FRACSEC), que incluye una fracción de segundo de 24 bits y una bandera de calidad de tiempo de 8 bits. La palabra SYNC proporciona la sincronización y la identificación de la trama. El IDCODE identifica afirmativamente la unidad a la que se envió o recibió el mensaje. Los Bits 4-6 en la palabra SYNC designa el tipo de trama. Esta palabra es detallada en la tabla 4. Todas las tramas terminan con la palabra de chequeo (CHK), que es un CRC-CCITT. Este CRC-CCITT usa la generación del polinomio $x^{16} + x^{12} + x^5 + 1$ el cual se inicializa con el valor de -1 (FFFF). Todas las tramas son transmitidas exactamente como se describen sin limitantes. La figura 3 ilustra el orden de la trama de transmisión. La palabra SYNC es transmitida primero y la última palabra es la CHK. Todos los tipos de tramas son en este mismo orden y formato. Esta primera edición publicada de este estándar deberá ser designada como la versión 1 (binario 0001) [43].

C.6.1.2 Tiempo y calidad de tiempo.

El FRAMESEC de 32 bits (4 bytes) es dividido dentro de dos componentes: uno de 24 bits que es la actual fracción de segundo y uno de 8 bits que es la bandera de calidad de tiempo. La transmisión de la medición de datos y el estampado de tiempo para tramas de datos es el SOC estampado de datos fijado en segundos enteros más la fracción de tiempo. La fracción de tiempo es determinada dividiendo los 24 bits enteros entre la fracción de segundo por el TIME_BASE dado en la configuración de la trama [43].

Los bits de la calidad del tiempo indican el comienzo del reporte de la calidad del tiempo además de indicar el estado del salto de segundo. El bit 7 es reservado para usos futuros. El bit 4 es el salto de segundo pendiente y debe ser fijado tan pronto como este se conozca pero no menos que un segundo o mas de 59 segundos después del salto de segundo ocurrido. Este deberá ser limpiado en el primer segundo después de que el salto de segundo ocurra. El bit 5 es el salto de segundo ocurrido y deberá ser fijado en el primer segundo después de que el salto de segundo ocurre. El Bit 6 es la dirección del salto de segundo que es 0 Para agregar un segundo y 1 para borrar un segundo. Este deberá ser fijado (a 0 o 1 como se requiera) en el mismo tiempo o después del pendiente salto de segundo es fijado y seguirá siendo el mismo por lo menos después de 24 horas. Este permitir un programa de análisis para un factor en un $\pm$ salto de segundo en cualquier análisis o diferencia de tiempo calculado [43].

El contenido del código de la calidad de tiempo indica en los 4 bits menos significativos el error máximo de tiempo como lo determina la función del reloj del PMU. Los bits 0-3 deben ser limpiados a 0 cuando la función del tiempo es cerrada encima de su fuente. Los bits 0-3 deben ser fijados a 1 cuando hay cualquier error en el reloj o cuando nunca se ha inicializado [43].

C.6.1.3.- Ejemplos de salto de segundo para el estampado.

El siguiente ejemplo muestra como el conteo del tiempo y el salto de un segundo aparece para los casos donde es agregado y borrado. La dirección puede ser cualquier estado después de que el segundo pendiente es saltado es fijado y después ocurre la limpieza del bit. Cuando cada uno de esos bits es fijado, la dirección de los bits debe ser fijada en el estado propiamente indicado en un incremento o decremento. El bit pendiente debe ser fijado tan pronto como el salto del segundo ocurre es desconocido pero no mas de 59 segundos y no menos de 1 segundo después de que el cambio es ocurrido [43].

C.6.2 Trama de datos

La trama de datos deberá contener la medición de los datos y deberá ser identificada por tener los bits 4-6 en la palabra SYNC fijados a cero. La trama de datos del fasor en tiempo real consistirá en datos binarios ordenados y descritos a detalle como se muestra en la tabla C.2. Toda la longitud de los campos deberán ser fijados como se describe, y no delimitaran su uso. La trama comienza con SYNC, FRAMESIZE, IDCODE y SOC y termina con un CRC-CCITT [43].

C.6.2.1 Explicación de la palabra STAT en la trama de datos.

La trama de datos consiste en secciones de tiempo y datos. Los datos pueden ser un bloque de un simple PMU o múltiples bloques de PMUs. Cada bloque de datos del PMU es encabezado por una palabra STAT que tiene un estatus completo para cada bloque. Los bits son fijados inicialmente en esta bandera STAT por el PMU que genera los datos y puede ser alterado por otro proceso en la cadena de datos, tal como por un DC. La palabra STAT da un estatus completo para los datos en su bloque de datos dentro de los límites de este estándar.

Bits 15 y 12 normalmente sólo se utilizan por la DC y debe ser configurado a 0 por un PMU excepto cuando Bit 15 se utiliza para indicar que el PMU se encuentra en modo de prueba y emisión de datos de prueba. Bits 14, 13, y 11 serán normalmente establecidos por el PMU hacer la medición, pero podrá ser modificado según sea necesario [43].

Bit -15 de validación de datos: Fijado a 1 cuando el bloque de datos del PMU contiene datos no válidos. Que normalmente será 0 (cuando el bloque contiene datos válidos). Con el fin de retransmitir un marco coherente de datos, un DC seguirá con interés los datos en el mismo formato, aun cuando la entrada de un solo PMU. El bloque de datos para la entrada no estará presente, pero que contienen ya sea

rellenado en los datos o datos no válidos indicaciones, en este caso el bit se establece en 1 [43].

Bit -14 PMU Error: Un termino Muy general o que vale para todo; que el PMU se establecerán para indicar un error interno como calibración analógico a digital (A / D), error de memoria, error de procesador, etc. Dado que el PMU normalmente sólo enviar tramas de datos, este bit se indicará un error que puede invalidar la totalidad o parte de los datos; las condiciones exactas se definirán por el fabricante. Bit 14 normalmente será 0 [43].

Bit -13 PMU Error de Sync: puesto a 1 para indicar que el PMU ha detectado una pérdida de tiempo de sincronización externa como una pérdida de seguimiento por satélite o un IRIG-B error de conexión de entrada. Se utilizará tanto en el momento de la sincronización de tiempo de entrada y no cuando la fuente de sincronización de tiempo pierde tiempo de bloqueo. La medición del PMU establecerá este bit a 1 cuando cualquier bit en el 4-bit tiempo la calidad en el campo FRACSEC se convierte ámbito distinto de cero. A DC también podrá fijar a 1 el Bit 13 si se detecta un error de sincronización en el flujo de datos a partir de un PMU. La longitud de tiempo entre la detección de un error de sincronización y el establecimiento de Bit 13 a 1 no se superará el plazo estimado para el error en tiempo de superar el límite de TVE o 1 min, el que sea menor [43].

Bit -12 Tipo de clasificación de datos: puesto a 1 cuando los datos para el PMU no está integrado en la estructura de datos mediante el uso de su estampado de tiempo (timetag). Un concentrador normalmente integrar datos procedentes de un número de las unidades de gestión en un único marco de la timetags proporcionada por el PMU. Si un PMU en el grupo pierde la sincronización de tiempo externa durante un largo período de tiempo, timetag proporcionada por el PMU puede impedir esta integración o hacer que el tiempo peor que la aproximación utilizando

otro método de integración. Como una alternativa a simplemente descartar todos los datos, el concentrador puede incluir los datos en el marco usando una "mejor estimación" en cuanto a que el marco de su entrada, y advierten de la falta de correlación momento preciso mediante el establecimiento de 12 Bit. El enfoque más sencillo para el concentrador en un sistema en tiempo real es incluir la falta de sincronización de datos con la más sincronizada de datos, utilizando el supuesto de que los retrasos de comunicación de datos son iguales. Esta "especie-por la llegada" es un método más sencillo adivinar la alineación de datos. Otros métodos pueden ser utilizados. En todos los casos, Bit 12 se establece en 1 cuando los datos no se correlacionan en su marco por timetag y autorizó a 0 cuando los datos se correlacionan por timetag [43].

Bit -11 de activación PMU recoger: fijado para indicar una condición de activación se ha detectado por el PMU que tienen la capacidad de activación. Los bits se fijarán por lo menos un marco de datos o 1 s, lo que es más largo, y se mantendrá fijado, siempre que la condición de activación se detecta [43].

Bit -10 de configuración si ha cambiado el bit queda fijado a un 1 para indicar que la configuración del PMU ha cambiado. Este bit se restablece a 0 después de 1 min. Esto sirve como una indicación de que el dispositivo receptor debe solicitar el archivo de configuración para asegurarse de que los datos de configuración es actualizada. Para estar seguro de tener un archivo de configuración válido, el dispositivo receptor debe solicitar un archivo de configuración cada vez que ha sido fuera de línea durante más de un minuto.

Bit 4-5-tiempo abierto: Indica una serie de segundos desde la pérdida de sincronización se ha detectado [43].

Bits 0-3- razón de activación: 4-bits que indica el código inicial de una causa desencadenante. Véase el cuadro 8 para la codificación [43].

C.6.3 Configuración de trama

Una configuración de trama es una maquina legible de bits de datos fijados que contienen información y procesan parámetros por el PMU y el dato actual del tiempo real fijado. Este es identificado por los bits 4-6 de la palabra SYNC como se muestra en la tabla 4. Dos tipos de configuraciones son identificadas, CFG-1 con el bit 4 de la palabra SYNC fijado a 0 y CFG-2 con el bit 4 de la palabra SYNC fijado a 1. CFG-1 denota la capacidad del PMU indicando las mediciones que el PMU es capaz de hacer. CFG-2 indica las mediciones que actualmente esta haciendo el PMU y transmitiendo en la trama de datos. Este puede ser solo un subconjunto de disponibles datos. Ambas tramas tienen 19 campos, con 7-16 repeticiones como sea necesario. Todos los campos deberán ser fijados con la longitud como se describe, y no delimita como deben ser usados [43].

C.6.4 Encabezado

Esta trama deberá ser legible para el hombre y da la información del PMU, la fuente de datos, escala, algoritmos, filtros u otra información. La trama tiene lo mismo que las anteriores tramas, la palabra SYNC, FRAMSIZE, SOC y CHK y es identificado por los bits 4-6 de la palabra SYNC. LA sección de datos no tiene un formato fijado [43].

C.6.5 Trama de comandos.

Un PMU deberá ser capaz de recibir comandos de un sistema de control y tomar las acciones apropiadas. La trama tiene lo mismo que las anteriores tramas, la palabra SYNC, FRAMSIZE, SOC y CHK y es identificado por los bits 4-6 de la palabra SYNC. El IDCODE deberá ser de 2 bytes y es la identificación del PMU asignado a un PMU y es el mismo campo 3 en la trama de configuración. El CHK es el CRC-CCITT de 16 bits descrito previamente. El PMU deberá marcar el IDCODE con una reserva internamente después de aceptar y ejecutar el comando. El comando deberá ser de 2 bytes [43].

APÉNDICE D. Precisión y Exactitud.

D.1 Exactitud

Aproximación con la cual la lectura de un instrumento se acerca al valor real de la variable medida [52]. La exactitud de un instrumento es el máximo error que el usuario espera del instrumento [53].

D.2 Precisión

Medida de la reproducibilidad de las mediciones [53]; esto es, dado el valor fijo de una variable, la precisión es una medida del grado con el cual las mediciones sucesivas difieren una de otra. Indica una zona dentro de la cual está comprendido el valor real de la magnitud [52, 53].

La precisión es normalmente caracterizada en términos de desviación estándar de las medidas [53]. La desviación estándar es una medida del grado de dispersión de los datos del valor promedio o la variación esperada con respecto de la media aritmética. Una desviación estándar grande indica que los puntos están lejos de la media y una desviación pequeña indica que los datos están agrupados cerca de la media.

D.3 Resolución

Cambio más pequeño en el valor medido al cual responde el instrumento [53].

APÉNDICE E.- Código implementado

E.1 Programa Principal.

```
/** #####
**      Filename   : PMU.C
**      Project    : PMU
**      Processor  : MCF52233CAF60
**      Version    : Driver 01.00
**      Compiler   : CodeWarrior MCF C Compiler
**      Date/Time  : 19/01/2009, 05:10 p.m.
**      Abstract   :
**                  Main module.
**                  This module contains user's application code.
**      Settings   :
**      Contents   :
**                  No public methods
**
**      (c) Copyright UNIS, a.s. 1997-2008
**      UNIS, a.s.
**      Jundrovská 33
**      624 00 Brno
**      Czech Republic
**      http      : www.processorexpert.com
**      mail      : info@processorexpert.com
** #####*/
/* MODULE PMU */

/* Including needed modules to compile this module/procedure */
#include "Cpu.h"
#include "Events.h"
#include "AD1.h"
#include "Bit1.h"
#include "Bit2.h"
#include "TI1.h"
#include "TI2.h"
#include "TI3.h"
#include "estandard.h"
#include "Bit3.h"
#include "Bit4.h"
#include "EInt1.h"
/* Including shared modules, which are used for whole project */
#include "PE_Types.h"
#include "PE_Error.h"
#include "PE_Const.h"
#include "IO_Map.h"

#include "tablas.h"
#include "funciones_standard.h"

word tam_paquete;
int N_2;
void main(void)
{
    /* Write your local variable definition here */

    /**** Processor Expert internal initialization. DON'T REMOVE THIS CODE!!! ****/
    PE_low_level_init();
    /**** End of Processor Expert internal initialization. ****/

    /* Write your code here */
    /* For example: */
    N_2=No_muestras/2;
    tam_paquete=No_muestras;

    for(;;)
    {
        if (ContRx>0 && timer_rx==0)
        {
            Bit3_NegVal();
            PMU_standard();
        }
    }

    /**** Don't write any code pass this line, or it will be deleted during code generation. ****/
    /**** Processor Expert end of main routine. DON'T MODIFY THIS CODE!!! ****/
    for(;;){}
    /**** Processor Expert end of main routine. DON'T WRITE CODE BELOW!!! ****/
    /**** End of main routine. DO NOT MODIFY THIS TEXT!!! ****/
}

/* END PMU */
/*
** #####
**
**      This file was created by UNIS Processor Expert 1.01 [04.08]
**      for the Freescale MCF series of microcontrollers.
** #####
*/
```

E.2 Eventos.

```

/** *****
**      Filename   : Events.C
**      Project    : PMU
**      Processor  : MCF52233CAF60
**      Beantype   : Events
**      Compiler   : CodeWarrior MCF C Compiler
**      Date/Time  : 19/01/2009, 05:10 p.m.
**      Abstract   :
**                  This is user's event module.
**                  Put your event handler code here.
**      Settings   :
**      Contents   :
**                  TI3_OnInterrupt      - void TI3_OnInterrupt(void);
**                  TI2_OnInterrupt      - void TI2_OnInterrupt(void);
**                  TI1_OnInterrupt      - void TI1_OnInterrupt(void);
**                  AD1_OnEnd            - void AD1_OnEnd(void);
**                  Cpu_OnCoreWatchdogINT - void Cpu_OnCoreWatchdogINT(void);
**
**      (c) Copyright UNIS, a.s. 1997-2008
**      UNIS, a.s.
**      Jundrovská 33
**      624 00 Brno
**      Czech Republic
**      http      : www.processorexpert.com
**      mail      : info@processorexpert.com
** *****
** MODULE Events */

#include "Cpu.h"
#include "Events.h"
#include "tablas.h"

#define T_ESPERA 18

unsigned char Buffer_RX[T_ESPERA];
byte err1=1, bandera=0;
word ContRx=0, timer_rx;
int contador=0,k=0;
int fase_A_V_r=0,fase_A_V_i=0, fase_B_V_r=0,fase_B_V_i=0, fase_C_V_r=0,fase_C_V_i=0;
int fase_A_I_r=0,fase_A_I_i=0, fase_B_I_r=0,fase_B_I_i=0, fase_C_I_r=0,fase_C_I_i=0;
int con_cal_frec=0;
int suma_muestras=0,fraccion_mues_new=0,fraccion_mues_act=0;
int Frecuencia=0;
word values[No_canales];
int values1[No_canales];
//uint16
int muestra_act=0, muestra_ant=0;
int i, PPS=1, PPS_B=0,a;
byte comp_simetricas=0;//Si es 1 se envian componentes simetricas
byte formato_polar=0;// Si es 1 se envia magnitud y angulo, si es 0 se envia real e imaginario

extern int coseno[], seno[];
extern byte conf_enviada;
extern unsigned int size_env_data_on;

int ex_r=0, ex_i=0;

/*
** =====
**      Event      : TI2_OnInterrupt (module Events)
**
**      From bean   : TI2 [TimerInt]
**      Description :
**                  When a timer interrupt occurs this event is called (only
**                  when the bean is enabled - <Enable> and the events are
**                  enabled - <EnableEvent>). This event is enabled only if a
**                  <interrupt service/event> is enabled.
**      Parameters  : None
**      Returns     : Nothing
** =====
*/
for(;;){}
/** Processor Expert end of main routine. DON'T WRITE CODE BELOW!!! ***/
} /** End of main routine. DO NOT MODIFY THIS TEXT!!! ***/

/* END PMU */
/*
** *****
**      This file was created by UNIS Processor Expert 1.01 [04.08]
**      for the Freescale MCF series of microcontrollers.
** *****
*/
void TI2_OnInterrupt(void)
{
    /* Write your code here ... */
    if (timer_rx)
    {
        timer_rx--;
        Bit1_NegVal();
    }
}

```

```

** =====
** Event      :  estandard_OnRxChar (module Events)
**
** From bean   :  estandard [AsynchroSerial]
** Description :
**     This event is called after a correct character is
**     received.
**     The event is available only when the <Interrupt
**     service/event> property is enabled and either the
**     <Receiver> property is enabled or the <SCI output mode>
**     property (if supported) is set to Single-wire mode.
** Parameters  :  None
** Returns    :  Nothing
** =====
*/
void estandard_OnRxChar(void)
{
    /* Write your code here ... */
    byte temp;
    timer_rx = T_ESPERA;
    temp = estandard_RecvChar(&Buffer_RX[ContRx++]);
    if (ContRx>T_ESPERA)
        ContRx=0;
}

/* =====
** Event      :  TI1_OnInterrupt (module Events)
**
** From bean   :  TI1 [TimerInt]
** Description :
**     When a timer interrupt occurs this event is called (only
**     when the bean is enabled - <Enable> and the events are
**     enabled - <EnableEvent>). This event is enabled only if a
**     <interrupt service/event> is enabled.
** Parameters  :  None
** Returns    :  Nothing
** =====
*/
void TI1_OnInterrupt(void)
{
    /* Write your code here ... */
    if(conf_enviada==1)
    {
        Bit2_NegVal();
        if (contador==No_muestras)
        {
            Bit2_NegVal();

            //Calcula la DFT para calcular los fasores
            fasores_ABC();

            if (comp_simetricas==1)//Si se requiere enviar las componentes simetricas
                //a partir de los fasores ABC
            {
                componentes_simetricas();
            }

            if(formato_polar==1)//Si se requiere enviar en formato polar los fasores ABC
                //o las componentes simetricas.
            {
                fasor();
            }

            fase_A_V_r=0;
            fase_A_V_i=0;
            fase_B_V_r=0;
            fase_B_V_i=0;
            fase_C_V_r=0;
            fase_C_V_i=0;
            fase_A_I_r=0;
            fase_A_I_i=0;
            fase_B_I_r=0;
            fase_B_I_i=0;
            fase_C_I_r=0;
            fase_C_I_i=0;

            k=0;
            contador=0;
        }
        inicia_txd(size_env_data_on);
    }
}

```

```

/*
** =====
**      Event      :   AD1_OnEnd (module Events)
**
**      From bean   :   AD1 [ADC]
**      Description :
**          This event is called after the measurement (which consists
**          of <1 or more conversions>) is/are finished.
**          The event is available only when the <Interrupt
**          service/event> property is enabled.
**      Parameters  :   None
**      Returns     :   Nothing
** =====
*/
void AD1_OnEnd(void)
{
    /* Write your code here ...*/
    if (PPS)
    {
        if (contador<No_muestras)
        {
            AD1_GetValue16(&values[0]);
            for (i=0;i<No_canales;i++)
            {
                values1[i]=(values[i]>>4)&(0x0fff);
                values1[i]=values1[i]-offset[i];
            }

            fase_A_V_r=(int)values1[0]*coseno[k]+fase_A_V_r;
            fase_A_V_i=(int)values1[0]*seno[k]+fase_A_V_i;

            fase_B_V_r=(int)values1[1]*coseno[k]+fase_B_V_r;
            fase_B_V_i=(int)values1[1]*seno[k]+fase_B_V_i;

            fase_C_V_r=(int)values1[2]*coseno[k]+fase_C_V_r;
            fase_C_V_i=(int)values1[2]*seno[k]+fase_C_V_i;

            fase_A_I_r=(int)values1[3]*coseno[k]+fase_A_I_r;
            fase_A_I_i=(int)values1[3]*seno[k]+fase_A_I_i;

            fase_B_I_r=(int)values1[4]*coseno[k]+fase_B_I_r;
            fase_B_I_i=(int)values1[4]*seno[k]+fase_B_I_i;

            fase_C_I_r=(int)values1[5]*coseno[k]+fase_C_I_r;
            fase_C_I_i=(int)values1[5]*seno[k]+fase_C_I_i;
            k++;
            //calcula frecuencia de la señal, con el primar dato erroneo.
            if(con_cal_frec==7)//7 si es cada 8 muestras (simula una señal de 16 muestras)
                                //15 si es cada 16 muestras (simula una señal de 8 muestras)
                                //3 si es cada 4 muestras (simula una señal de 32 muestras)
            {
                muestra_act=muestra_ant;
                muestra_act=(int)values1[0];
                if(muestra_act<0 & muestra_ant>0)
                {
                    muestra_act=-muestra_act;
                    fraccion_mues_new=(muestra_act*10000)/(muestra_act+muestra_ant);
                    fraccion_mues_act=10000-fraccion_mues_new;
                    suma_muestras=suma_muestras+fraccion_mues_act;
                    Frecuencia=Inv_periodo_frec/suma_muestras;
                    suma_muestras=fraccion_mues_new;
                }
                else
                {
                    suma_muestras=suma_muestras+10000;
                }

                con_cal_frec=0;
            }
            else
            {
                con_cal_frec++;
            }
            contador++;
        }
    }
}

```

```

/*
** =====
**      Event      :  TI3_OnInterrupt (module Events)
**
**      From bean   :  TI3 [TimerInt]
**      Description :
**          When a timer interrupt occurs this event is called (only
**          when the bean is enabled - <Enable> and the events are
**          enabled - <EnableEvent>). This event is enabled only if a
**          <interrupt service/event> is enabled.
**      Parameters  :  None
**      Returns     :  Nothing
** =====
*/
void TI3_OnInterrupt(void)
{
    /* Write your code here ... */
    if (contador<No_muestras)
    {
        Bit4_NegVal();
        AD1_Measure(FALSE);
    }
}

/*
** =====
**      Event      :  EInt1_OnInterrupt (module Events)
**
**      From bean   :  EInt1 [ExtInt]
**      Description :
**          This event is called when an active signal edge/level has
**          occurred.
**      Parameters  :  None
**      Returns     :  Nothing
** =====
*/
void EInt1_OnInterrupt(void)
{
    /* place your EInt1 interrupt procedure body here */

    PPS=1;
    PPS_B=1;
    a=1;
    Bit1_NegVal();
}

/*
** =====
**      Event      :  Cpu_OnCoreWatchdogINT (module Events)
**
**      From bean   :  Cpu [MCF52233_80_LQFP]
**      Description :
**          This event is called when the OnCoreWatchdog interrupt had
**          occurred. This event is automatically enabled when the <Mode>
**          is set to 'Interrupt'.
**      Parameters  :  None
**      Returns     :  Nothing
** =====
*/
void Cpu_OnCoreWatchdogINT(void)
{
    /* Write your code here ... */
}

/* END Events */

/*
** #####
**
**      This file was created by UNIS Processor Expert 1.01 [04.08]
**      for the Freescale MCF series of microcontrollers.
**
** #####
*/

```

E.3 Funciones.

```
#include "funciones_standard.h"

#include "estandard.h"

#include "tablas.h"

unsigned short crc_tem;
unsigned char tam_conf_enviada=0;
unsigned int size_env_data_on,num_bytes_enviar;

int real_V1,imag_V1,magn_V1,angle_V1;
int real_V2,imag_V2,magn_V2,angle_V2;
int real_V3,imag_V3,magn_V3,angle_V3;
int real_I1,imag_I1,magn_I1,angle_I1;
int real_I2,imag_I2,magn_I2,angle_I2;
int real_I3,imag_I3,magn_I3,angle_I3;
unsigned int frecuencial2=96;
long int ang,ang1;
int x, l,angle_V1_t;
int x1,x2,v1,v2;
/*~~~~~
Funcion para calcular el CRC de un vector recibido
~~~~~*/
uint16_t CRC16_CCITT(unsigned char *Buffer,unsigned int NumBytes)
{
    uint16_t crc = 0xFFFF;
    // uint16_t asi viene en el standard, con este tipo de en lugar de int
    int temp, quick;
    int i;
    for (i=0;i<NumBytes;i++)
    {
        temp = (crc>>8)^ Buffer[i];
        crc <<= 8;
        quick = temp ^ (temp >> 4);
        crc ^= quick;
        quick <<=5;
        crc ^= quick;
        quick <<=5;
        crc ^= quick;
        quick <<=7;
        crc ^= quick;
    }
    return crc;
}

/*~~~~~
Funcion que identifica el comando requerido por el maestro.
~~~~~*/
void PMU_standard(void)
{
    tam=18;
    if (CRC16_CCITT(&Buffer_RX[0],tam)==0)// Si no hay error en la recepción
    {
        crc_tem=CRC16_CCITT(&Buffer_RX[0],16);
        if (Buffer_RX[5] == NUM_ESCLAVO)// Si es mi ID_MODBUS contesto
        {
            switch (Buffer_RX[15])
            {
                case CMD_DATA_OFF:
                    conf_enviada=0;
                    break;

                case CMD_DATA_ON:
                    size_env_data_on=TAMTXDAT-2;
                    conf_enviada=1;
                    break;

                case CMD_SEND_HDR:
                    conf_enviada=0;
                    break;

                case CMD_SEND_CFG1:
                    conf_enviada=0;
                    num_bytes_enviar=TAMTXCONF-2;
                    inicia_tx(num_bytes_enviar);
                    break;

                case CMD_SEND_CFG2:
                    conf_enviada=0;
                    num_bytes_enviar=TAMTXCONF-2;
                    inicia_tx(num_bytes_enviar);
                    break;

                case CMD_EXT_FRAME:
                    conf_enviada=0;
                    break;

                default:
                    break;
            }
        }
    }
    ContRx = 0; // Ya terminó de atender este comando, espera siguiente
}
```

```

/*
Transmite los datos en tiempo real del PMU de la estación SEPI.
*/
void inicia_txd(unsigned int size_TX_dat)
{
    unsigned int temp,j;
    temp =CRC16_CCITT(&Buffer_TX_resp[0],size_TX_dat);
    Buffer_TX_resp[size_TX_dat] = (unsigned char)(temp>>8 & 0xFF); // Parte alta del CRC
    Buffer_TX_resp[size_TX_dat+1] = (unsigned char)(temp & 0xFF); // Parte baja del CRC
    for(j = 0; j < TAMTXDAT; j++)
    {
        while(standard_SendChar((byte)Buffer_TX_resp[j]) != ERR_OK) {}
        // Trata de enviar un caracter hasta que el metodo sendChar regresa ERROR_OK
    }
}

/*
Transmite la configuración del PMU de la estación SEPI.
*/
void inicia_tx(unsigned int size_TX_conf)
{
    unsigned int temp,j;
    temp =CRC16_CCITT(&Buffer_TX_conf[0],size_TX_conf);
    Buffer_TX_conf[size_TX_conf] = (unsigned char)(temp>>8 & 0xFF); // Parte alta del CRC
    Buffer_TX_conf[size_TX_conf+1] = (unsigned char)(temp & 0xFF); // Parte baja del CRC
    for(j = 0; j < TAMTXCONF; j++)
    {
        while(standard_SendChar((byte)Buffer_TX_conf[j]) != ERR_OK) {}
        // Trata de enviar un caracter hasta que el metodo sendChar regresa ERROR_OK
    }
}

/*
Comienza a calcular la raiz de una variable
*/
int raiz(int x)
{
    int r=x,t=0;
    if (x>0)
    {
        do
        {
            t=r;
            r=(x/r+r)/2;
        }
        while (t>r);
    }
    return t;
}

/*
Calcular el angulo de los fasores
*/
int angulo (int re, int mag, int im)
{
    x=(re*frac_12s*10)/mag;
    if (x<0)
        x=-x;
    for (l=0;l<No_muestras_ang;l++)
    {
        if(x==Tab_identifica[l])
        {
            ang1=Tab_acos[l];
        }
        else
        {
            if (x<Tab_identifica[l])//if dos
            {
                x1=Tab_identifica[l];
                x2=Tab_identifica[l-1];
                y1=Tab_acos[l];
                y2=Tab_acos[l-1];
                ang1=((y2-y1)/(x2-x1))*(x-x1)+y1;
            }
            //End If dos
        }
        //End Else
    }
    //End for
    if (im>0 & re==0)
        ang=90000;
    if (re<0 & im>0)
        ang=180000-ang1;
    if (re<0 & im<0)
        ang=ang1-180000;
    if (re>0 & im<0)
        ang=-ang1;

    return ang;
}
//End Función

```

```

/*%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
%%  Calcular real e imaginario de los fasores (A,B y C)
%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%*/

void fasores_ABC(void)
{
    //Calculo de real e imaginario para el fasor A de voltaje
    |
    real_V1=fase_A_V_r/frac_12;
    real_V1=real_V1/N_2;
    imag_V1=fase_A_V_i/frac_12;
    imag_V1=imag_V1/N_2;

    Buffer_TX_resp[16]=(unsigned char)(real_V1>>8 & 0xFF); // Parte alta de la parte real
    Buffer_TX_resp[17]=(unsigned char)(real_V1 & 0xFF); // Parte baja de la parte real
    Buffer_TX_resp[18]=(unsigned char)(imag_V1>>8 & 0xFF); // Parte alta de la parte imaginaria
    Buffer_TX_resp[19]=(unsigned char)(imag_V1 & 0xFF); // Parte baja de la parte imaginaria

    //Calculo de real e imaginario para el fasor B de voltaje
    real_V2=fase_B_V_r/frac_12;
    real_V2=real_V2/N_2;
    imag_V2=fase_B_V_i/frac_12;
    imag_V2=imag_V2/N_2;

    Buffer_TX_resp[20]=(unsigned char)(real_V2>>8 & 0xFF); // Parte alta de la parte real
    Buffer_TX_resp[21]=(unsigned char)(real_V2 & 0xFF); // Parte baja de la parte real
    Buffer_TX_resp[22]=(unsigned char)(imag_V2>>8 & 0xFF); // Parte alta de la parte imaginaria
    Buffer_TX_resp[23]=(unsigned char)(imag_V2 & 0xFF); // Parte baja de la parte imaginaria

    //Calculo de real e imaginario para el fasor C de voltaje
    real_V3=fase_C_V_r/frac_12;
    real_V3=real_V3/N_2;
    imag_V3=fase_C_V_i/frac_12;
    imag_V3=imag_V3/N_2;

    Buffer_TX_resp[24]=(unsigned char)(real_V3>>8 & 0xFF); // Parte alta de la parte real
    Buffer_TX_resp[25]=(unsigned char)(real_V3 & 0xFF); // Parte baja de la parte real
    Buffer_TX_resp[26]=(unsigned char)(imag_V3>>8 & 0xFF); // Parte alta de la parte imaginaria
    Buffer_TX_resp[27]=(unsigned char)(imag_V3 & 0xFF); // Parte baja de la parte imaginaria

    //Calculo de real e imaginario para el fasor A de corriente
    real_I1=fase_A_I_r/frac_12;
    real_I1=real_I1/N_2;
    imag_I1=fase_A_I_i/frac_12;
    imag_I1=imag_I1/N_2;

    Buffer_TX_resp[28]=(unsigned char)(real_I1>>8 & 0xFF); // Parte alta de la parte real
    Buffer_TX_resp[29]=(unsigned char)(real_I1 & 0xFF); // Parte baja de la parte real
    Buffer_TX_resp[30]=(unsigned char)(imag_I1>>8 & 0xFF); // Parte alta de la parte imaginaria
    Buffer_TX_resp[31]=(unsigned char)(imag_I1 & 0xFF); // Parte baja de la parte imaginaria

    //Calculo de real e imaginario para el fasor B de corriente
    real_I2=fase_B_I_r/frac_12;
    real_I2=real_I2/N_2;
    imag_I2=fase_B_I_i/frac_12;
    imag_I2=imag_I2/N_2;

    Buffer_TX_resp[32]=(unsigned char)(real_I2>>8 & 0xFF); // Parte alta de la parte real
    Buffer_TX_resp[33]=(unsigned char)(real_I2 & 0xFF); // Parte baja de la parte real
    Buffer_TX_resp[34]=(unsigned char)(imag_I2>>8 & 0xFF); // Parte alta de la parte imaginaria
    Buffer_TX_resp[35]=(unsigned char)(imag_I2 & 0xFF); // Parte baja de la parte imaginaria

    //Calculo de real e imaginario para el fasor C de corriente
    real_I3=fase_C_I_r/frac_12;
    real_I3=real_I3/N_2;
    imag_I3=fase_C_I_i/frac_12;
    imag_I3=imag_I3/N_2;

    Buffer_TX_resp[36]=(unsigned char)(real_I3>>8 & 0xFF); // Parte alta de la parte real
    Buffer_TX_resp[37]=(unsigned char)(real_I3 & 0xFF); // Parte baja de la parte real
    Buffer_TX_resp[38]=(unsigned char)(imag_I3>>8 & 0xFF); // Parte alta de la parte imaginaria
    Buffer_TX_resp[39]=(unsigned char)(imag_I3 & 0xFF); // Parte baja de la parte imaginaria
}

```



```

/*~~~~~
&&
&& Calcular real e imaginario de las componentes simetricas (+, - y 0)
&&
~~~~~*/

void componentes_simetricas(void)
{
    int real_V1_p, imag_V1_p, real_V2_n, imag_V2_n, real_V3_c, imag_V3_c;
    int real_I1_p, imag_I1_p, real_I2_n, imag_I2_n, real_I3_c, imag_I3_c;

    //Calculo de real e imaginario para la componente simetrica "cero" de voltaje

    real_V3_c=UN_TERCIO*(real_V1+real_V2+real_V3);
    imag_V3_c=UN_TERCIO*(imag_V1+imag_V2+imag_V3);

    Buffer_TX_resp[16]=(unsigned char)(real_V3_c>>8 & 0xFF); // Parte alta de la parte real
    Buffer_TX_resp[17]=(unsigned char)(real_V3_c & 0xFF); // Parte baja de la parte real
    Buffer_TX_resp[18]=(unsigned char)(imag_V3_c>>8 & 0xFF); // Parte alta de la parte imaginaria
    Buffer_TX_resp[19]=(unsigned char)(imag_V3_c & 0xFF); // Parte baja de la parte imaginaria

    //Calculo de real e imaginario para la componente simetrica "positiva" de voltaje

    real_V1_p=real_V1*UN_TERCIO+real_V2*A_real-imag_V2*A_imag+real_V3*A2_real-imag_V3*A2_imag;
    imag_V1_p=imag_V1*UN_TERCIO+imag_V2*A_real+real_V2*A2_imag+real_V3*A2_imag+imag_V3*A2_real;

    Buffer_TX_resp[20]=(unsigned char)(real_V1_p>>8 & 0xFF); // Parte alta de la parte real
    Buffer_TX_resp[21]=(unsigned char)(real_V1_p & 0xFF); // Parte baja de la parte real
    Buffer_TX_resp[22]=(unsigned char)(imag_V1_p>>8 & 0xFF); // Parte alta de la parte imaginaria
    Buffer_TX_resp[23]=(unsigned char)(imag_V1_p & 0xFF); // Parte baja de la parte imaginaria

    //Calculo de real e imaginario para la componente simetrica "negativa" de voltaje

    real_V2_n=real_V1*UN_TERCIO+real_V2*A2_real-imag_V2*A2_imag+real_V3*A_real-imag_V3*A_imag;
    imag_V2_n=imag_V1*UN_TERCIO+imag_V2*A2_real+real_V2*A2_imag+real_V3*A_imag+imag_V3*A_real;

    Buffer_TX_resp[24]=(unsigned char)(real_V2_n>>8 & 0xFF); // Parte alta de la parte real
    Buffer_TX_resp[25]=(unsigned char)(real_V2_n & 0xFF); // Parte baja de la parte real
    Buffer_TX_resp[26]=(unsigned char)(imag_V2_n>>8 & 0xFF); // Parte alta de la parte imaginaria
    Buffer_TX_resp[27]=(unsigned char)(imag_V2_n & 0xFF); // Parte baja de la parte imaginaria

    //Calculo de real e imaginario para la componente simetrica "cero" de corriente

    real_I3_c=UN_TERCIO*(real_I1+real_I2+real_I3);
    imag_I3_c=UN_TERCIO*(imag_I1+imag_I2+imag_I3);

    Buffer_TX_resp[28]=(unsigned char)(real_I3_c>>8 & 0xFF); // Parte alta de la parte real
    Buffer_TX_resp[29]=(unsigned char)(real_I3_c & 0xFF); // Parte baja de la parte real
    Buffer_TX_resp[30]=(unsigned char)(imag_I3_c>>8 & 0xFF); // Parte alta de la parte imaginaria
    Buffer_TX_resp[31]=(unsigned char)(imag_I3_c & 0xFF); // Parte baja de la parte imaginaria

    //Calculo de real e imaginario para la componente simetrica "positiva" de corriente

    real_I1_p=real_V1*UN_TERCIO+real_I2*A_real-imag_I2*A_imag+real_I3*A2_real-imag_I3*A2_imag;
    imag_I1_p=imag_V1*UN_TERCIO+imag_I2*A_real+real_I2*A2_imag+real_I3*A2_imag+imag_I3*A2_real;

    Buffer_TX_resp[32]=(unsigned char)(real_I1_p>>8 & 0xFF); // Parte alta de la parte real
    Buffer_TX_resp[33]=(unsigned char)(real_I1_p & 0xFF); // Parte baja de la parte real
    Buffer_TX_resp[34]=(unsigned char)(imag_I1_p>>8 & 0xFF); // Parte alta de la parte imaginaria
    Buffer_TX_resp[35]=(unsigned char)(imag_I1_p & 0xFF); // Parte baja de la parte imaginaria

    //Calculo de real e imaginario para la componente simetrica "negativa" de corriente

    real_I2_n=real_V1*UN_TERCIO+real_I2*A2_real-imag_I2*A2_imag+real_I3*A_real-imag_I3*A_imag;
    imag_I2_n=imag_V1*UN_TERCIO+imag_I2*A2_real+real_I2*A2_imag+real_I3*A_imag+imag_I3*A_real;

    Buffer_TX_resp[36]=(unsigned char)(real_I2_n>>8 & 0xFF); // Parte alta de la parte real del
    Buffer_TX_resp[37]=(unsigned char)(real_I2_n & 0xFF); // Parte baja de la parte real del
    Buffer_TX_resp[38]=(unsigned char)(imag_I2_n>>8 & 0xFF); // Parte alta de la parte imaginaria
    Buffer_TX_resp[39]=(unsigned char)(imag_I2_n & 0xFF); // Parte baja de la parte imaginaria

    real_V1=real_V3_c;
    imag_V1=imag_V3_c;
    real_V2=real_V1_p;
    imag_V2=imag_V1_p;
    real_V3=real_V2_n;
    imag_V3=imag_V2_n;
    real_I1=real_I3_c;
    imag_I1=imag_I3_c;
    real_I2=real_I1_p;
    imag_I2=imag_I1_p;
    real_I3=real_I2_n;
    imag_I3=imag_I2_n;
}

```

```

/*%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
&&
&&  Calcular angulo y magnitud
&&
&&%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%*/

void fasor(void)
{
    //Calcula magnitud y angulo para el fasor A de voltaje

    magn_V1=(real_V1*real_V1)+(imag_V1*imag_V1);
    magn_V1=raiz(magn_V1);
    angle_V1=angulo(real_V1,magn_V1,imag_V1 );

    Buffer_TX_resp[16]=(unsigned char)(magn_V1>>8 & 0xFF); // Parte alta de la parte real
    Buffer_TX_resp[17]=(unsigned char)(magn_V1 & 0xFF); // Parte baja de la parte real
    Buffer_TX_resp[18]=(unsigned char)(angle_V1>>8 & 0xFF); // Parte alta de la parte imaginaria
    Buffer_TX_resp[19]=(unsigned char)(angle_V1 & 0xFF); // Parte baja de la parte imaginaria

    //Calcula magnitud y angulo para el fasor B de voltaje

    magn_V2=(real_V1*real_V2)+(imag_V1*imag_V2);
    magn_V2=raiz(magn_V2);
    angle_V2=angulo(real_V2,magn_V2,imag_V2 );

    Buffer_TX_resp[20]=(unsigned char)(magn_V2>>8 & 0xFF); // Parte alta de la parte real
    Buffer_TX_resp[21]=(unsigned char)(magn_V2 & 0xFF); // Parte baja de la parte real
    Buffer_TX_resp[22]=(unsigned char)(angle_V2>>8 & 0xFF); // Parte alta de la parte imaginaria
    Buffer_TX_resp[23]=(unsigned char)(angle_V2 & 0xFF); // Parte baja de la parte imaginaria

    //Calcula magnitud y angulo para el fasor C de voltaje

    magn_V3=(real_V1*real_V3)+(imag_V3*imag_V3);
    magn_V3=raiz(magn_V3);
    angle_V3=angulo(real_V3,magn_V3,imag_V3 );

    Buffer_TX_resp[24]=(unsigned char)(magn_V3>>8 & 0xFF); // Parte alta de la parte real
    Buffer_TX_resp[25]=(unsigned char)(magn_V3 & 0xFF); // Parte baja de la parte real
    Buffer_TX_resp[26]=(unsigned char)(angle_V3>>8 & 0xFF); // Parte alta de la parte imaginaria
    Buffer_TX_resp[27]=(unsigned char)(angle_V3 & 0xFF); // Parte baja de la parte imaginaria

    //Calcula magnitud y angulo para el fasor A de corriente

    magn_I1=(real_I1*real_I1)+(imag_I1*imag_I1);
    magn_I1=raiz(magn_I1);
    angle_I1=angulo(real_I1,magn_I1,imag_I1 );

    Buffer_TX_resp[28]=(unsigned char)(magn_I1>>8 & 0xFF); // Parte alta de la parte real
    Buffer_TX_resp[29]=(unsigned char)(magn_I1 & 0xFF); // Parte baja de la parte real
    Buffer_TX_resp[30]=(unsigned char)(angle_I1>>8 & 0xFF); // Parte alta de la parte imaginaria
    Buffer_TX_resp[31]=(unsigned char)(angle_I1 & 0xFF); // Parte baja de la parte imaginaria

    //Calcula magnitud y angulo para el fasor B de corriente

    magn_I2=(real_I1*real_I2)+(imag_I1*imag_I2);
    magn_I2=raiz(magn_I2);
    angle_I2=angulo(real_I2,magn_I2,imag_I2 );

    Buffer_TX_resp[32]=(unsigned char)(magn_I2>>8 & 0xFF); // Parte alta de la parte real
    Buffer_TX_resp[33]=(unsigned char)(magn_I2 & 0xFF); // Parte baja de la parte real
    Buffer_TX_resp[34]=(unsigned char)(angle_I2>>8 & 0xFF); // Parte alta de la parte imaginaria
    Buffer_TX_resp[35]=(unsigned char)(angle_I2 & 0xFF); // Parte baja de la parte imaginaria

    //Calcula magnitud y angulo para el fasor C de corriente

    magn_I3=(real_I1*real_I3)+(imag_I3*imag_I3);
    magn_I3=raiz(magn_I3);
    angle_I3=angulo(real_I3,magn_I3,imag_I3 );

    Buffer_TX_resp[36]=(unsigned char)(magn_I3>>8 & 0xFF); // Parte alta de la parte real
    Buffer_TX_resp[37]=(unsigned char)(magn_I3 & 0xFF); // Parte baja de la parte real
    Buffer_TX_resp[38]=(unsigned char)(angle_I3>>8 & 0xFF); // Parte alta de la parte imaginaria
    Buffer_TX_resp[39]=(unsigned char)(angle_I3 & 0xFF); // Parte baja de la parte imaginaria

}

real_I2=real_I1_p;
imag_I2=imag_I1_p;
real_I3=real_I2_n;
imag_I3=imag_I2_n;
}

```